У. М. Котаў А. І. Лапо А. М. Вайцеховіч

ІНФАРМАТЫКА

Вучэбны дапаможнік для 7 класа
ўстаноў агульнай сярэдняй адукацыі
з беларускай мовай навучання

Дапушчана
Міністэрствам адукацыі
Рэспублікі Беларусь

Мінск «Народная асвета» 2017

УДК 004(075.3=161.3)
ББК 32.81я721
К73

Пераклад з рускай мовы *К. І. Даніленка*

Рэцэнзенты:
кафедра інфармацыйных тэхналогій у культуры факультэта культуралогіі і сацыякультурнай дзейнасці ўстановы адукацыі «Беларускі дзяржаўны ўніверсітэт культуры і мастацтваў» (кандыдат фізіка-матэматычных навук, дацэнт, загадчык кафедры *П. У. Глякаў*); настаўнік інфарматыкі вышэйшай кваліфікацыйнай катэгорыі дзяржаўнай установы адукацыі «Гімназія № 25 г. Мінска» *М. Ю. Сімакова*

ISBN 978-985-03-2892-2

ЗМЕСТ

Г л а в а 4. Апаратнае і праграмнае забеспячэнне камп'ютара

Г л а в а 5. Работа з вектарнай графікай

Ад аўтараў

Дарагія сямікласнікі! Вы трымаеце ў руках вучэбны дапаможнік па інфарматыцы — прадмеце, вывучэнне якога дазволіць вам атрымаць неабходныя веды і ўменні ў галіне інфармацыйных тэхналогій, што сталі неад'емнай часткай нашага жыцця.

У цяперашні час даследаванні па інфарматыцы асабліва запатрабаваныя і актуальныя. Роля гэтай дысцыпліны ва ўмовах сучаснага свету ўсё больш узрастае. Мы, аўтары вучэбнага дапаможніка, паспрабавалі зрабіць так, каб вывучэнне інфарматыкі было для вас цікавым і захапляючым. Спадзяёмся, што атрыманыя веды вы зможаце выкарыстаць для рашэння практычных задач з розных прадметных галін.

Матэрыял вучэбнага дапаможніка падзелены на дзве калонкі. Колер фону дапаможа вам разабрацца ў прызначэнні размешчанай на гэтым фоне інфармацыі:

— асноўныя матэрыялы, абавязковыя для вывучэння;

— прыклады, якія ілюструюць асноўныя матэрыялы;

— азначэнні асноўных паняццяў;

— гістарычныя звесткі, інфармацыя пра вучоных, якія ўнеслі ўклад у развіццё інфарматыкі, і іншыя цікавыя факты.

У вучэбным дапаможніку выкарыстоўваюцца наступныя ўмоўныя абазначэнні:


— пытанні і заданні для праверкі ведаў;


— раздзел «Практыкаванні» змяшчае заданні, пры выкананні якіх выкарыстоўваецца камп'ютар;


— раздзел «Практыкаванні» змяшчае заданні для выканання ў сшытку;


— раздзел «Практыкаванні» змяшчае заданні, пры выкананні якіх можа быць выкарыстана інфармацыя, размешчаная на Нацыянальным адукацыйным партале;

* — заданне або прыклад для цікаўных.

У тэксце некаторых заданняў вам будзе прапанавана адкрыць файл. Гэта азначае, што заданне можна выканаць, выкарыстоўваючы файл, размешчаны на Нацыянальным адукацыйным партале («Электронное навучанне» → «Электронныя адукацыйныя рэсурсы» → «Інфарматыка» → «Інфарматыка. 7 клас»). Зайсці на партал і спампаваць файлы да практыкаванняў можна па спасылцы http://e-vedy.adu.by або з дапамогай матрычнага QR-кода:

Імя файла для спампоўвання змяшчае нумар параграфа і нумар задання з практыкавання пасля гэтага параграфа. Напрыклад, імя файла upr4_3 азначае, што файл належыць да трэцяга задання з практыкавання пасля чацвёртага параграфа. Таксама на партале размешчаны файлы з праграмамі, якія разглядаюцца ў прыкладах. Такія файлы маюць імя pr8_3.pas (праграма для прыкладу 8.3).

Да дадзенага вучэбнага дапаможніка ёсць электронны дадатак (рэжым доступу: http://informatika7.adu.by).

Глава 1
ІНФАРМАЦЫЯ І ІНФАРМАЦЫЙНЫЯ ПРАЦЭСЫ

§ 1. Інфармацыя ў жыцці чалавека

Паняцце «інфармацыя» мае мноства азначэнняў.

Савецкі вучоны ў галіне агульнай механікі і прыкладной матэматыкі акадэмік М. М. Майсееў (1917—2000) лічыў, што «з прычыны шырыні гэтага паняцця няма і не можа быць строгага і досыць універсальнага азначэння інфармацыі».

Норберт Вінэр (1894—1964), амерыканскі матэматык і філосаф, заснавальнік кібернетыкі і тэорыі штучнага інтэлекту, гаварыў: «Інфармацыя — гэта не матэрыя і не энергія, інфармацыя — гэта інфармацыя».

1.1. Віды інфармацыі

Кожны з нас не раз чуў слова «інфармацыя». Інфармацыю мы атрымліваем з кніг і газет, з Інтэрнэту, ад людзей, з якімі маем зносіны. А што ж азначае дадзенае паняцце?

Інфармацыя — звесткі пра прадметы, падзеі, з'явы і працэсы навакольнага свету.

Вялікая частка звестак, на аснове якіх фарміруецца ўяўленне чалавека пра свет, паступае да яго дзякуючы органам пачуццяў. Наяўнасць зроку, слыху, дотыку, смаку і нюху дазваляе нам атрымліваць веды пра навакольную рэчаіснасць. У залежнасці ад таго, з дапамогай якіх органаў пачуццяў інфармацыя паступіла да чалавека, яе класіфікуюць паводле спосабу ўспрымання (прыклад 1.1).

- **Візуальная інфармацыя** ўспрымаецца органамі зроку (вачамі), якія адрозніваюць форму,

аб'ём, колер, перамяшчэнне і змяненне аб'ектаў.

• Органы слыху (вушы) успрымаюць **гукавую інфармацыю.** З іх дапамогай можна распазнаваць гукі, адрозніваць іх тэмбр, вышыню, рытм.

• **Тактыльная інфармацыя** звязана з органамі дотыку, якія дазваляюць навобмацак вызначыць характар паверхні, яе тэмпературу, адчуць дакрананне.

• З дапамогай органаў смаку чалавек атрымлівае **смакавую інфармацыю** пра ежу: горкая, салодкая, кіслая, салёная.

• Орган нюху (нос) успрымае і распазнае **інфармацыю пра пахі.**

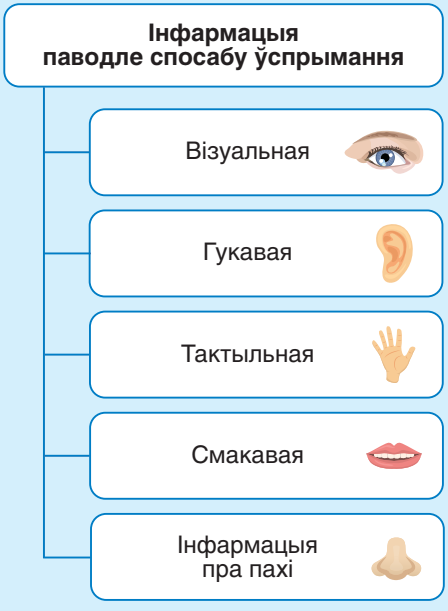
Існуе таксама класіфікацыя інфармацыі паводле формы ўяўлення (прыклад 1.2).

• **Графічная інфармацыя** — звесткі ў выглядзе рысункаў, схем, фатаграфій.

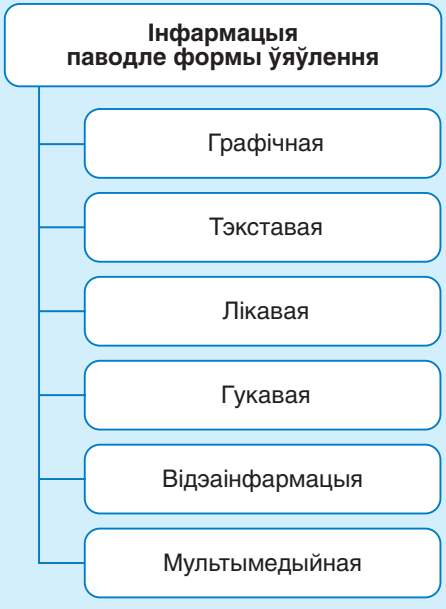
• **Тэкставая інфармацыя** — звесткі ў выглядзе спецыяльных сімвалаў (літар розных алфавітаў; іерогліфаў, з дапамогай якіх запісваюць асобныя склады або словы).

• **Лікавая (лічбавая) інфармацыя** — звесткі, якія адлюстроўваюць колькасную меру

Прыклад 1.1.



Прыклад 1.2.



Акрамя класіфікацыі інфармацыі паводле спосабу ўспрымання і формы ўяўлення, існуюць таксама і іншыя класіфікацыі. Да прыкладу, паводле сферы ўзнікнення інфармацыю можна падзяліць на наступныя групы:

- механічная;
- біялагічная;
- сацыяльная.

Механічная інфармацыя адлюстроўвае працэсы і з'явы неадушаўлёнай прыроды.

Сфера ўзнікнення біялагічнай інфармацыі звязана з працэсамі жывёльнага і раслінага свету.

Сацыяльная інфармацыя адлюстроўвае працэсы чалавечага грамадства.

Прыклад 1.3. Прыклады старажытных носьбітаў інфармацыі.

Васковыя таблічкі

Пергамент

Гліняныя таблічкі

аб'ектаў і іх уласцівасцей з дапамогай лікаў і лічбаў.

- **Гукавая інфармацыя** — звесткі ў выглядзе гукаў.

Існуюць і камбінаваныя віды інфармацыі — **відэаінфармацыя** і **мультымедыйная інфармацыя.**

Адзін і той жа від інфармацыі можа паступаць і захоўвацца ў розных форме. Напрыклад, музычны твор можа захоўвацца ў выглядзе аўдыя- або нотнага запісу.

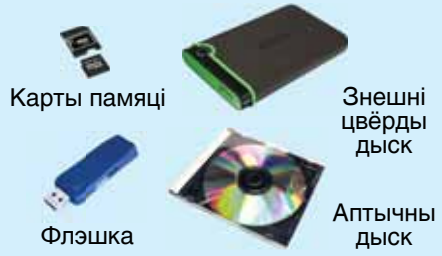
Для пераўтварэння інфармацыі з аднаго фармату ў іншы выкарыстоўваюцца розныя алгарытмы і ўстройствы.

1.2. Носьбіты інфармацыі

Для запісу, захоўвання і счытвання інфармацыі выкарыстоўваюцца носьбіты інфармацыі. У старажытнасці чалавек захоўваў важныя звесткі толькі ва ўласнай памяці, г. зн. памяць чалавека з'яўляецца натуральным носьбітам інфармацыі. Патрэба запамінаць і захоўваць аб'ёмы інфармацыі, якія ўвесь час узрастаюць, прывяла да выкарыстання і стварэння розных матэрыялаў і ўстройстваў.

Да носьбітаў інфармацыі належаць папера, кніга, фатаграфія, аптычныя дыскі, флэш-памяць і інш. (прыклады 1.3 і 1.4). Для апрацоўкі, захоўвання і распаўсюджвання інфармацыі будуюць спецыяльныя будынкі — датацэнтры.

Прыклад 1.4. Камп'ютарныя носьбіты інфармацыі.

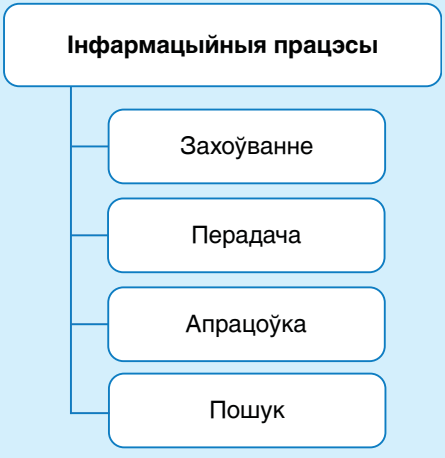


1.3. Інфармацыйныя працэсы

У паўсядзённым жыцці мы запісваем, запамінаем і счытваем атрыманую інфармацыю. Чалавек можа падзяліцца вядомай яму інфармацыяй з іншымі людзьмі. Акрамя таго, на аснове апрацоўкі ўжо наяўнай інфармацыі можна ствараць новую.

Любая дзейнасць чалавека, звязаная з інфармацыяй, з'яўляецца **інфармацыйным працэсам.** Адрозніваюць наступныя інфармацыйныя працэсы: **захоўванне, перадача, апрацоўка, пошук** інфармацыі (прыклад 1.5).

Прыклад 1.5.

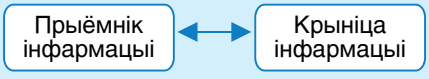


Прыклад 1.6.

Людзі захоўваюць інфармацыю або ва ўласнай памяці, або на якіх-небудзь знешніх носьбітах (прыклад 1.6). Часам чалавек забывае інфармацыю, а інфармацыя на знешніх носьбітах

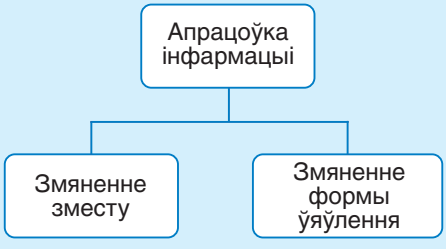
Прыклад 1.7.

Перадача інфармацыі

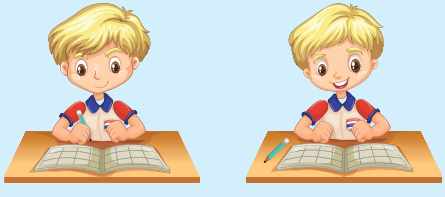


Канал сувязі

Прыклад 1.8.



Прыклад 1.9.



$2x + 3 = 7$

$2x = 7 - 3$

$2x = 4$

$x = \frac{4}{2}$

$x = 2$

можа захоўвацца доўга і быць даступная розным людзям.

Перадача інфармацыі адбываецца, да прыкладу, пры размове двух людзей, пры чытанні кнігі або часопіса, пры прагляданні старонак у сетцы Інтэрнэт і інш.

У працэсе перадачы заўсёды ўдзельнічаюць два бакі: **крыніца** і **прыёмнік.** Перадача інфармацыі адбываецца праз **канал сувязі:** гукавыя хвалі пры непасрэднай размове, паслугі паштовага сэрвісу пры перапісцы, сотавая сувязь пры размове па мабільным тэлефоне (прыклад 1.7).

У выніку змянення зместу ці формы ўяўлення інфармацыі адбываецца яе апрацоўка (прыклад 1.8).

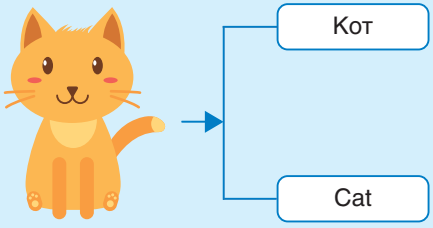
Змест інфармацыі можа змяніцца ў выніку вылічальных дзеянняў, дапусцім, пры рашэнні любой матэматычнай задачы, ураўнення (прыклад 1.9). Працэс разважанняў чалавека таксама можа прыводзіць да з'яўлення новай інфармацыі. Новая інфармацыя ўзнікае пры даследаванні з'яў прыроды, фізічных працэсаў і інш.

Форма ўяўлення інфармацыі змяняецца пры маляванні карцін па тэкставым апісанні або

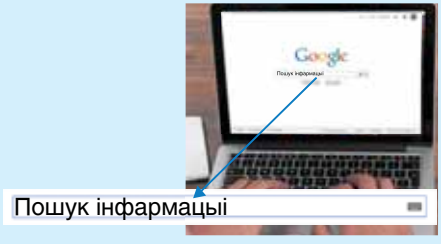
пры апісанні сюжэта відэафільма ў выглядзе артыкула ў часопісе, пры перакладзе тэксту з адной мовы на іншую (прыклад 1.10) і г. д.

Калі чалавеку неабходна знайсці звесткі, якія яго цікавяць, то ён ажыццяўляе пошук інфармацыі. Для пошуку патрэбнай інфармацыі выкарыстоўваюцца разнастайныя спосабы і метады: чытанне энцыклапедый, слоўнікаў, кніг і часопісаў, прагляд відэафільмаў і тэлевізійных перадач, пошук у сетцы Інтэрнэт (прыклад 1.11) і г. д.

Прыклад 1.10.



Прыклад 1.11.



1. Што разумеюць пад інфармацыяй?
2. Як можна класіфікаваць інфармацыю паводле спосабу яе ўспрымання чалавекам?
3. На якія віды падзяляюць інфармацыю паводле формы ўяўлення?
4. Што такое носьбіт інфармацыі?
5. Якія інфармацыйныя працэсы выконвае чалавек?

Практыкаванні

1 Якія інфармацыйныя працэсы выконвае сямікласнік пры наступных відах дзейнасці?

1. Запіс у канспект матэрыялаў урока.
2. Вусны адказ ля дошкі.
3. Пераклад тэксту з рускай мовы на англійскую.
4. Падбор матэрыялаў для рэферата па гісторыі.

2 Прывядзіце прыклады прафесій, у якіх асноўная дзейнасць спецыяліста звязана з інфармацыяй.

3 Апішыце сітуацыі, у якіх вы можаце адыгрываць ролю крыніцы інфармацыі; прыёмніка інфармацыі. Якім каналам сувязі вы карыстаецеся пры гэтым?

4 Падрыхтуйце паведамленне на адну з наступных тэм:
1. «Старажытныя носьбіты інфармацыі».
2. «Электронныя носьбіты інфармацыі».
3*. «Дапоўненая рэальнасць як форма ўяўлення інфармацыі».

§ 2. Уяўленне інфармацыі ў камп'ютары

2.1. Кадзіраванне інфармацыі

Для зносін людзі выкарыстоўваюць натуральную мову, напрыклад беларускую або рускую. У аснове натуральнай мовы ляжыць алфавіт — сістэма графічных знакаў для перадачы гукаў вуснага маўлення. Алфавіт натуральнай мовы з'яўляецца ўніверсальным кодам любой пісьмовай культуры.

Акрамя натуральных, чалавек выкарыстоўвае штучна створаныя мовы са сваімі спецыяльнымі кодамі: мову матэматычных ці хімічных формул, ноты і інш.

Код — сукупнасць умоўных знакаў, кожнаму з якіх прысвойваецца пэўнае значэнне (прыклады 2.1 і 2.2).

Працэс запісу ці пераўтварэння інфармацыі ў адпаведнасці з правіламі, зададзенымі некаторым кодам, называюць **кадзіраваннем.** Працэс, адваротны кадзіраванню, называюць **дэкадзіраваннем.**

Кадзіраванне інфармацыі выкарыстоўвалася са старажытнасці. Шырока вядомы шыфр Юлія Цэзара, які ўжываўся для запісу сакрэтных паведамленняў. Кожны сімвал у тэксце замяняўся сімвалам, што знаходзіўся на некаторай пастаяннай адлегласці лявей ці правей за яго ў алфавіце.

Напрыклад, пры кадзіраванні інфармацыі з дапамогай літар рускага алфавіта шляхам зруху ўправа на 3 літара «А» была б заменена на «Г», літара «Б» стала б «Д» і г. д.

Прыклад 2.1. Сёння шырока ўжываюцца штрых-коды на розных таварах. Перад вамі штрых-код згушчанага малака:

Кадзіраваць і перадаваць інфармацыю можна рознымі спосабамі: вусна, пісьмова, жэстамі і інш. Камп'ютар можа апрацоўваць лікавую, тэкставую, графічную і гукавую інфармацыю толькі ў лічбавым фармаце, які ў камп'ютары прадстаўлены ў выглядзе двайковага кода.

Двайковы код — спосаб кадзіравання, у якім кожны **разрад** прымае адно з двух магчымых значэнняў, што звычайна абазначаюцца лічбамі 0 і 1. **Разрад** у гэтым выпадку называецца **двайковым разрадам.**

Такі спосаб кадзіравання звязаны з тым, што прасцей за ўсё рэалізуюцца тэхнічныя ўстройствы, якія валодаюць двума ўстойлівымі станамі: уключана/выключана, злучана/раз'яднана і інш.

Для кадзіравання лікавай інфармацыі ў камп'ютары замест дзесятковай сістэмы лічэння выкарыстоўваецца двайковая, заснаваная на двайковым кодзе.

Кадзіраванне тэкставай інфармацыі ў камп'ютары выконваецца пры дапамозе спецыяльных кодавых табліц, дзе кожнаму знаку ставіцца ў адпаведнасць пэўная паслядоўнасць з нулёў і адзінак (прыклад 2.3).

Прыклад 2.2. Са з'яўленнем смартфонаў пачалі распаўсюджвацца QR-коды. Яны дазваляюць хутка заносіць у тэлефон тэкставую інфармацыю, дадаваць кантакты ў адрасную кнігу, пераходзіць па web-спасылках, адпраўляць SMS-паведамленні і г. д. Вось, напрыклад, QR-код са спасылкай на артыкул у Wikipedia пра QR-коды:

Прыклад 2.3. Кадзіраванне некаторых літар англійскага алфавіта на камп'ютары.

Літара	Двайковы код
A	01000001
B	01000010
C	01000011
D	01000100
E	01000101
F	01000110
G	01000111
H	01001000
I	01001001
J	01001010
K	01001011
L	01001100
M	01001101
N	01001110

Прыклад 2.4.

Прыклад 2.5. Суадносіны паміж бітам і байтам.

2.2. Адзінкі вымярэння аб'ёму інфармацыі

Чалавек ужывае розныя адзінкі вымярэння. Так, для вымярэння часу выкарыстоўваюцца секунды, мінуты, гадзіны, для вымярэння адлегласці — метры, кіламетры і інш. Вымярэнні праводзяць з дапамогай вымяральных прыбораў (прыклад 2.4).

Для вызначэння колькасці інфармацыі ёсць свае адзінкі вымярэння. Мінімальную колькасць інфармацыі, для кадзіравання якой дастаткова аднаго двайковага разраду, называюць **бітам** (bit).

Слова «біт» утварылася ад англійскіх слоў *binary* (двайковы) і *digit* (знак). Біт — мінімальная адзінка, якая паказвае колькасць інфармацыі. Ён можа прымаць адно з двух значэнняў — 0 або 1. Для зручнасці ўведзена больш буйная адзінка вымярэння інфармацыі — байт.

Байт — адзінка вымярэння колькасці інфармацыі, якая складаецца з васьмі паслядоўных і ўзаемазлучаных бітаў.

1 байт = 2^3 біт = 8 біт (прыклад 2.5).

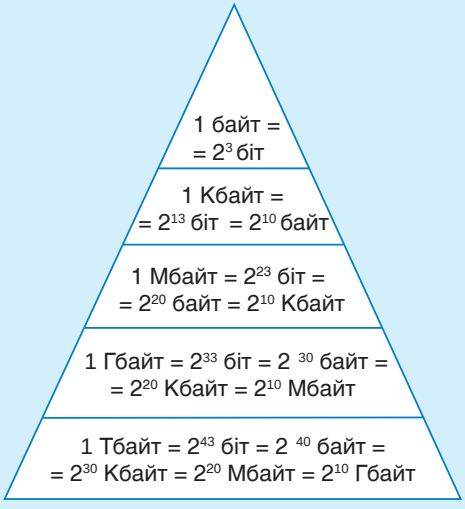
Для абазначэння большага аб'ёму інфармацыі выкарыстоўваюцца іншыя адзінкі вымярэння:

1 Кбайт (кілабайт) =
= 1024 байты;
1 Мбайт (мегабайт) =
= 1 048 576 байт;
1 Гбайт (гігабайт) =
= 1 073 741 824 байты;
1 Тбайт (тэрабайт) =
= 1 099 511 627 776 байт.

Значэнні дадзеных адзінак вымярэння інфармацыі для зручнасці кадзіравання звязаны са ступенню ліку 2 (прыклад 2.6).

У гэтых адзінках вымяраюцца колькасць (аб'ём) аператыўнай ці знешняй памяці камп'ютара, памеры файлаў. У прыкладзе 2.7 паказана, якім чынам выконваецца пераход адных адзінак вымярэння інфармацыі ў іншыя.

Прыклад 2.6. Суадносіны адзінак вымярэння інфармацыі.



Прыклад 2.7.
Перавядзём 2364 Мбайт у кілабайты і гігабайты:

2368 Мбайт =
$= (2368 \cdot 2^{10})$ Кбайт =
= 2 424 832 Кбайт;
2368 Мбайт =
$= (2368 / 2^{10})$ Гбайт $\approx$
$\approx$ 2,3 Гбайт.

1. Што такое код?
2. Які працэс называюць кадзіраваннем інфармацыі?
3. Які код выкарыстоўваюць для кадзіравання інфармацыі ў камп'ютары?
4. Якія адзінкі вымярэння інфармацыі вы ведаеце?

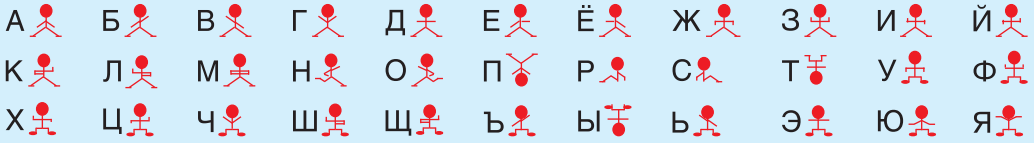
Практыкаванні

1 Выкарыстоўваючы шыфр Юлія Цэзара са зрухам управа на 3, закадзіруйце фразу «Хто валодае інфармацыяй, той валодае светам».

2 Выкарыстоўваючы шыфр Юлія Цэзара са зрухам улева на 2, закадзіруйце фразу Сціва Джобса «Камп'ютар — гэта як веласіпед для нашага мозгу».

3 У адным з апавяданняў А. Конан Дойля вялікі сышчык Шэрлак Холмс разгадвае шыфр чалавечкаў, якія скачуць. Расшыфруйце фразу, выкарыстоўваючы алфавіт, што ўжываўся пры яе кадзіраванні.

Алфавит для кодирования информации



4 У азбуцы Морзэ літары і лічбы замяняюцца паслядоўнасцямі з кароткіх і доўгіх сігналаў — кропак і працяжнікаў:

А	Б	В	Г	Д	Е	Ж	З	И	К	Л	М	Н	О
.-	-...	.--	--.	-..	.	...-	--..	..	-.-	.-..	--	-.	---
П	**Р**	**С**	**Т**	**У**	**Ф**	**Х**	**Ц**	**Ч**	**Ш**	**Щ**	**Ъ, Ь**	**Ы**	**Э**
.--.	.-.	...	-	..-	..-.	....	-.-.	---.	----	--.-	-..-	-.--	..-..
Ю	**Я**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**0**		
..--	.-.-	.----	..---	...--	....-	.....	-....	--...	---..	----.	-----		

1. З дапамогай азбукі Морзэ запішыце: «Запас беды не чинит».
2. Расшыфруйце інфармацыю, запісаную на азбуцы Морзэ.

-.	.-		---	----	..	-...	-.-	.-	....		..-	---.	.-	-	...	.-.-

5 Выканайце перавод адзінак вымярэння інфармацыі:

1. 174 байта ў біты.
2. 342,3 Кбайт у байты.
3. 45 638 Мбайт у гігабайты.

Глава 2
УЯЎЛЕННЕ ПРА ЛОГІКУ ВЫКАЗВАННЯЎ. МНОСТВЫ І АПЕРАЦЫІ НАД ІМІ

§ 3. Логіка выказванняў

Магчымасці камп'ютара вялікія. Ён можа дапамагчы ўрачу паставіць правільны дыягназ пацыенту, пасажыру — выбраць білет на патрэбны цягнік; камп'ютар можа кіраваць аўтамабілем, складаць прагнозы надвор'я і шмат што іншае.

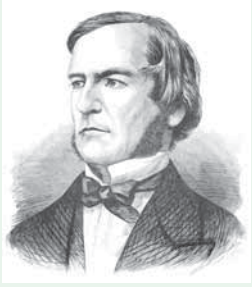
Для таго каб высветліць, ці можа камп'ютар «думаць», спачатку трэба зразумець, як думае чалавек. Бо менавіта чалавек стварыў камп'ютар, і камп'ютар выконвае толькі тыя дзеянні, якім яго навучыў чалавек.

Нашы веды пра навакольны свет мы перадаём у апавядальных сказах. Такія сказы могуць адлюстроўваць рэчаіснасць правільна або няправільна. Думаючы, чалавек будуе свае разважанні, засноўваючыся на ўласных ведах.

Яшчэ Арыстоцель заўважыў, што правільнасць разважанняў не залежыць ад зместу, а вызначаецца формай.

Старажытнагрэчаскі філосаф Арыстоцель (384—322 гг. да н. э.) першым сістэматызаваў формы і правілы мыслення, распрацаваў тэорыю высноў і доказаў, апісаў лагічныя аперацыі. Арыстоцелю належаць фармулёўкі асноўных законаў мыслення.

Ля вытокаў сучаснай логікі стаіць нямецкі матэматык Готфрыд Вільгельм Лейбніц (1646—1716). Вучоны прапанаваў ідэю паказаць лагічныя разважанні як вылічэнні, падобныя да вылічэнняў у матэматыцы.



Англійскі матэматык і логік Джордж Буль (1815—1864) перанёс на логіку законы і правілы матэматычных (алгебраічных) дзеянняў, стварыўшы тым самым алгебру логікі.

На лагічных элементах будуюцца лагічныя схемы электронных устройстваў. Законы булевай алгебры выкарыстоўваюцца і ў праграмiраванні.

Навука, якая вывучае формы разважанняў, называецца **фармальнай логікай.**

Матэматычная логіка выкарыстоўвае матэматычныя метады для даследавання спосабаў пабудовы разважанняў, доказаў, высноў.

Адным з раздзелаў сучаснай матэматычнай логікі з'яўляецца **логіка выказванняў.**

На правілах матэматычнай логікі пабудаваны працэсы «разважанняў» камп'ютара. Вывучэнне логікі выказванняў дапаможа зразумець, як можна навучыць камп'ютар «думаць».

3.1. Паняцце выказвання

Выказванне — апавядальны сказ (сцверджанне), пра які ў цяперашні час можна сказаць, праўдзівы ён ці непраўдзівы (прыклад 3.1).

Прыклад 3.1. Наступныя сказы з'яўляюцца выказваннямі:

1. Атам вадароду самы лёгкі (праўдзіва).
2. Клетка — цэнтральная частка атама (непраўдзіва).
3. Кірыла Тураўскі — вядомы англійскі пісьменнік і аратар, які жыў у другой палове XII ст. (непраўдзіва).
4. Пры дзяленні любога ліку (акрамя нуля) на самога сябе атрымліваецца лік 1 (праўдзіва).

Пра праўдзівасць выказвання можна гаварыць толькі ў цяперашнім часе: выказванне «Ідзе дождж» можа быць праўдзівым зараз і непраўдзівым праз гадзіну.

Як правіла, выказванні абазначаюць вялікімі лацінскімі літарамі. Калі выказванне A праў-

дзівае, пішуць $A = 1$, калі непраўдзівае — $A = 0$ (прыклад 3.2). Часта выкарыстоўваюць такія абазначэнні: A = true (праўдзіва) і A = false (непраўдзіва).

3.2. Лагічная аперацыя НЕ

З выказваннямі можна выконваць розныя аперацыі, як у матэматыцы — з лікамі (складанне, множанне, адніманне і інш.).

Лагічная аперацыя **НЕ** (адмаўленне) мяняе значэнне выказвання на супрацьлеглае: праўдзівае на непраўдзівае, а непраўдзівае на праўдзівае.

Лагічнае адмаўленне атрымліваецца з выказвання шляхам дадавання часціцы «не» да выказніка ці з выкарыстаннем звароту «няпраўда, што...» (прыклад 3.3). Часам пры пабудове адмаўленняў некаторыя словы замяняюць іх антонімамі, калі гэта магчыма.

Калі выказванне ўключае словы «ўсе», «усякі», «любы», то яго адмаўленне будуецца з выкарыстаннем слоў «некаторыя», «хоць бы адзін». І наадварот, для выказванняў са словамі «некаторыя», «хоць бы адзін» адмаўленне будзе ўключаць словы «ўсе», «усякі», «любы» (прыклад 3.4).

Прыклад 3.2.

A = «a^0 роўна 1, калі $a \neq 0$»;

B = «Масу вымяраюць у літрах».

Для прыведзенага прыкладу $A = 1$, $B = 0$.

Прыклад 3.3. Пабудуем адмаўленне выказванняў.

Выказванні:

1. У кветкавых раслін развіваецца плод.
2. Фрэска — гэта жывапіс вадзянымі фарбамі па свежай тынкоўцы.

Адмаўленне выказванняў:

1. У кветкавых раслін **не** развіваецца плод.
2. **Няпраўда, што** фрэска — гэта жывапіс вадзянымі фарбамі па свежай тынкоўцы.

Прыклад 3.4. Пабудуем адмаўленне выказванняў.

Выказванні:

1. Усе навучэнцы займаюцца спортам.
2. Некаторыя птушкі ўмеюць плаваць.
3. Любая кветка мае пах.
4. Часам у мамы бывае дрэнны настрой.

Адмаўленне выказванняў:

1. **Некаторыя** навучэнцы **не** займаюцца спортам.
2. **Усе** птушкі **не** ўмеюць плаваць.
3. **Хоць бы адна** кветка **не** мае паху.
4. У мамы **заўсёды** бывае **добры** настрой.

Прыклад 3.5. Вызначэнне праўдзівасці выказванняў з адмаўленнямі.

1. Елка — гэта дрэва (праўдзівае выказванне). Елка — гэта не дрэва (непраўдзівае выказванне).

$A = 1$, **НЕ** $A = 0$.

2. Лік -7 з'яўляецца дадатным (непраўдзівае выказванне). Лік -7 не з'яўляецца дадатным (праўдзівае выказванне).

$A = 0$, **НЕ** $A = 1$.

3. Усе рэчывы — металы (непраўдзівае выказванне). Некаторыя рэчывы не металы (праўдзівае выказванне).

$A = 0$, **НЕ** $A = 1$.

4. Усе складнікі паветра з'яўляюцца газамі (праўдзівае выказванне). Некаторыя складнікі паветра не з'яўляюцца газамі (непраўдзівае выказванне).

$A = 1$, **НЕ** $A = 0$.

5. Працягласць сутак не залежыць ад хуткасці вярчэння планеты (непраўдзівае выказванне). Працягласць сутак залежыць ад хуткасці вярчэння планеты (праўдзівае выказванне).

$A = 0$, **НЕ** $A = 1$.

6. Дамы на левым баку вуліцы маюць цотныя нумары (непраўдзівае выказванне). Няпраўда, што дамы на левым баку вуліцы маюць цотныя нумары (праўдзівае выказванне).

$A = 0$, **НЕ** $A = 1$.

Любую аперацыю над лікамі ў матэматыцы абазначаюць якімнебудзь знакам: «+», «−», «·», «:». Для лагічных аперацый таксама вызначаны свае абазначэнні. Калі аперацыю адмаўлення ўжываюць у дачыненні да выказвання A, то гэта можна запісаць так: **НЕ** A. Можна сустрэць і іншыя абазначэнні для лагічнай аперацыі адмаўлення: **Not** A, $\neg A$, $\overline{A}$, $\sim A$.

Калі нас цікавіць праўдзівасць выказвання **НЕ** A, то яе (незалежна ад зместу) можна вызначыць па табліцы праўдзівасці:

A	**НЕ** A
1	0
0	1

З табліцы праўдзівасці вынікае, што адмаўленнем праўдзівага выказвання будзе непраўдзівае, а адмаўленнем непраўдзівага — праўдзівае (прыклад 3.5). Выказванне і яго адмаўленне ніколі не могуць быць праўдзівымі або непраўдзівымі адначасова.

Напрыклад, адмаўленнем выказвання «У мяне ёсць камп'ютар» будзе выказванне «У мяне няма камп'ютара» (ці выказванне «Няпраўда, што ў мяне ёсць

камп'ютар»). Праўдзівасць гэтых выказванняў залежыць ад канкрэтнага чалавека. Для адных будзе праўдзівым першае выказванне, а для іншых — другое. Але абодва выказванні не могуць быць праўдзівымі ці непраўдзівымі адначасова для аднаго і таго ж чалавека.

Часта цяжка вызначыць праўдзівасць выказвання. Выказванне «Плошча возера Нарач 79,6 км^2» у адной сітуацыі можна палічыць непраўдзівым, а ў іншай — праўдзівым. Непраўдзівым — таму што названае значэнне не з'яўляецца пастаянным. Праўдзівым — калі разглядаць яго як некаторае прыбліжэнне, што прынята на практыцы.

? 1. Што такое выказванне?
2. Якія значэнні могуць мець выказванні?
3. Што робіць лагічная аперацыя *адмаўленне*?
4. Як пабудаваць адмаўленне выказвання?

Практыкаванні

1 Якія са сказаў з'яўляюцца выказваннямі, а якія — не?

1. Уключы манітор.
2. Кісларод — гэта газ.
3. Шышка — гэта кветка.
4. Колькі вады ўцякло?
5. Усе дзеці — навучэнцы.
6. Хоць бы адзін пароль будзе правільным.

2 Вызначыце праўдзівасць выказванняў.

1. 123 — гэта лічба.
2. Слова «стол» — гэта назоўнік.
3. Лік 46 з'яўляецца ступенню ліку 2.
4. Значэнне выразу $a = \frac{x + y}{3}$ роўна 0,75.
5. Жалеза лягчэйшае за ваду.

3 Пабудуйце адмаўленні выказванняў.

1. Міша не можа пайсці ў кіно.
2. Соня любіць маляваць.
3. Усе планеты не маюць атмасферы.

4. У верасні не бывае дажджоў.
5. Сонца свеціць ярка.
6. Некаторыя птушкі адлятаюць на поўдзень.

4 Адкрыйце файл з дадзенымі ніжэй сказамі і адрэдагуйце іх, выдаліўшы ці ўставіўшы часціцу «не» так, каб усе сказы сталі праўдзівымі выказваннямі.

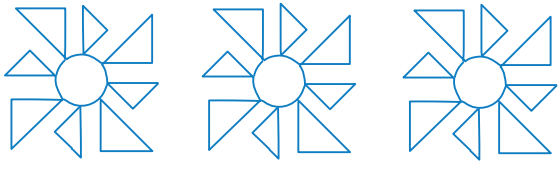
Возера Нарач не з'яўляецца найбуйнейшым возерам Беларусі.
Усе горы з'яўляюцца вулканамі.
Дуб — хвойнае дрэва.
Лік 27 з'яўляецца простым лікам.
Тэрмометр не дазваляе вызначыць тэмпературу цела.
Лік 2016 не дзеліцца на 3.
Трохвугольнік не з'яўляецца геаметрычнай фігурай.

5 Якія сцверджанні пра жывёл, паказаных на рысунках, праўдзівыя, а якія — непраўдзівыя?



1. Некаторыя з гэтых жывёл умеюць лазіць па дрэвах.
2. Усе жывёлы жывуць у лясах.
3. Ніводнае з жывёл не з'яўляецца хатнім.
4. Кожную жывёлу можна пагладзіць.
5. Усе людзі любяць мышэй.
6. Ніводнае з жывёл не ўмее плаваць.

6 Адкрыйце файл з рысункам трох кветак. Размалюйце іх так, каб кожнае з выказванняў было праўдзівым.



1. Усе кветкі маюць жоўты круг у сярэдзіне.
2. На рысунку ёсць кветка з сінімі пялёсткамі.
3. На рысунку няма кветкі з чырвонымі пялёсткамі.
4. Няпраўда, што колер круга ў сярэдзіне кветкі супадае з колерам пялёсткаў.
5. Хоць бы ў адной кветкі пялёсткі рознага колеру.

7 Стварыце 4 копіі рысункаў, атрыманых у заданні 6. Дапоўніце кожную копію відарысамі ваз (выберыце з файла) так, каб адпаведнае з ніжэйпрыведзеных выказванняў было непраўдзівым.

1. Усе відарысы ваз — чатырохвугольнікі.
2. На вазах ёсць арнамент у выглядзе кругоў.
3. Усе кругі ў арнаменце рознага памеру.
4. Хоць бы адзін круг у арнаменце белага колеру.

8 Задуманы некаторы лік x. Сярод выказванняў $x > 1$, $x > 2$, $x > 3$, $x > 4$, $x > 5$ ёсць два правільныя і тры няправільныя. Якія выказванні няправільныя?

9* Рашыце задачу-верш.

Собаки с рыжими хвостами
Себе овсянку варят сами.
Тем, чьи хвосты стального цвета,
Не позволяют делать это.
Кто варит сам себе овсянку,
Гулять выходит спозаранку.
Все, кто гулять выходят рано,
Не терпят фальши и обмана.
Вид добродушный у Барбоса,
Но на сорок он смотрит косо.
Он видит: норовят сороки
У воробьев списать уроки!
Скажите — проще нет вопроса! —
Какого цвета хвост Барбоса?[1]

---

[1] Разговоров, Н. «Собаки с рыжими хвостами...» [Электронны рэсурс] / Н. Разговоров. — Рэжым доступу: http://po.m-necropol.ru/razgovorov-nikita.html. — Дата доступу: 26.06.2017.

§ 4. Лагічныя аперацыі І і АБО

У 1936—1938 гг. амерыканскі інжынер і матэматык Клод Шэнан (1916 — 2001) знайшоў ужыванне булевай логіцы пры канструяванні схем з рэле і пераключальнікаў. У далейшым гэта адкрыццё паслужыла асновай для пабудовы лагічных элементаў, на якіх працуе камп'ютарная тэхніка. Стан элементаў камп'ютара адпавядае лагічным значэнням:

- калі сігнал прысутнічае, атрымліваем лагічную 1;
- калі сігнал адсутнічае, атрымліваем лагічны 0.

Прыклад 4.1. Прааналізуем выказванне «Лік 456 трохзначны і цотны».

Дадзенае выказванне з'яўляецца састаўным, паколькі змяшчае два простыя выказванні:

«Лік 456 трохзначны» (выказванне A);

«Лік 456 цотны» (выказванне B).

Выказванні A і B злучаны разам лагічнай аперацыяй **І**, у выніку атрымана састаўное выказванне A **І** B. Выказванне A праўдзівае, выказванне B праўдзівае. Таму выказванне A **І** B праўдзівае: (A **І** B) = 1.

Логіка выказванняў дазваляе будаваць **састаўныя выказванні.** Яны ствараюцца з некалькіх простых выказванняў шляхам злучэння іх адно з адным з дапамогай лагічных аперацый **НЕ, І, АБО** і інш.

4.1. Лагічная аперацыя І

Вызначэнне праўдзівасці або непраўдзівасці састаўнога выказвання залежыць ад таго, ці з'яўляюцца праўдзівымі або непраўдзівымі простыя выказванні, якія ўваходзяць у яго склад, а таксама ад той лагічнай аперацыі, што іх звязвае.

Састаўное выказванне A **І** B, утворанае ў выніку аб'яднання двух простых выказванняў A і B лагічнай аперацыяй **І**, праўдзіва тады і толькі тады, калі A і B адначасова праўдзівыя.

Калі хоць бы адно з простых выказванняў, звязаных аперацыяй **І**, будзе непраўдзівым, то і састаўное выказванне будзе непраўдзівым (прыклады 4.1 і 4.2).

Аперацыю **І** называюць **лагічным множаннем.** Роўнасці

$1 \cdot 1 = 1$, $1 \cdot 0 = 0$, $0 \cdot 1 = 0$, $0 \cdot 0 = 0$, правільныя для звычайнага множання, правільныя і для лагічнага множання.

Прывядзём табліцу праўдзівасці для лагічнай аперацыі **І**:

A	B	A **І** B
1	1	1
0	1	0
1	0	0
0	0	0

Для запісу лагічнай аперацыі **І** выкарыстоўваюць наступныя абазначэнні: A **І** B, A **AND** B, $A \cdot B$, $A * B$, $A \wedge B$, $A \,\&\, B$.

4.2. Лагічная аперацыя АБО

Састаўное выказванне A **АБО** B, утворанае ў выніку аб'яднання двух простых выказванняў A і B лагічнай аперацыяй **АБО**, непраўдзівае тады і толькі тады, калі A і B адначасова непраўдзівыя.

Іншымі словамі, састаўное выказванне A **АБО** B будзе праўдзівым толькі ў тым выпадку, калі праўдзіва хоць бы адно з двух простых выказванняў, што складаюць гэта выказванне

Прыклад 4.2. Выказванне A: «Геракл — герой старажытнарускай міфалогіі». **Непраўдзіва**, $A = 0$.

Выказванне B: «Геракл — сын бога Зеўса». **Праўдзіва**, $B = 1$.

Выказванне A **І** B: «Геракл — герой старажытнарускай міфалогіі **І** сын бога Зеўса». **Непраўдзіва**, (A **І** B) = 0.

Прыклад 4.3. Прааналізуем выказванне «Сямікласнікі вывучаюць філасофію або астраномію».

Дадзенае састаўное выказванне ўтворана з двух простых:

«Сямікласнікі вывучаюць філасофію» (выказванне A);

«Сямікласнікі вывучаюць астраномію» (выказванне B).

Выказванні звязаны лагічнай аперацыяй **АБО**. У выніку атрымалася састаўное выказванне A **АБО** B. Выказванне A непраўдзівае, выказванне B непраўдзівае. Таму выказванне A **АБО** B непраўдзівае: (A **АБО** B) = 0.

Прыклад 4.4. Выказванне A: «Францыск Скарына — беларускі першадрукар». **Праўдзіва,** $A = 1$.

Выказванне B: «Стэфан Баторый — турэцкі султан». **Непраўдзіва,** $B = 0$.

Францыск Скарына

Стэфан Баторый

Выказванне «Францыск Скарына — беларускі першадрукар, **АБО** Стэфан Баторый — турэцкі султан» будзе **праўдзівым,** (A **АБО** B) = 1.

Прыклад 4.5*. Разгледзім выраз: A **АБО** B **І НЕ** C. Распішам па дзеяннях вылічэнне яго значэння:

1) D = **НЕ** C;
2) E = B **І** D;
3) F = A **АБО** E.

Значэнне выказвання F, атрыманае ў 3-м дзеянні, вызначыць значэнне зыходнага лагічнага выразу.

Прыклад 4.6*. Няхай выказванне $A = 1$, $B = 0$, $C = 0$. Знойдзем значэнне лагічнага выразу A **АБО** B **І НЕ** C.

1) D = **НЕ** C = 1;
2) E = B **І** D = 0 **І** 1 = 0;
3) F = A **АБО** E = 1 **АБО** 0 = 1.

Значыць, пры пачатковых значэннях $A = 1$, $B = 0$, $C = 0$ значэнне лагічнага выразу A **АБО** B **І НЕ** C праўдзівае.

(гл. прыклад 4.3 на с. 27 і прыклад 4.4).

Табліца праўдзівасці для лагічнай аперацыі **АБО:**

A	B	A **АБО** B
1	1	1
0	1	1
1	0	1
0	0	0

Аперацыю **АБО** называюць **лагічным складаннем.** Роўнасці 1 + 0 = 1, 0 + 1 = 1, 0 + 0 = 0, правільныя для звычайнага складання, правільныя і для лагічнага складання.

Для запісу лагічнай аперацыі **АБО** можна выкарыстоўваць наступныя выразы: A **АБО** B, A **OR** B, $A + B$, $A \vee B$, $A \mid B$.

Калі ў лагічным выразе прысутнічае некалькі лагічных аперацый, то важна вызначыць парадак іх выканання. Найвышэйшым прыярытэтам валодае аперацыя **НЕ.** Лагічная аперацыя **І,** г. зн. лагічнае множанне, выконваецца раней за аперацыю **АБО** — лагічнае складанне (прыклады 4.5* і 4.6*). Для змянення парадку выканання лагічных аперацый выкарыстоўваюць дужкі: у гэтым выпадку спачатку выконваюцца аперацыі ў дужках, а затым — усе астатнія.

Лагічныя аперацыі **І** і **АБО** падпарадкоўваюцца перамяшчальнаму закону:

A **І** $B = B$ **І** A;

A **АБО** $B = B$ **АБО** A.

Каб вызначыць значэнне састаўнога лагічнага выразу, часам дастаткова ведаць значэнне толькі аднаго простага выказвання. Так, калі ў састаўным выказванні з аперацыяй **І** хоць бы адно простае выказванне з'яўляецца непраўдзівым, то значэнне састаўнога выказвання будзе непраўдзівым. Калі ў састаўным выказванні з аперацыяй **АБО** хоць бы адно простае выказванне праўдзівае, то значэнне састаўнога выказвання будзе праўдзівым (прыклад 4.7).

Прыклад 4.7. Выказванне A: «Прагноз надвор'я абяцае дажджы». Выказванне B: «Зараз на вуліцы ідзе дождж».

Выказванне A **І** B будзе непраўдзівым, калі мы ўбачылі, што на вуліцы няма дажджу (незалежна ад таго, што абяцаў прагноз надвор'я).

Выказванне A **АБО** B будзе праўдзівым, калі прагноз надвор'я абяцаў дождж (незалежна ад таго, якое надвор'е мы назіраем зараз).

1. У якіх выпадках састаўное выказванне A **І** B можа быць праўдзівым?

2. У якіх выпадках састаўное выказванне A **АБО** B можа быць непраўдзівым?

Практыкаванні

1 Вызначыце, праўдзівымі ці непраўдзівымі з'яўляюцца ніжэйпрыведзеныя састаўныя выказванні.

1. Мяч круглы, **АБО** Зямля плоская.
2. Трусы — хатнія жывёлы, **І** баабаб расце ў Белавежскай пушчы.
3. Клавіятура — устройства ўводу інфармацыі, **АБО** мыш — устройства вываду інфармацыі.

4. І. А. Крылоў напісаў байку «Квартэт», **І** М. Ю. Лермантаў напісаў верш «Ветразь».
5. Сасна — хвойнае дрэва, **І** кедр — не хвойнае дрэва.
6. Манітор — устройства ўводу інфармацыі, **АБО** сканер — **НЕ** ўстройства вываду інфармацыі.
7. Кантыненты **І** астравы — гэта вялікія ўчасткі сушы.

2 Пра тое, як прайшлі летнія канікулы, Кіра расказала сваім сябрам наступнае:

1. Я была ў бабулі ў вёсцы, і побач з вёскай было возера.
2. Па возеры плавала лодка або качка.
3. Мы з бабуляй назбіралі малін і парэчак.
4. Я саставіла букет з кветак. У ім былі рамонкі або гваздзікі.

Падрыхтуйце да кожнага з выказванняў Кіры рысункі, улічваючы, што ўсе выказванні праўдзівыя.

3 Адкрыйце файл з рысункам і раскладзіце грыбы па кошыках так, каб было праўдзівым наступнае выказванне: «У вялікім кошыку ўсе грыбы ядомыя, і ў маленькім кошыку ўсе грыбы неядомыя».

4 Адкрыйце файл з рысункам і пастаўце ўсе кветкі ў вазы так, каб было праўдзівым выказванне: «У сіняй вазе ўсе кветкі ружы, або ў чырвонай вазе ўсе кветкі не чырвонага колеру».

5* Знайдзіце значэнні лагічных выразаў, калі $A = 1$, $B = 1$, $C = 0$, $D = 0$.

1. A **АБО** B **І НЕ** C.
2. A **І НЕ** B **АБО** C.
3. A **АБО** B **І НЕ** (C **І** D).
4. (A **І** B) **АБО НЕ** C **І** (A **АБО** B) **АБО НЕ** D.

§ 5. Множествы

5.1. Паняцце множества

Разгледзім выказванне «Усе навучэнцы нашага класа маюць дома камп'ютар». Праўдзівае яно ці непраўдзівае? Для адказу на гэта пытанне вам трэба ў кожнага з аднакласнікаў удакладніць: «У цябе дома ёсць камп'ютар?» Калі ўсе навучэнцы класа адкажуць сцвярджальна, то выказванне праўдзівае, калі хоць бы адзін з навучэнцаў адкажа «няма», то і выказванне будзе непраўдзівым. Для розных класаў гэта выказванне будзе мець розныя значэнні, таму што рознымі будуць множествы навучэнцаў класа.

Множества — сукупнасць якіх-небудзь аб'ектаў, што валодаюць агульнай уласцівасцю. Гэтыя аб'екты называюць **элементамі множества.**

Можна гаварыць пра множества навучэнцаў 7 А класа, множества адзнак у класным журнале, множества гарадоў Беларусі, множества літар рускага алфавіта і г. д. Паняцце множества з'яўляецца адным з асноўных у матэматыцы.

Множествы, як правіла, абазначаюць вялікімі лацінскімі літарамі,

Множествы, у тым ліку і бясконцыя, у нейкай форме выкарыстоўваліся ў матэматыцы з часоў Старажытнай Грэцыі.

Да XIX ст. лічылася, што дакладнага азначэння множества няма. Множествам называлі любую сукупнасць, аб'яднанне прадметаў.

У канцы XIX ст. нямецкі матэматык Георг Кантар (1845—1918) вызначыў множества як «адзінае імя для сукупнасці ўсіх аб'ектаў, якія валодаюць дадзенай уласцівасцю».

Паводле тэорыі Г. Кантара некаторыя множествы канчатковыя (напрыклад, цэлыя лікі ад 1 да 7), а некаторыя — бясконцыя (напрыклад, цэлыя лікі). У некаторых выпадках элементы аднаго множества строга адпавядаюць элементам іншага множества, напрыклад множества колераў вясёлкі і множества цэлых лікаў ад 1 да 7.

Прыклад 5.1. Няхай M — мноства любімых вучэбных прадметаў сямікласніка Ігара, якое складаецца з элементаў: матэматыка, інфарматыка, англійская мова. Тады можна запісаць:

M = {матэматыка, інфарматыка, англійская мова};

інфарматыка $\in M$;

літаратура $\notin M$.

Прыклад 5.2. Няхай у мноства M уваходзяць усе вучэбныя прадметы, якія вывучаюцца ў 7-м класе. Пералічыць усе яго элементы можна, напрыклад, гледзячы на старонку школьнага дзённіка. Тады можна запісаць:

інфарматыка $\in M$;

астраномія $\notin M$.

Леанард Эйлер (1707—1783) — вучоны, які ўнёс значны ўклад у развіццё матэматыкі і механікі, а таксама фізікі, астраноміі і шэрага прыкладных навук.

Распрацаваў зручны метад для графічнага адлюстравання адносін паміж мноствамі.

а элементы мностваў — маленькімі. Нагадаем, што для абазначэння прыналежнасці элемента мноству выкарыстоўваюць спецыяльныя знакі:

$a \in M$ (элемент a належыць мноству M), $a \notin M$ (элемент a не належыць мноству M). Калі мноства M складаецца з элементаў a, b, c, то гэта запісваюць так: $M = \{a, b, c\}$.

Каб задаць мноства, неабходна пералічыць яго элементы (прыклад 5.1) або назваць іх агульную ўласцівасць (прыклад 5.2).

5.2. Паняцце падмноства

Разгледзім мноства навучэнцаў якога-небудзь класа. У гэтым мноства можна вылучыць не толькі асобнага навучэнца, але і некаторыя групы навучэнцаў: выдатнікі; навучэнцы, што ўмеюць гуляць у тэніс, і г. д. Кожная з такіх груп утварае падмноства — частку мноства навучэнцаў.

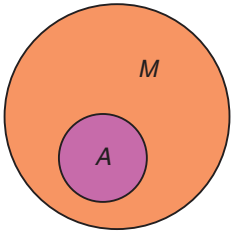
Калі мноства A з'яўляецца падмноствам мноства M, то гэта запісваюць так: $A \subset M$. Запіс $A \not\subset M$ абазначае, што мноства A не з'яўляецца падмноствам мноства M.

Падмноства можа змяшчаць усе элементы мноства, а можа не

змяшчаць ніводнага (пустое мноства; абазначаецца знакам ∅).

Некаторыя элементы мноства могуць належаць адначасова розным падмноствам (прыклад 5.3).

Для наглядная геаметрычнай ілюстрацыі мностваў і адносін паміж імі выкарыстоўваюць кругі Эйлера. Кожнае мноства адлюстроўваецца кругам. Калі якое-небудзь мноства з'яўляецца падмноствам іншага мноства, то адзін круг адлюстроўваецца ўнутры іншага. Напрыклад, калі M — мноства ўсіх драпежнікаў, A — мноства ўсіх ільвоў ($A \subset M$), то гэта абазначаецца так:



Прыклад 5.3. Няхай M = {Вера, Сяргей, Вася, Віка, Ліза, Косця, Надзя} — мноства навучэнцаў 7 Б класа, якія займаюцца ў драмгуртку. З гэтага мноства можна вылучыць наступныя падмноствы:

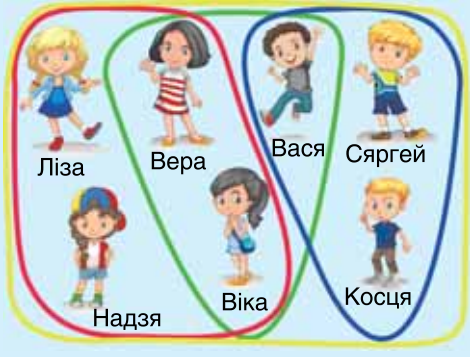
A = {Вера, Віка, Ліза, Надзя} — мноства дзяўчынак (чырвоная мяжа).

B = {Сяргей, Вася, Косця} — мноства хлопчыкаў (сіняя мяжа).

C = {Вера, Вася, Віка} — мноства дзяцей, чые імёны пачынаюцца на літару «В» (зялёная мяжа).

D = {Вера, Сяргей, Вася, Віка, Ліза, Косця, Надзя} — мноства дзяцей, у імёнах якіх па 2 галосныя гукі (жоўтая мяжа).

$E = \varnothing$ — мноства трохгадовых дзяцей.



1. Што разумеюць пад мноствам?
2. Прывядзіце прыклады мностваў.
3. Што разумеюць пад падмноствам?
4. Што выкарыстоўваецца для геаметрычнай ілюстрацыі мностваў?
5. Што разумеюць пад пустым мноствам? Як яно абазначаецца?
6. Ці можа элемент мноства адначасова належаць розным падмноствам?

Практыкаванні

1 Дапоўніце кожнае з мностваў 1—2 элементамі.

1. A = {матэматыка, інфарматыка, гісторыя, літаратура}.
2. B = {яблык, груша, апельсін, банан}.
3. C = {клавіятура, манітор, мыш}.
4. D = {аловак, ручка, рызінка, фламастар}.

2 Якія элементы могуць уваходзіць у наступныя мноствы?

1. Транспартныя сродкі.
2. Колеры вясёлкі.
3. Хатнія жывёлы.
4. Цотныя лікі.

3 Адкрыйце файл з групамі слоў. Падзяліце словы кожнай групы на два мноствы. Словы першага мноства вылучыце чырвоным колерам, а другога — сінім. Па якіх прыметах вы падзялілі словы?

Узор:

1. Тэкст у файле: гусь, лебедзь, заяц, воўк, паўлін, курыца, дзік, лось.

Вынік: A = {гусь, лебедзь, паўлін, курыца};
B = {заяц, воўк, дзік, лось}.

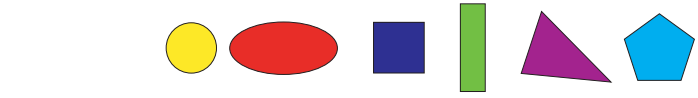
Прыметы: A — мноства птушак, B — мноства звяроў.

2. Мяч, стол, крэсла, канькі, шафа, клюшка, шайба, камода.
3. Сом, вуж, карась, акунь, шчупак, гадзюка, кобра, пітон.

4 З мноства геаметрычных фігур A = {круг, авал, квадрат, прамавугольнік, трохвугольнік, пяцівугольнік} вылучыце падмноствы:

1. Фігур, што не маюць вуглоў.
2. Фігур, што з'яўляюцца чатырохвугольнікамі.
3. Фігур, колькасць вуглоў у якіх большая за тры.

5 Адкрыйце файл з рысункамі геаметрычных фігур. З дапамогай аперацыі капіравання стварыце падмноствы *а*, *б*, *в*, якія валодаюць прыметамі, названымі ў заданні 4. Усе элементы кожнага падмноства змясціце ўнутры адпаведнага прамавугольніка.



а б в

6 Адкрыйце файл з рысункамі матылькоў. Выкарыстоўваючы аперацыю капіравання, стварыце ніжэйпералічаныя падмноствы і змясціце іх у прамавугольніках.

1. Матылькі, у афарбоўцы якіх ёсць сіні колер.
2. Матылькі, у афарбоўцы якіх ёсць чырвоны колер.
3. Матылькі, у афарбоўцы якіх ёсць зялёны колер.
4. Матылькі, у афарбоўцы якіх ёсць жоўты колер.

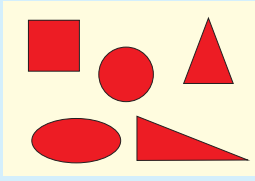
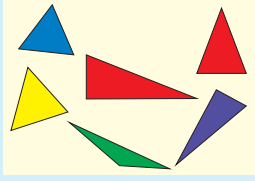
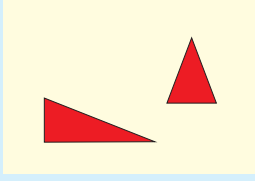
7 Зададзены два мноствы: *K* — мноства кніг у школьнай бібліятэцы; *U* — мноства вучэбных дапаможнікаў у гэтай жа бібліятэцы. Якое з мностваў з'яўляецца падмноствам іншага? Адлюструйце іх з дапамогай кругоў Эйлера.

8 Складзіце ланцужок уключэнняў так, каб кожнае наступнае мноства з'яўлялася падмноствам папярэдняга: *A* — мноства ўсіх прамавугольнікаў; *B* — мноства ўсіх чатырохвугольнікаў; *C* — мноства ўсіх квадратаў; *D* — мноства ўсіх многавугольнікаў.

9* Прыдумайце прыклады ланцужкоў, якія складаюцца з мностваў і іх падмностваў і змяшчаюць не менш за тры ўключэнні.

§ 6. Аперацыі над мноствамі

Прыклад 6.1. Знойдзем перасячэнне мностваў A і B.

Мноства A	Фігуры чырвонага колеру 
Мноства B	Трохвугольнікі 
Мноства $A \cap B$	Трохвугольнікі чырвонага колеру 

Прыклад 6.2. Знойдзем перасячэнне мностваў A і B.

Мноства A — жывёлы, якія ўмеюць лятаць: пчала, журавель, майскі жук, верабей, бусел, страказа.

Мноства B — птушкі: страус, журавель, пінгвін, бусел, курыца, верабей.

Перасячэнне $A \cap B$ = {журавель, бусел, верабей} — птушкі, якія ўмеюць лятаць.

Для мностваў, як і для выказванняў, вызначаны свае аперацыі. Такімі аперацыямі з'яўляюцца аперацыі перасячэння і аб'яднання мностваў.

Разгледзім мноства навучэнцаў 7-га класа. Вылучым у ім два падмноствы: мноства аматараў гульні ў настольны тэніс і мноства навучэнцаў, у якіх дома ёсць камп'ютар. Некаторыя з навучэнцаў могуць і мець камп'ютар, і захапляцца тэнісам. Значыць, яны будуць уваходзіць у абодва падмноствы.

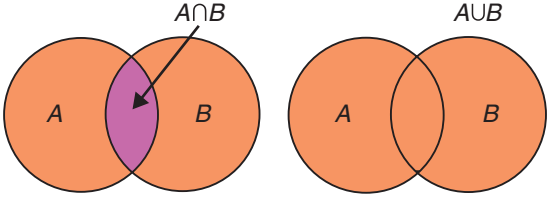
Перасячэннем мностваў A і B называецца мноства, у якое ўваходзяць толькі тыя элементы, што належаць як мноству A, так і мноству B. Для абазначэння аперацыі перасячэння выкарыстоўваецца знак $\cap$. Узоры выканання заданняў на знаходжанне перасячэння мностваў паказаны ў прыкладах 6.1 і 6.2.

Вылучым сярод навучэнцаў 7-га класа два падмноствы: мноства аматараў гульні ў настольны тэніс і мноства аматараў гульні ў вялікі тэніс. Тады мноства аматараў тэніса будзе ўключаць у сябе і тых, хто гуляе ў настольны тэніс, і тых, хто гуляе ў

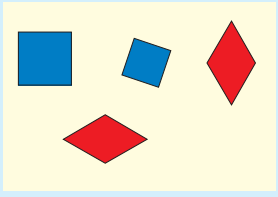
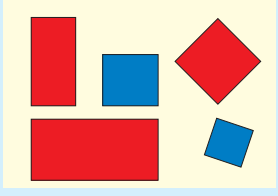
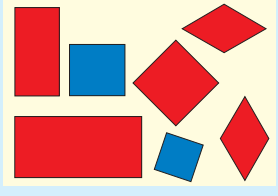
вялікі тэніс. Калі хтосьці гуляе і ў вялікі, і ў настольны тэніс, то ён таксама будзе ўваходзіць у мноства аматараў тэніса.

Аб'яднаннем мностваў A і B называецца мноства, у якое ўваходзяць элементы, што належаць хоць бы аднаму з мностваў A ці B. Для абазначэння аперацыі аб'яднання мностваў выкарыстоўваецца знак $\cup$. Узор выканання задання на аб'яднанне мностваў паказаны ў прыкладзе 6.3.

Перасячэнне і аб'яднанне двух мностваў можна адлюстраваць з дапамогай кругоў Эйлера.



Прыклад 6.3. Знойдзем аб'яднанне мностваў A і B.

Мноства A	Ромбы 
Мноства B	Прамавугольнікі 
Мноства $A \cup B$	Чатырохвугольнікі 

1. Што называюць перасячэннем мностваў?
2. Што называюць аб'яднаннем мностваў?
3. Як абазначаюцца аперацыі перасячэння і аб'яднання мностваў?

Практыкаванні

1 Знайдзіце перасячэнне і аб'яднанне мностваў A і B.

1. A = {матэматыка, інфарматыка, гісторыя, літаратура};
   B = {англійская мова, матэматыка, хімія, гісторыя}.
2. A = {яблык, апельсін, мандарын, лімон, ківі};
   B = {апельсін, персік, мандарын, груша, лімон}.

2 Зададзены два мноствы. Знайдзіце іх перасячэнне і аб'яднанне.

1. Мноства задач, якія рашаюцца з дапамогай праграмы *графічны рэдактар* = {адкрыць, захаваць, стварыць, заліўка колерам, друк}.

2. Мноства задач, якія рашаюцца з дапамогай праграмы *тэкставы рэдактар* = {адкрыць, захаваць, стварыць, павялічыць памер шрыфту, друк}.

3 Рашыце задачы з выкарыстаннем кругоў Эйлера (нарысуйце іх у графічным рэдактары).

1. Пра навучэнцаў школы, якія ўдзельнічалі ў фізіка-матэматычным конкурсе, вядома, што 7 з іх рашылі задачы і па матэматыцы, і па фізіцы, 11 — задачы па матэматыцы, 9 — задачы па фізіцы. Колькі навучэнцаў прымалі ўдзел у конкурсе?

2. У кіёску каля школы прадаецца марожанае двух відаў: «Эскімо» і «Пламбір». Пасля ўрокаў 24 сямікласнікі купілі марожанае. Пры гэтым 15 з іх выбралі «Эскімо», а 17 — «Пламбір». Колькі сямікласнікаў купілі марожанае двух відаў?

3*. Са 100 турыстаў, якія ад'язджаюць у падарожжа, нямецкай мовай валодаюць 30 чалавек, англійскай — 28, французскай — 42. Англійскай і нямецкай мовамі адначасова валодаюць 8 чалавек, англійскай і французскай — 10, нямецкай і французскай — 5, усімі трыма мовамі — 3. Колькі турыстаў не валодаюць ніводнай мовай?

4* Выкарыстоўваючы рысунак, выканайце заданні.

1. Стварыце два падмноствы мноства дзяўчынак. Для ўсіх дзяўчынак, якія ўваходзяць у першае падмноства, праўдзівым з'яўляецца выказванне: «Дзяўчынка носіць штаны сіняга колеру, **І** на яе майцы ёсць чырвоны колер». Для ўсіх дзяўчынак, якія ўваходзяць у другое падмноства, праўдзівае выказванне: «Дзяўчынка апранута не ў штаны **АБО** мае валасы жоўтага колеру».

2. Знайдзіце перасячэнне і аб'яднанне гэтых мностваў.
3. Колькі дзяўчынак не трапіла ні ў адно падмноства?
4. Выканайце практыкаванне ў графічным рэдактары. Вакол дзяўчынак з першага мноства нарысуйце мяжу чырвоным колерам, а вакол дзяўчынак з другога мноства — сінім. Вобласць перасячэння абазначце жоўтым колерам.

§ 7. Выкарыстанне лагічных аперацый для пабудовы пошукавых запытаў у Інтэрнэце

7.1. Пошук інфармацыі

У сучаснае стагоддзе інфармацыйных тэхналогій людзі выкарыстоўваюць для пошуку інфармацыі сэрвісы Інтэрнэту. Пошукавыя сістэмы ўвесь час збіраюць, сістэматызуюць і захоўваюць інфармацыю па ўсім свеце. Пошук інфармацыі ў пошукавай сістэме ажыццяўляецца па запыце.

Пад **запытам** у пошукавай сістэме разумеюць набор слоў, фраз, сімвалаў, якія карыстальнік уводзіць у радок пошуку, каб знайсці інфармацыю, што цікавіць яго.

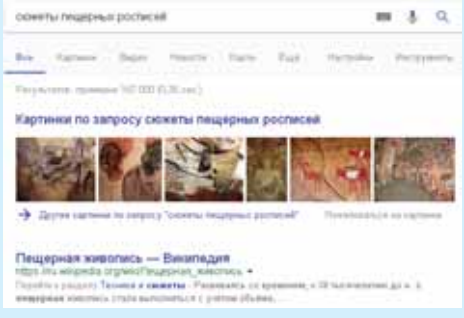
Сучасныя пошукавыя сістэмы дазваляюць ажыццяўляць пошук па галасавых запытах ці выкарыстоўваць у якасці запыту выяву.

Вынікам пошуку з'яўляецца пералік сайтаў (прыклад 7.1 і

Пры пошуку інфармацыі ў Інтэрнэце важныя паўната, дакладнасць і актуальнасць атрыманых вынікаў. Карыстальнік можа паўплываць на якасць вынікаў пошуку, калі будзе:

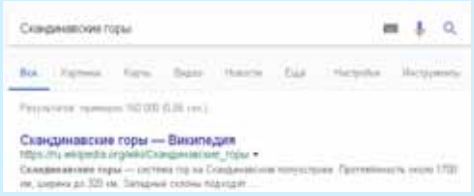
- прадумана выбіраць пошукавую службу;
- улічваць асаблівасці пошукавай сістэмы;
- граматна фармуляваць запыты на пошук інфармацыі.

Прыклад 7.1. Знойдзем інфармацыю пра сюжэты пячорных роспісаў.

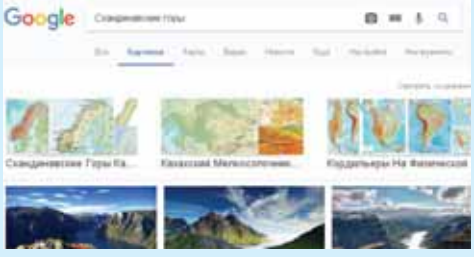


У выніку пошуку знойдзена больш за 160 тыс. сайтаў, якія змяшчаюць шуканую інфармацыю.

Прыклад 7.2. Знойдзем інфармацыю пра Скандынаўскія горы.



У выніку пошуку знойдзена 160 тыс. сайтаў. Можна перайсці ў раздзел «Карцінкі»:



Прыклад 7.3. Знойдзем інфармацыю пра бактэрыі ў жыцці чалавека.

Знойдзена больш за 1 млн сайтаў. Калі нас не цікавяць бактэрыі, якія выклікаюць хваробы, то пошукавы запыт можна змяніць, дадаўшы ў канцы «-хваробы». Колькасць знойдзеных сайтаў скароціцца да 428 тыс.

прыклад 7.2). Колькасць знойдзеных сайтаў можа быць вельмі вялікай, і прагледзець іх усе часта немагчыма. На практыцы карыстальнікі Інтэрнэту звычайна праглядаюць 5—15 сайтаў, знойдзеных першымі.

Выніковасць пошуку ў значнай ступені залежыць ад уменння карыстальніка карэктна сфармуляваць пошукавы запыт. Граматная фармулёўка фразы ці выбар слоў для пошуку дазволяць атрымаць больш дакладны вынік.

7.2. Скарачэнне вобласці пошуку

Для кожнага з сайтаў, знойдзеных у выніку пошукавага запыту, будзе праўдзівым наступнае выказванне: «На старонцы сайта прысутнічае інфармацыя, якая адпавядае пошукаваму запыту». Усе такія сайты ўтвараюць мноства сайтаў, што задавальняюць пошукавы запыт.

Пры пабудове пошукавага запыту некаторыя сайты можна выключыць з разгляду. Для гэтага да асноўнага запыту дадаецца слова са знакам мінус («-») перад ім. Сайты, якія змяшчаюць словы, адзначаныя гэтым знакам,

не будуць уключаны ў спіс знойдзеных (прыклады 7.3 і 7.4).

Атрыманы пералік сайтаў утварае падмноства мноства сайтаў, што задавальняюць асноўны запыт. Для ўсіх такіх сайтаў выказванне «На старонцы сайта прысутнічае інфармацыя, якая адпавядае слову, адзначанаму знакам “-”» будзе ўспрымацца пошукавай сістэмай як непраўдзівае.

7.3. Выкарыстанне аператараў у пошукавых запытах

Аператары пошуку — словы або сімвалы, якія дадаюцца да пошукавых запытаў для ўдакладнення вынікаў.

Аператар «+» дазваляе ажыццяўляць пошук дакументаў, дзе абавязкова прысутнічае слова, што стаіць за сімвалам. Дапушчальна выкарыстоўваць некалькі аператараў «+» у адным запыце (прыклад 7.5).

Аператар «-» мы разгледзелі ў папярэднім пункце параграфа.

Аператар «*» замяняе любое невядомае слова ў запыце (прыклад 7.6).

Калі змясціць словы ці фразу ў двукоссе, то ў выніках пошуку будуць паказаны толькі тыя

Прыклад 7.4. Знойдзем значэнне паняцця «шчыт».

У выніку пошуку будзе знойдзена больш за 20 млн сайтаў, прычым на некалькіх першых старонках знаходзіцца інфармацыя пра серыялы, фільмы, магазіны.

Для ўдакладнення інфармацыі ўвядзём запыт «шчыт -магазін -серыял -зброя». У такім выпадку колькасць спасылак скароціцца да 7 млн 270 тыс.

Прыклад 7.5. Знойдзем сайты, дзе ёсць інфармацыя пра кожнага з пісьменнікаў: М. дэ Сервантэса, У. Шэкспіра і Ф. Рабле.

Прыклад 7.6. Складзём запыт для пошуку поўнай цытаты «Тучки ... разгоняючи, Заря ... подымается;». З якога яна твора? Хто яго аўтар?

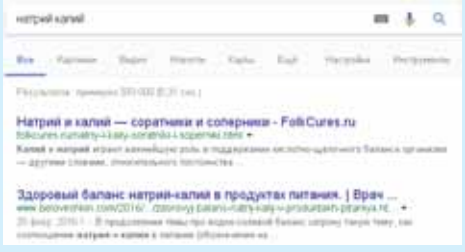
Шматкроп’е ў пошукавым запыце заменім знакам *.

Як бачым, гэта цытата з твора М. Ю. Лермантава «Песня про царя Ивана Васильевича, молодого опричника и удалого купца Калашникова».

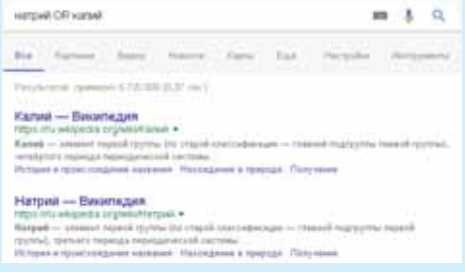
Прыклад 7.7. Знойдзем інфармацыю пра натрый ці калій.

Параўнаем вынікі запытаў «натрый калій» і «натрый OR калій».

Па першым запыце спачатку размешчаны спасылкі на сайты, якія змяшчаюць інфармацыю пра два хімічныя элементы, а затым — пра кожны з іх:



Па другім запыце спачатку размешчаны спасылкі на сайты пра асобныя элементы, а потым — агульная інфармацыя:



старонкі, на якіх гэтыя словы (фразы) размешчаны ў тым жа парадку, што і ў запыце ў двукоссі. Двукоссе выкарыстоўваецца тады, калі неабходна знайсці дакладнае слова ці фразу, цытату.

Аператары, што разглядаюцца далей, маюць розныя абазначэнні для розных пошукавых сістэм (напрыклад, для Google і Яндэкс).

Аператар **OR** (пошукавая сістэма Google) дазваляе знайсці старонкі, якія змяшчаюць хоць бы адно з некалькіх слоў, і адпавядае лагічнай аперацыі **АБО** (прыклад 7.7). Для пошукавай сістэмы Яндэкс аналагічны аператар абазначаецца |.

Некаторыя з аператараў могуць не мець аналагаў у іншых пошукавых сістэмах. Аператар & пошукавай сістэмы Яндэкс ажыццяўляе пошук дакументаў, у якіх словы запыту, аб'яднаныя дадзеным аператарам, сустракаюцца ў адным сказе.

1. Што называюць запытам у пошукавай сістэме?
2. Як выключыць некаторыя запісы з вобласці пошуку?
3. Якія аператары можна выкарыстоўваць у пошукавых запытах?

Практыкаванні

1 Знайдзіце з дапамогай розных пошукавых сістэм інфармацыю пра бегавыя віды лёгкай атлетыкі. Запішыце вынікі ў табліцу (у сшытак або ў электронным выглядзе). Параўнайце атрыманыя вынікі.

Пошукавая сістэма	Колькасць вынікаў пошуку

2 Знайдзіце з дапамогай пошукавай сістэмы выявы манет Вялікага Княства Літоўскага. Выпішыце ў сшытак 5—6 назваў манет.

3 Сфармулюйце запыт па пошуку сюжэтаў пячорных роспісаў, які выключае роспісы храмаў.

4 Складзіце запыт для пошуку поўнай цытаты «Старость боится… Жизнь я… куплю». Хто аўтар гэтай фразы? У якім творы яна сустракаецца?

5 З дапамогай адпаведных пошукавых запытаў атрымайце адказ на пытанне: якая з падзей адбылася раней — адкрыццё Мендзялеевым перыядычнага закона ці вынаходства Эдысанам фанографа?

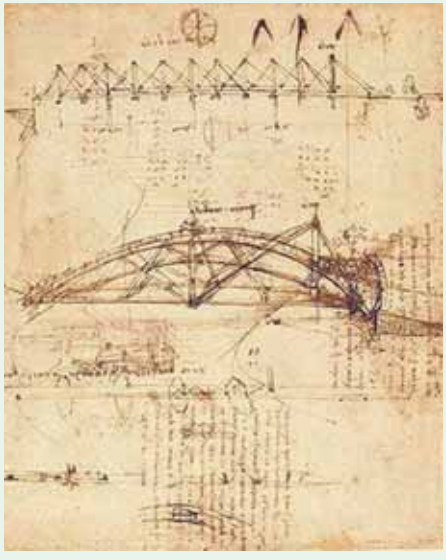
6 Складзіце запыты на пошук назваў беларускіх азёр, выкарыстоўваючы інфармацыю з табліцы. Запішыце назвы знойдзеных азёр у сшытак.

Бярэзінскі біясферны запаведнік	П							
Знаходзіцца на Палессі, трэцяе па плошчы ў Беларусі					О			
Расонскі раён Віцебскай вобласці			Ш					
Браслаўскі раён Віцебскай вобласці			У					
Найбуйнейшае сярод Блакітных азёр						К		

Глава 3
АСНОЎНЫЯ АЛГАРЫТМІЧНЫЯ КАНСТРУКЦЫІ

§ 8. Алгарытмы і выканаўцы

Алгарытмы пабудовы чарцяжоў чалавек распрацоўвае з глыбокай старажытнасці. З'яўленне чарцяжоў звязана з практычнай дзейнасцю чалавека — узвядзеннем умацаванняў і гарадскіх будынкаў. Першыя звесткі пра чарцяжы, што нагадваюць сучасныя, звязаны з імем Леанарда да Вінчы (1452—1519) — італьянскага вучонага і мастака, які ў тэхнічных рысунках і эскізах раскрываў свае ідэі ў галіне тэхнікі і будаўніцтва.



8.1. Паняцце алгарытму

Успомнім некаторыя паняцці, вывучаныя ў 6-м класе.

Алгарытм — зразумелая і канечная паслядоўнасць дакладных дзеянняў (каманд), фармальнае выкананне якіх дазваляе атрымаць рашэнне пастаўленай задачы.

Выканаўца алгарытму — чалавек, група людзей або тэхнічнае ўстройства, якія разумеюць каманды алгарытму і ўмеюць правільна іх выконваць.

Сістэма каманд выканаўцы — каманды, якія разумее і можа выконваць выканаўца.

Любы выканаўца мае абмежаваную сістэму каманд. Усе яны падзяляюцца на групы:

1. Каманды, якія непасрэдна выконвае выканаўца.
2. Каманды, якія мяняюць парадак выканання іншых каманд выканаўцы.

Камп'ютар з'яўляецца ўніверсальным выканаўцам.

Запіс алгарытму ў выглядзе паслядоўнасці каманд, якую можа выканаць камп'ютар, называюць **праграмай.**

Існуюць наступныя спосабы ўяўлення алгарытмаў:

- **слоўны** (апісанне алгарытму сродкамі натуральнай мовы з дакладнай і канкрэтнай фармулёўкай фраз);
- **графічны (блок-схема)** (графічнае адлюстраванне каманд алгарытму з выкарыстаннем геаметрычных фігур, або блокаў, і стрэлак, што злучаюць гэтыя блокі і паказваюць на парадак выканання каманд);
- **праграмны** (запіс алгарытму ў выглядзе праграмы).

(Схематычна дадзеныя спосабы паказаны ў прыкладзе 8.1.)

8.2. Выканаўца Чарцёжнік

У 6-м класе вы пазнаёміліся з выканаўцам Чарцёжнік, прызначаным для пабудовы рысункаў і чарцяжоў на каардынатнай плоскасці (прыклад 8.2).

Чарцёжнік мае пяро, якім ён можа рысаваць адрэзкі на плоскасці. У зыходным становішчы пяро ўзнята і знаходзіцца над пунктам (0, 0) — пачаткам каардынат. Пасля завяршэння рысавання пяро таксама павінна быць узнята.

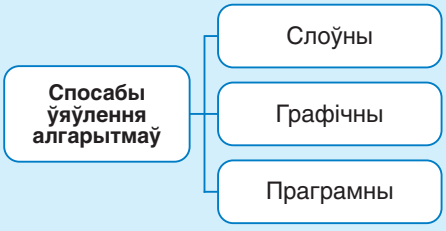
У цяперашні час чарцяжы шырока ўжываюцца ў розных галінах будаўніцтва, сельскай гаспадаркі, прамысловасці і г. д. Сёння для пабудовы чарцяжоў выкарыстоўваюцца спецыяльныя праграмы, якія дазваляюць аўтаматызаваць працэс чарчэння. Вось лагатыпы такіх праграм:

AutoCAD

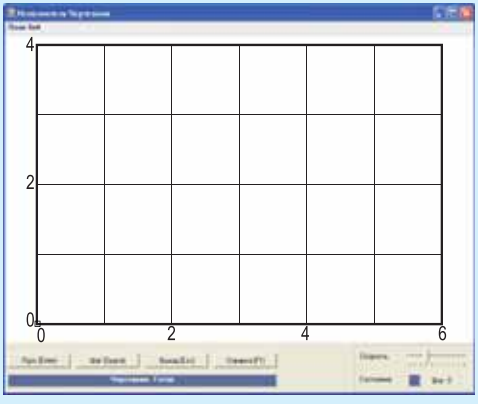

Компас-3D


NanoCAD

Прыклад 8.1.



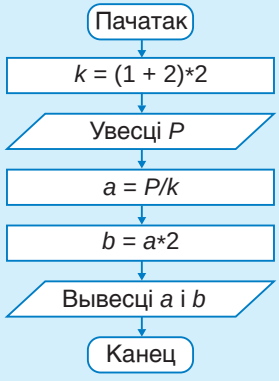
Прыклад 8.2. Поле выканаўцы Чарцёжнік.



Прыклад 8.3. Запіс алгарытму па дзеяннях:

1) 1 + 2 = 3 (часткі);
2) 3 · 2 = 6 (частак);
3) 120 : 6 = 20 (м);
4) 20 · 2 = 40 (м).

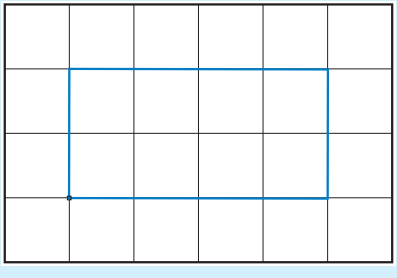
Блок-схема алгарытму:



Праграма для выканаўцы:

```
uses Drawman;
begin
  Field(6, 4);
  ToPoint(1, 1);
  PenDown;
  OnVector(4, 0);
  OnVector(0, 2);
  OnVector(-4, 0);
  OnVector(0, -2);
  PenUp;
end.
```

Вынік работы праграмы:



Сістэма каманд Чарцёжніка:

Каманда	Дзеянне
`ToPoint(x,y)`	Перамясціць пяро ў пункт (x, y)
`PenUp`	Узняць пяро
`PenDown`	Апусціць пяро
`Field(n,m)`	Стварыць поле памерам $n \times m$
`OnVector(a,b)`	Зрушыць пяро на a адзінак па гарызанталі і b адзінак па вертыкалі

Прыклад 8.3. Складзём алгарытм рашэння задачы.

Прамавугольны ўчастак, даўжыня якога ў 2 разы большая за шырыню, абгарадзілі плотам даўжынёй 120 м. Вызначыце даўжыню і шырыню ўчастка. Напішыце праграму, выканаўшы якую выканаўца Чарцёжнік пабудуе чарцёж плота гэтага ўчастка. Маштаб: 1 клетка роўна 10 м.

Слоўнае апісанне алгарытму:

1. Даўжыня ўчастка ў 2 разы большая за шырыню, таму ў суме даўжыня і шырыня складуць 3 аднолькавыя часткі. Плот абгароджвае ўчастак па перыметры, роўным падвоенай суме даўжыні і шырыні, г. зн. перыметр роўны 6 аднолькавым часткам.

2. Шырыня: 120 : 6 = 20 м.

3. Даўжыня ў 2 разы большая за шырыню: 20 · 2 = 40 м.

8.3. Алгарытмічная канструкцыя *паслядоўнасць*

Існуе вялікая колькасць алгарытмаў, у якіх усе каманды выконваюцца паслядоўна адна за адной у тым парадку, у якім яны запісаны. У такіх алгарытмах адсутнічаюць каманды, што мяняюць парадак выканання іншых каманд. Такія праграмы вы складалі ў мінулым годзе для выканаўцы Чарцёжнік.

Алгарытмічная канструкцыя *паслядоўнасць* — паслядоўнасць каманд алгарытму, што выконваюцца ў тым парадку, у якім яны запісаны.

Алгарытмічная канструкцыя *паслядоўнасць* адлюстроўвае натуральны, паслядоўны парадак выканання дзеянняў у алгарытме.

Паслядоўнасць выкарыстоўвалася ў прыкладзе 8.3, у якім апісваліся алгарытмы вылічэння даўжыні і шырыні ўчастка і пабудовы прамавугольніка выканаўцам Чарцёжнік.

Алгарытмічная канструкцыя *паслядоўнасць* паказана ў прыкладах 8.4 і 8.5.

Прыклад 8.4. Алгарытм гатавання бутэрброда:

1. Адрэзаць лустачку батона.
2. Пакласці на батон ліст салаты.
3. Адрэзаць кавалачак вяндліны.
4. Пакласці вяндліну на ліст салаты.
5. Адрэзаць кавалачак памідора.
6. Пакласці памідор на вяндліну.

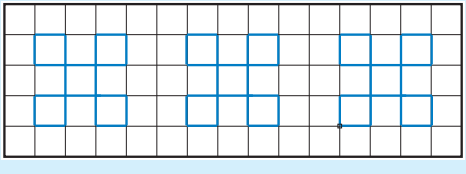
Прыклад 8.5. Алгарытм выканання лабараторнай работы па біялогіі «Будова інфузорыі туфелькі»:

1. Разгледзець знешні выгляд і ўнутраную будову інфузорыі туфелькі.
2. Зарысаваць інфузорыю туфельку і абазначыць назвы яе органаў.
3. Падвесці вынік работы.

Прыклад 8.6. Праграма для выканаўцы Чарцёжнік будзе наступнай:

```
uses Drawman;
procedure figura;
begin
  PenDown;
  OnVector(1,  0);
  OnVector(0,  3);
  OnVector(-1,  0);
  OnVector(0,  -1);
  OnVector(3,  0);
  OnVector(0,  1);
  OnVector(-1,  0);
  OnVector(0,  -3);
  OnVector(1,  0);
  OnVector(0,  1);
  OnVector(-3,  0);
  OnVector(0,  -1);
  PenUp;
end;
begin
  Field(15,  5);
  ToPoint(1,  1);
  Figura;
  ToPoint(6,  1);
  Figura;
  ToPoint(11,  1);
  Figura;
end.
```

Вынік выканання праграмы:



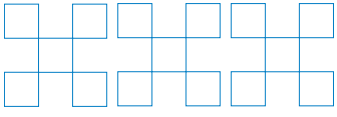
8.4. Дапаможныя алгарытмы

Часта ў адной праграме трэба рысаваць адзін і той жа відарыс некалькі разоў. Атрыманне гэтага відарыса зручна аформіць у выглядзе дапаможнага алгарытму, які можна выкарыстоўваць патрэбную колькасць разоў.

Дапаможны алгарытм — алгарытм, што цалкам выкарыстоўваецца ў складзе іншага алгарытму.

Дапаможны алгарытм рашае некаторую падзадачу асноўнай задачы. Выклік дапаможнага алгарытму ў праграме замяняе некалькі каманд адной.

Прыклад 8.6. Напішам праграму, выканаўшы якую Чарцёжнік нарысуе відарыс:



Дадзены рысунак складаецца з аднолькавых фігур. Для рысавання адной з іх можна аформіць дапаможны алгарытм `figura`.

Апісанне асноўнага алгарытму:

```
  перамяшчэнне ў пачатковы пункт;
  рысаванне фігуры;
  перамяшчэнне да другой фігуры;
  рысаванне фігуры;
```

перамяшчэнне да трэцяй фігуры;

рысаванне фігуры.

Пры рашэнні задач над праектам могуць працаваць некалькі чалавек. Кожны з членаў калектыву робіць частку сваёй працы і афармляе яе як асобны дапаможны алгарытм.

Пабудову алгарытмаў часта выконваюць **метадам пашагавай дэталізацыі**. Пры гэтым складаная задача разбіваецца на шэраг больш простых. Для кожнай падзадачы складаецца свой дапаможны алгарытм. Падзадачы могуць разбівацца на яшчэ больш простыя падзадачы.

1. Што такое алгарытм?
2. Якія спосабы запісу алгарытмаў вам вядомыя?
3. Што называюць алгарытмічнай канструкцыяй *паслядоўнасць*?
4. Які алгарытм называецца дапаможным?
5. Для чаго патрэбны дапаможныя алгарытмы?

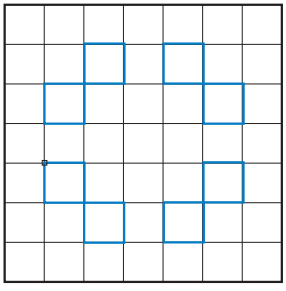
Практыкаванні

1 Які рысунак атрымаецца пасля выканання Чарцёжнікам наступнай праграмы? Адлюструйце рысунак і праверце правільнасць сваіх дзеянняў, выканаўшы праграму на камп'ютары.

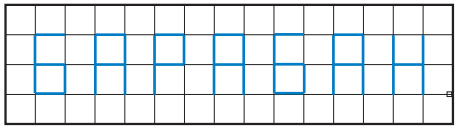
```
uses Drawman;
begin
  Field(8, 8);
  ToPoint(2, 1);
  PenDown;
  OnVector(4, 0);
  OnVector(0, 1);
  OnVector(1, 0);
  OnVector(0, 4);
  OnVector(-1, 0);
  OnVector(0, 1);
  OnVector(-4, 0);
  OnVector(0, -1);
  OnVector(-1, 0);
  OnVector(0, -4);
  OnVector(1, 0);
  OnVector(0, -1);
  PenUp;
end.
```

2 Напішыце для выканаўцы Чарцёжнік праграмы для атрымання наступных відарысаў:

а



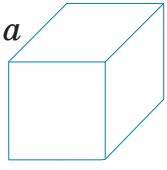
б



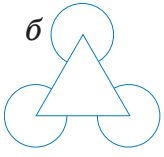
3 Прыдумайце свае рысункі і складзіце праграмы для іх рысавання з дапамогай выканаўцы Чарцёжнік.

4* Прааналізуйце рысункі. Якія з іх мог нарысаваць выканаўца Чарцёжнік? Чаму? Якія каманды вы можаце прапанаваць дадаць выканаўцы для выканання астатніх рысункаў?

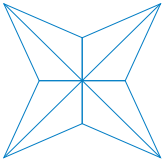
а



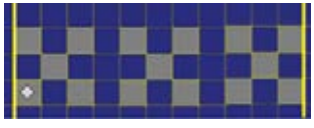
б



в



г



§ 9. Выканаўца Робат

9.1. Робаты ў жыцці чалавека

Робаты развозяць заказы ў рэстаране ў г. Харбін (Кітай)[1].

Чалавек з глыбокай старажытнасці марыў пра штучнае стварэнне, якое магло б выконваць яго загады. Сёння гэта мара стала рэальнасцю — у жыцці людзей з'явіліся робаты. Яны здольныя выконваць практычна любую працу, даступную чалавеку, а

[1]Матэрыялы пра робатаў узяты з сайтаў http://www.robogeek.ru і http://fishki.net/1211999-roboty-v-nashej-zhizni.html (дата доступу: 07.02.2017).

таксама рабіць шмат якія рэчы, што людзям выканаць складана ці наогул немагчыма.

Робаты выкарыстоўваюцца на вытворчасці і ў быце, могуць працаваць у сферы паслуг і забаў. Бываюць робаты, падобныя да чалавека, а бываюць зусім непадобныя.

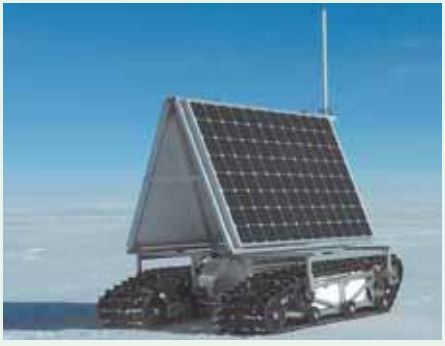
Робат — аўтаматычнае ўстройства, якое дзейнічае па загадзя складзенай праграме.

Робат атрымлівае інфармацыю пра навакольны свет ад датчыкаў — аналагаў органаў пачуццяў жывых арганізмаў — і прызначаны для ажыццяўлення розных аперацый.

Свет робатаў вельмі разнастайны. У быце сучаснага чалавека выкарыстоўваюцца аўтаматычныя пральныя і пасудамыечныя машыны, робаты-пыласосы і інш. З дапамогай робатаў можна вырошчваць расліны ці кіраваць домам.

Робат можа быць матэрыяльным або віртуальным. Віртуальны робат — спецыяльная праграма, якая выконвае пэўныя дзеянні.

Робат LS3, створаны для транспарціроўкі грузаў па перасечанай мясцовасці.



Аўтаномны робат GROVER, які вывучае пласты льду на ледніковым шчыце Грэнландыі.

Рабатызаваная сістэма, прызначаная для вырошчвання агародніны. Кіраванне дадзенай сістэмай ажыццяўляецца праз Wi-Fi. Ёсць магчымасць аддаленага кантролю праз Інтэрнэт.

Робат-пыласос з дапамогай сістэмы камер і сэнсараў можа арыентавацца ў пакоі і будаваць маршрут уборкі.

Робаты з'яўляюцца выканаўцамі. Для выканаўцаў звычайна вызначаюць асяроддзе існавання і сістэму каманд.

Агульным для ўсіх робатаў з'яўляецца тое, што чалавек можа імі кіраваць. Робат атрымлівае каманды ад аператара і выконвае іх па адной ці дзейнічае аўтаномна па папярэдне складзенай праграме.

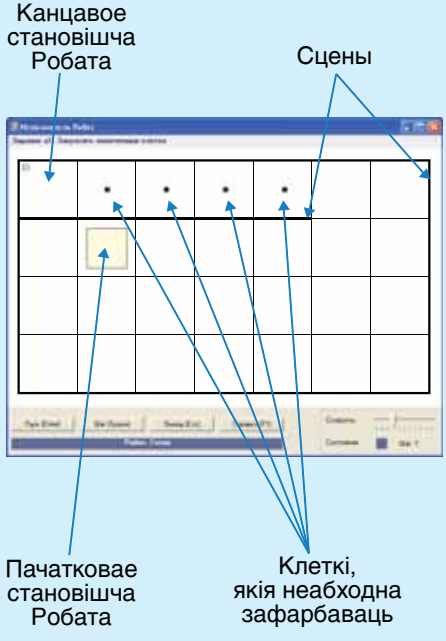
9.2. Асяроддзе існавання і сістэма каманд выканаўцы Робат

У асяроддзі праграмірования PascalABC, акрамя выканаўцы Чарцёжнік, можна выбраць выканаўцу Робат.

Асяроддзем існавання выканаўцы Робат з'яўляецца прамавугольнае клятчастае поле. Памеры поля, як і для выканаўцы Чарцёжнік, задаюцца камандай `Field(n, m)`. Першапачаткова Робат знаходзіцца ў цэнтральнай клетцы поля.

Паміж некаторымі клеткамі, а таксама па перыметры поля знаходзяцца сцены. Робат можа перамяшчацца па полі і зафарбоўваць паказаныя клеткі. Вялікі жоўты квадрат унутры клеткі азначае пачатковае становішча Робата, маленькі — канцавое (прыклад 9.1).

Прыклад 9.1. Поле выканаўцы Робат з пачатковым становішчам мае наступны выгляд:



Поле Робата, на якім вызначана знаходжанне сцен, пачатковае і канцавое размяшчэнне выканаўцы, называюць **становішчам.**

Для падключэння выканаўцы Робат у праграме прапісваецца каманда **uses** Robot. Гатовыя заданні са становішчамі для Робата захоўваюцца ў задачніку, убудаваным у сістэму праграмірования, і выклікаюцца камандай task. Гэта ж каманда выкарыстоўвалася для Чарцёжніка.

Сістэма каманд выканаўцы:

Каманда	Дзеянне
Right	Перамяшчае Робата ўправа
Left	Перамяшчае Робата ўлева
Up	Перамяшчае Робата ўверх
Down	Перамяшчае Робата ўніз
Paint	Зафарбоўвае дадзеную ячэйку

Робат можа станавіцца на звычайную і на зафарбаваную клетку, але не можа перамясціцца з клеткі на клетку, калі паміж імі сцяна. Робат не можа перамясціцца за межы поля. Гэтыя дзеянні выклікаюць памылку (прыклад 9.2). Робат можа зафарбаваць ужо зафарбаваную клетку. Такое дзеянне памылкі не выклікае.

Прыклад 9.2. Выклік задачы a1 з убудаванага задачніка:

Program1.pas

```
uses Robot;
begin
  Task('a1');
end.
```

Запіс каманд выканаўцы:

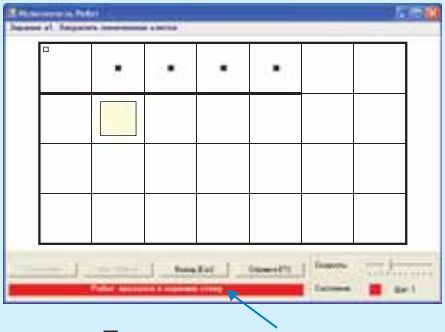
Program1.pas*

```
uses Robot;
begin
  Task('a1');
  right;
  right;
end.
```

Запішам у праграме каманду up.

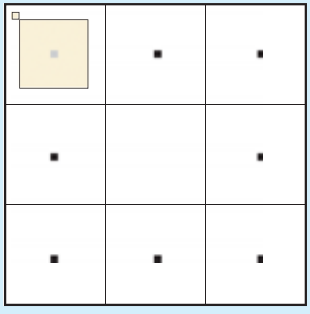
```
uses Robot;
begin
  Task('a1');
  up;
end.
```

Зверху знаходзіцца сцяна, таму перамяшчэнне Робата ўверх немагчымае:



Паведамленне пра памылку

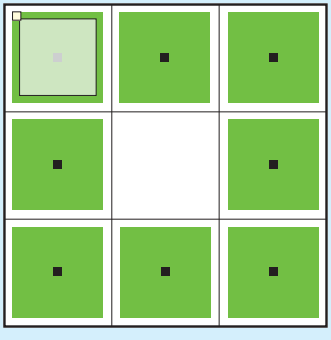
Прыклад 9.3. Пачатковае становішча:



Праграма для выканаўцы Робат:

```
uses Robot;
begin
  Task('a2');
  paint; right;
  paint; right;
  paint; down;
  paint; down;
  paint; left;
  paint; left;
  paint; up;
  paint; up;
end.
```

Вынік работы праграмы мае наступны выгляд:



9.3. Выкарыстанне алгарытмічнай канструкцыі *паслядоўнасць* для выканаўцы Робат

Разгледзім прыклады рашэння задач для выканаўцы Робат.

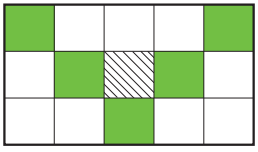
Прыклад 9.3. Робат знаходзіцца на полі памерам 3 × 3 клеткі. Трэба зафарбаваць усе клеткі, акрамя сярэдняй (задача a2).

Для рашэння задачы Робат павінен выканаць наступны алгарытм:

```
зафарбаваць;
управа;
зафарбаваць;
управа;
зафарбаваць;
уніз;
зафарбаваць;
уніз;
зафарбаваць;
улева;
зафарбаваць;
улева;
зафарбаваць;
уверх;
зафарбаваць;
уверх.
```

У дадзеным алгарытме Робат абыходзіць клеткі, рухаючыся па гадзіннікавай стрэлцы. Той жа вынік можна атрымаць, калі Робат будзе абыходзіць поле супраць гадзіннікавай стрэлкі, першапачаткова рухаючыся ўніз.

Прыклад 9.4. Складзём праграму для зафарбоўвання клетак поля Робата па ўзоры:



Такога становішча няма ў задачніку, таму спачатку трэба стварыць поле Робата памерам 5×3. Пачатковае становішча Робата на такім полі адзначана заштрыхаванай клеткай.

Для рашэння задачы Робат павінен выканаць алгарытм:

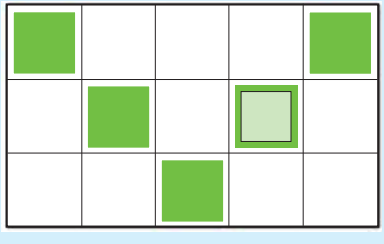
```
стварыць поле;
уніз;
зафарбаваць;
улева;
уверх;
зафарбаваць;
улева;
уверх;
зафарбаваць;
управа;
управа;
управа;
управа;
зафарбаваць;
улева;
уніз;
зафарбаваць.
```

**Якімі яшчэ спосабамі можна рашыць дадзеную задачу?*

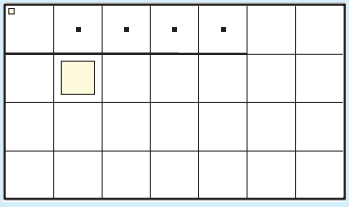
Прыклад 9.4. У дадзеным выпадку праграма для вучэбнага камп'ютарнага выканаўцы Робат можа быць складзена такім чынам:

```
uses Robot;
begin
  Field(5, 3);
  down;
  paint;
  left;
  up;
  paint;
  left;
  up;
  paint;
  right;
  right;
  right;
  right;
  paint;
  left;
  down;
  paint;
end.
```

Вынік работы запісанай вышэй праграмы будзе мець наступны выгляд:



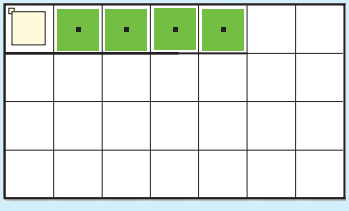
Прыклад 9.5. Пачатковае становішча:



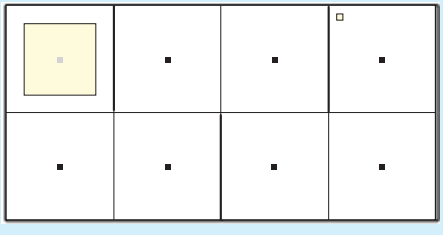
Праграма для Робата:

```
uses Robot;
begin
  Task('a1');
  right;  right;
  right;  right;
  up;
  left;  paint;
  left;  paint;
  left;  paint;
  left;  paint;
  left;
end.
```

Вынік работы праграмы:



Прыклад 9.6. Пачатковае становішча:



Прыклад 9.5. Рэшым задачу a1 з убудаванага задачніка.

Для рашэння дадзенай задачы Робат павінен абысці лінію (сцяну), зафарбаваць паказаныя клеткі і перамясціцца ў клетку, якая вызначае канцавое становішча выканаўцы.

Алгарытм рашэння задачы:

```
зрушыцца ўправа на 4 клеткі;
уверх;
зрушыцца ўлева на 4 клеткі,
зафарбоўваючы іх на шляху;
улева.
```

У прыкладах 9.3 — 9.5 каманды выканаўцы Робат выконваліся паслядоўна, адна за другой, у тым парадку, у якім яны былі запісаны. Таму ўсе прыведзеныя алгарытмы рэалізаваны з выкарыстаннем алгарытмічнай канструкцыі *паслядоўнасць*.

9.4. Дапаможныя алгарытмы

Прыклад 9.6. Рэшым задачу a3 з убудаванага задачніка.

Робат павінен зафарбаваць усе клеткі поля. Але рухацца па прамой яму перашкаджаюць сцены, якія выканаўца павінен абыходзіць.

Алгарытм рашэння задачы:

```
зафарбаваць; уніз;
зафарбаваць; управа;
```

```
зафарбаваць; уверх;
зафарбаваць;
управа;
зафарбаваць; уніз;
зафарбаваць; управа;
зафарбаваць; уверх;
зафарбаваць.
```

Калі прааналізаваць дадзены алгарытм, то можна заўважыць, што двойчы паўтараецца паслядоўнасць каманд, якая зафарбоўвае квадрат з чатырох клетак:

```
зафарбаваць; уніз;
зафарбаваць; управа;
зафарбаваць; уверх;
зафарбаваць.
```

Афармім гэтыя каманды як дапаможны алгарытм, які назавём `квадрат`. Тады алгарытм рашэння зыходнай задачы можа быць запісаны так:

```
квадрат;
управа;
квадрат.
```

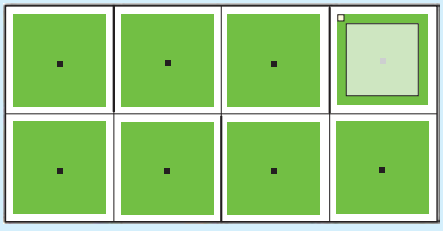
Пры рашэнні дадзенай задачы выкарыстанне дапаможнага алгарытму дазволіла не запісваць двойчы адну і тую ж паслядоўнасць каманд.

Дапаможныя алгарытмы можна выкарыстоўваць і ў тым выпадку, калі зыходная задача разбіваецца на некалькі незалежных адна ад адной задач. Тады

Праграма 1 для выканаўцы Робат:

```
uses Robot;
begin
  Task('a3');
  paint; down;
  paint; right;
  paint; up;
  paint;
  right;
  paint; down;
  paint; right;
  paint; up;
  paint;
end.
```

Вынік работы праграмы:



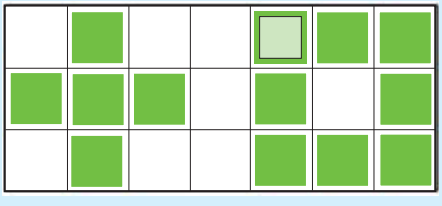
Праграма 2 (з выкарыстаннем дапаможнага алгарытму) для выканаўцы Робат:

```
uses Robot;
procedure kvadrat;
begin
  paint; down;
  paint; right;
  paint; up;
  paint;
end;
begin
  Task('a3');
  kvadrat;
  right;
  kvadrat;
end.
```

Прыклад 9.7. Праграма для вучэбнага камп'ютарнага выканаўцы Робат:

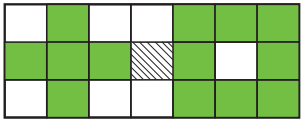
```
uses Robot;
procedure krest;
begin
  left; paint;
  down; left; paint;
  up; left; paint;
  right; paint;
  up; paint;
end;
procedure kvadrat;
begin
  paint; right;
  paint; right;
  paint; down;
  paint; down;
  paint; left;
  paint; left;
  paint; up;
  paint; up;
end;
begin
  field(7,3);
  krest;
  right; right; right;
  kvadrat;
end.
```

Вынік работы запісанай вышэй праграмы будзе мець наступны выгляд:



кожную з іх можна аформіць як дапаможны алгарытм.

Прыклад 9.7. Напішам праграму для зафарбоўвання клетак поля Робата па ўзоры:



Такога становішча няма ў задачніку, таму створым поле 7×3. Пачатковае становішча Робата адзначана заштрыхаванай клеткай.

У дадзенай задачы Робат павінен нарысаваць дзве асобныя фігуры: крыж і квадрат. Складзём два дапаможныя алгарытмы.

Дапаможны алгарытм крыж:

```
улева; зафарбаваць;
уніз; улева; зафарбаваць;
уверх; улева; зафарбаваць;
управа; зафарбаваць;
уверх; зафарбаваць.
```

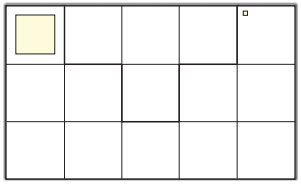
У якасці дапаможнага алгарытму для рысавання квадрата можна выкарыстоўваць алгарытм рашэння задачы а2 (прыклад 9.3). Для пераходу ад адной фігуры да іншай Робат павінен зрушыцца на 3 клеткі ўправа:

```
крыж;
управа; управа; управа;
квадрат.
```

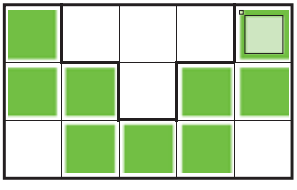
1. Што такое робат?
2. Якія каманды ўваходзяць у сістэму каманд вучэбнага камп'ютарнага выканаўцы Робат?
3. Апішыце асяроддзе існавання вучэбнага выканаўцы Робат.

Практыкаванні

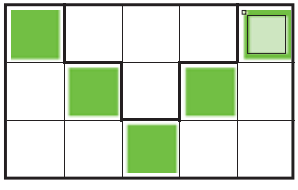
1 Пачатковае становішча на полі Робата адлюстравана на рысунку справа. Трое навучэнцаў склалі і выканалі алгарытм, па якім Робат зафарбаваў усе клеткі шляху ад пачатковай да канцавой. На якім з рысункаў — *а*, *б* або *в* — адлюстравана рашэнне дадзенай задачы? Чаму?



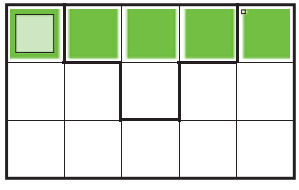
а



б



в



2 Які з прыведзеных алгарытмаў рашае задачу, сфармуляваную ў папярэднім заданні? Растлумачце, чаму іншыя праграмы не могуць быць алгарытмамі рашэння дадзенай задачы.

а)
```
paint;  down;
right;
paint;  down;
right;
paint;  right;
up;
paint;  right;
up;
paint;
```

б)
```
paint;  down;
paint;  right;
paint;  down;
paint;  right;
paint;  right;
paint;  up;
paint;  right;
paint;  up;
paint;
```

в)
```
ToPoint(0,3);
PenDown;
OnVector(1,0);
OnVector(0,-1);
OnVector(1,0);
OnVector(0,-1);
OnVector(1,0);
OnVector(0,1);
OnVector(1,0);
OnVector(0,1);
OnVector(1,0);
```

3 Для якога выканаўцы прыведзены алгарытм у заданні 2, *в*? Сфармулюйце для гэтага выканаўцы задачу, рашэннем якой будзе прыведзены алгарытм.

4 Для выканаўцы Робат была складзена наступная праграма:

```
paint;
right;  up;
paint;
right;  down;
```

Адлюструйце ў сшытку «ўзор», які нарысуе Робат. Пры якіх мінімальных памерах поля Робат зможа выканаць дадзеную праграму?

5 Усе каманды ў праграме з задання 4 навучэнец скапіраваў тры разы. Як зменіцца «ўзор» пасля выканання праграмы? Як можна па-іншаму запісаць гэты алгарытм? Якога памеру поле трэба стварыць? П а д к а з к а. Выкарыстайце дапаможны алгарытм.

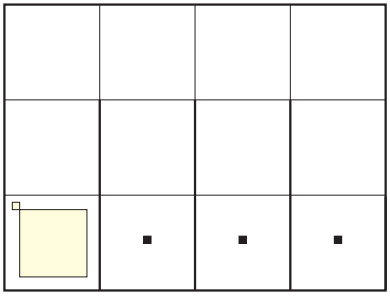
6 Праграма рашэння задачы была запісана на дошцы. Два навучэнцы, перапісваючы гэты алгарытм для выканаўцы Робат, прапусцілі з прычыны няўважлівасці па адной камандзе. Якую каманду прапусціў кожны з навучэнцаў? Што будзе вынікам работы кожнай праграмы?

Праграма, запісаная першым навучэнцам	Праграма, запісаная другім навучэнцам
uses Robot; **begin** Field(15,15); paint; right; paint; right; paint; down; paint; down; paint; down; left; down; left; paint; down; paint; down; paint; right; paint; right; paint; up; paint; up; paint; up; left; up; left; paint; up; paint; up; **end.**	**uses** Robot; **begin** Field(15,15); paint; right; paint; right; paint; down; paint; down; paint; down; left; paint; down; left; paint; down; paint; down; paint; right; paint; right; paint; up; paint; up; paint; up; left; up; paint; up; paint; up; **end.**

7 Складзіце праграму для рашэння задачы а4 з убудаванага задачніка (гл. рыс. справа). Прапануйце два алгарытмы:

1. З выкарыстаннем алгарытмічнай канструкцыі *паслядоўнасць*.
2. З выкарыстаннем дапаможнага алгарытму.

Параўнайце атрыманыя рашэнні.



8* Робат-агароднік можа разбіць градку на пасадачныя зоны-клеткі. На рысунку справа адлюстравана схема пасадкі агародніны (чырвоная клетка — таматы, зялёная — агуркі). Прапануйце сістэму каманд для робата-агародніка і распрацуйце алгарытм пасадкі агародніны (робат саджае адну расліну ў адну клетку).

§ 10. Алгарытмічная канструкцыя *паўтарэнне*

10.1. Алгарытмы з цыкламі

У навакольным свеце можна назіраць сітуацыі, пры якіх розныя дзеянні і працэсы паўтараюцца. Некаторыя паўтараюцца некалькі разоў і завяршаюцца. Іншыя могуць паўтарацца вельмі доўга (напрыклад, кругаварот вады ў прыродзе, рух планет у касмічнай прасторы, змена пор года і г. д.). Чалавеку таксама рэгулярна даводзіцца выконваць дзеянні, якія паўтараюцца: мыцца, апранацца, наведваць цырульню, снедаць, хадзіць на работу і інш.

Паняцце цыкла выкарыстоўваецца ў розных сферах чалавечай дзейнасці.

Пад цыклам разумеюць сукупнасць з'яў, працэсаў, якія складаюць кругаабарот на працягу пэўнага прамежку часу. З гэтага пункту гледжання можна гаварыць пра гадавы цыкл вярчэння Зямлі вакол Сонца ці пра вытворчы цыкл.

Цыклам з'яўляецца скончаны шэраг якіх-небудзь твораў; чаго-небудзь, што выкладаецца, выконваецца: цыкл лекцый, цыкл вершаў.

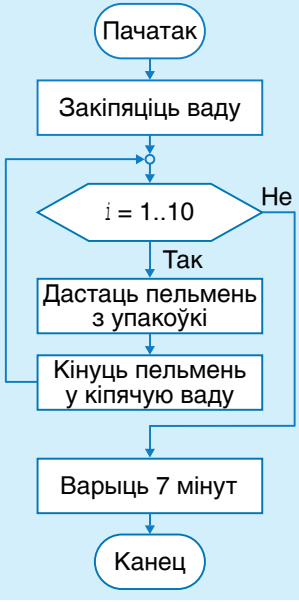
Прыклад 10.1. Гатаванне пельменяў.

Алгарытм:

1. Закіпяціць ваду.
2. Для i = 1..10 паўтараць:
   2.1. Дастаць пельмень з упакоўкі.
   2.2. Кінуць пельмень у кіпячую ваду.
3. Варыць 7 мінут.

У дадзеным прыкладзе параметр цыкла i змяняецца ад 1 да 10. Дзеянні «дастаць пельмень з упакоўкі» і «кінуць пельмень у кіпячую ваду» выконваюцца 10 разоў і складаюць цела цыкла.

Блок-схема дадзенага алгарытму выглядае такім чынам:



Як правіла, чалавек складае праграмы, у якіх кожная каманда паасобку і ўвесь алгарытм у цэлым выконваюцца за канчатковы лік паўтарэнняў.

Алгарытмічная канструкцыя *паўтарэнне (цыкл)* вызначае паслядоўнасць дзеянняў, якія выконваюцца шмат разоў. Гэтую паслядоўнасць дзеянняў называюць **целам цыкла.**

Існуе некалькі магчымасцей кіраваць тым, колькі разоў будзе паўтарацца цела цыкла.

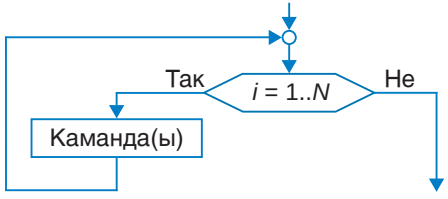
Алгарытмічная канструкцыя *цыкл з параметрам (цыкл з лічыльнікам)* — спосаб арганізацыі цыкла, пры якім колькасць паўтораў залежыць ад пачатковага і канчатковага значэнняў параметра цыкла.

Такім чынам, цыкл з параметрам арганізуе выкананне каманд цела цыкла загадзя вядомую колькасць разоў (прыклад 10.1).

Параметр цыкла вызначае нумарацыю дзеянняў у цыкле. Параметр цыкла можа прымаць толькі цэлыя значэнні. Часта нумарацыю пачынаюць з 1 і заканч-

ваюць лікам N (прыклад 10.2). У гэтым выпадку цыкл выканаецца N разоў. Калі нумарацыя вызначана двума адвольнымі лікамі $N1$ (пачатковае значэнне) і $N2$ (канчатковае значэнне), то цыкл выканаецца ($N2 - N1 + 1$) разоў.

Алгарытмічная канструкцыя цыкла з параметрам можа адлюстроўвацца на блок-схеме наступным чынам (значэнне параметра змяняецца ад 1 да N):



У дадзенай канструкцыі ў прамавугольніку(-ах) запісваюцца каманды алгарытму (цела цыкла), што паўтараюцца і выконваюцца N разоў (Так). Пры гэтым пасля кожнага выканання каманд цела цыкла адбываецца праверка, які раз выконваецца цыкл. На блок-схеме пераход на праверку ўмовы адлюстроўваецца ў выглядзе стрэлкі, што выходзіць з цела цыкла і вяртаецца да праверкі. Як толькі каманды цела цыкла выканаюцца N разоў (Не), цыкл завяршыцца (прыклад 10.3). Калі $N \leqslant 0$, то каманда цела цыкла не выканаецца ні разу.

Прыклад 10.2. Вылічым a^n (напрыклад, $3^5 = 243$).

Алгарытм узвядзення ліку ў ступень:

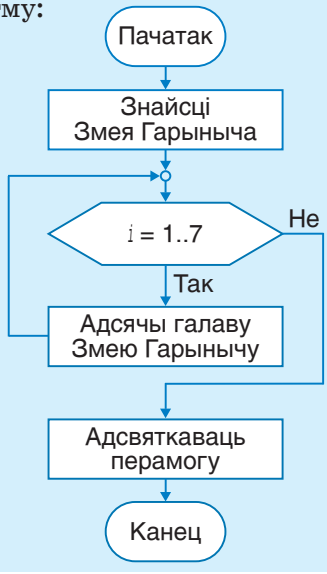
1. Увесці значэнні a і n.
2. Вызначыць пачатковае значэнне выніку $r = 1$.
3. Для $i = 1..n$ паўтараць:
   3.1. Памножыць вынік на a.
4. Запісаць вынік.

Прыклад 10.3. У фальклорных творах часта сустракаецца шматгаловы Змей Гарыныч (колькасць галоў можа быць, напрыклад, 7).

Алгарытм перамогі над Змеем Гарынычам:

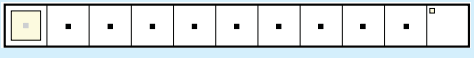
1. Знайсці Змея Гарыныча.
2. Для $i = 1..7$ паўтараць:
   2.1. Адсячы галаву Змею Гарынычу.
3. Адсвяткаваць перамогу.

Блок-схема дадзенага алгарытму:



Шмат якія робаты, што выкарыстоўваюцца ў быце і на вытворчасці, могуць выконваць цыклічныя алгарытмы. Прыкладам такога робата з'яўляецца сушы-робат, які можа вырабляць ад 450 да 4000 нарыхтовак для сушы за 1 гадзіну.

Прыклад 10.4. Пачатковае становішча:



Праграма для выканаўцы Робат:

```
uses Robot;
begin
  Task('c2');
  for var i:= 1 to 10 do
  begin
    paint;
    right;
  end;
end.
```

Вынік работы праграмы:

10.2. Выкарыстанне каманды цыкла з параметрам для выканаўцы Робат

Каб складаць алгарытмы з цыкламі для камп'ютарнага выканаўцы Робат, трэба ведаць, як запісваецца каманда цыкла.

Для запісу цыкла з параметрам выкарыстоўваецца каманда **for**. Фармат запісу каманды:

```
for var i:= N1 to N2 do[1]
begin
  цела цыкла;
end;
```

Радок **for var** i:= N1 **to** N2 **do** з'яўляецца загалоўкам цыкла. Яго чытаюць так: «Для пераменнай i ад $N1$ да $N2$ рабі». Калі $N2 \geqslant N1$, то каманды цела цыкла выконваюцца $(N2 - N1 + 1)$ разоў, інакш цыкл не выконваецца ні разу.

Словы **begin** і **end**; з'яўляюцца аператарнымі дужкамі ў мове Pascal. Калі цела цыкла складаецца з адной каманды, аператарныя дужкі можна прапусціць.

Аператарныя дужкі — пара слоў, якія вызначаюць у мове праграмірования блок каманд, што ўспрымаецца як адно цэлае, як адна каманда.

Прыклад 10.4. Рэшым задачу с2 з убудаванага задачніка.

[1] Каманда ў такім фармаце запісваецца толькі ў асяроддзі PascalABC.Net.

Робат павінен зафарбаваць усе клеткі поля (акрамя апошняй), перамяшчаючыся ўправа. Для гэтага ў цыкле трэба 10 разоў выканаць каманды:

```
зафарбаваць;
управа.
```

Дадзеныя каманды ўтвараюць цела цыкла.

Камандамі, што ўтвараюць цела цыкла, могуць быць любыя каманды з сістэмы каманд выканаўцы. Акрамя таго, у целе цыкла можа выклікацца дапаможны алгарытм. Выкарыстанне дапаможнага алгарытму дазволіць скараціць запіс цела цыкла і зробіць праграму больш зразумелай.

Прыклад 10.5. Рэшым задачу с7 з убудаванага задачніка.

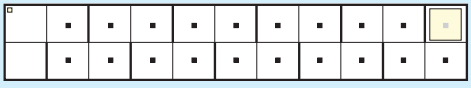
На полі выканаўцы Робат ёсць сцены. Пры абходзе сцен Робат выконвае наступныя каманды:

```
зафарбаваць;  уніз;
зафарбаваць;  улева;
зафарбаваць;  уверх;
зафарбаваць;  улева.
```

Каб рашыць задачу, Робат павінен паўтарыць гэтыя каманды 5 разоў. Афармім дадзеныя каманды як дапаможны алгарытм `kvadrat` і выклікаем яго ў цыкле.

У дадзеным прыкладзе цела цыкла складаецца з адной каманды `kvadrat`, таму аператарныя дужкі можна не выкарыстоўваць.

Прыклад 10.5. Пачатковае становішча для вучэбнага камп'ютарнага выканаўцы Робат:



Праграма для выканаўцы Робат складаецца наступным чынам:

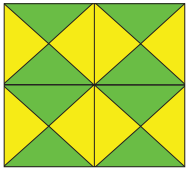
```
uses Robot;
procedure kvadrat;
begin
  paint;
  down;
  paint;
  left;
  paint;
  up;
  paint;
  left;
end;
begin
  Task('c7');
  for var i:= 1 to 5 do
    kvadrat;
end.
```

Вынік работы запісанай вышэй праграмы будзе мець наступны выгляд:

1. Што разумеюць пад алгарытмічнай канструкцыяй *паўтарэнне*?
2. Што такое цыкл з параметрам?
3. Што такое аператарныя дужкі?
4. Прывядзіце прыклады выкарыстання цыкла.

Практыкаванні

1 Апішыце слоўна або адлюструйце з дапамогай блок-схемы наступныя алгарытмы:

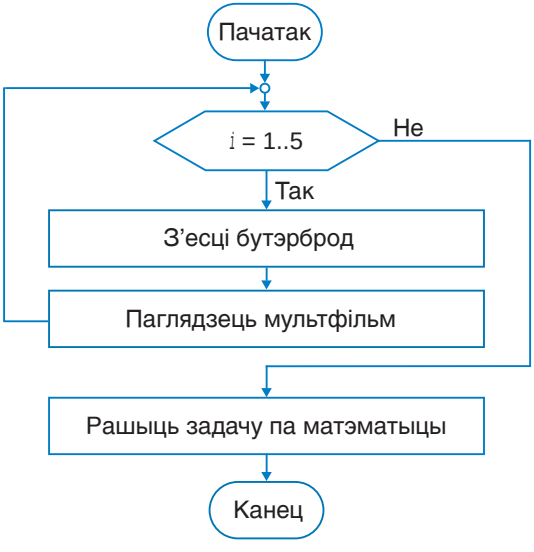


1. Рысаванне ў графічным рэдактары відарыса з 4 квадратаў з дыяганалямі і зафарбаванымі абласцямі (гл. рыс. справа).
2. Кожную мінуту бактэрыя дзеліцца на дзве. Першапачаткова ёсць адна бактэрыя. За бактэрыямі назіралі 10 мінут. Вызначыце колькасць бактэрый у канцы назірання. Запоўніце табліцу.

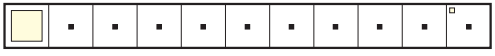
Час, мін	0	1	2	3	4	5	6	7	8	9	10
Колькасць бактэрый	1	2									

3. Свідраванне 10 адтулін.
4. Сервіроўка стала да абеду на 6 персон.

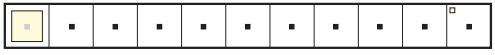
2 Сямікласнік Андрэй пасля школы запрасіў свайго сябра Юру дапамагчы яму ў рашэнні 5 задач па матэматыцы. У гасцях Юра параіў Андрэю правесці астатак дня, выкарыстаўшы алгарытм, запісаны ў выглядзе блок-схемы (гл. рыс. справа). Растлумачце, чаму за выкананне гэтага задання Андрэй атрымаў двойку па матэматыцы.



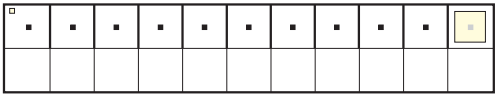
3 Складзіце праграму для рашэння задачы с3 з убудаванага задачніка. Параўнайце алгарытм рашэння гэтай задачы з прыкладам 10.4. Што ў іх агульнае? Чым яны адрозніваюцца?



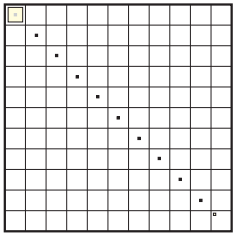
4 Складзіце праграму для рашэння задачы с4 з убудаванага задачніка. Параўнайце яе рашэнне з папярэднім практыкаваннем і з прыкладам 10.4.



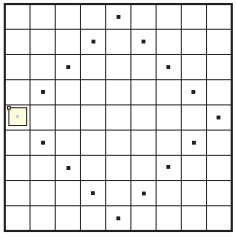
5 Складзіце праграму для рашэння задачы с8 з убудаванага задачніка. Выкарыстоўвайце дапаможны алгарытм.



6 Складзіце праграму для рашэння задачы с5 з убудаванага задачніка.



7 Для рашэння задачы с14 Пеця склаў алгарытм і запісаў праграму. Пецеў малодшы брат Алег выдаліў некалькі каманд. Вызначыце, колькі каманд выдаліў Алег. Аднавіце праграму, якую напісаў Пеця.



```
uses Robot;
begin
 Task('c14');
 paint;
 for var i:= 1 to 4 do
 begin
  paint;
  right;
  down;
 end;
 for var i:= 1 to 4 do
 begin
  right;
  up;
 end;
 for var i:= 1 to 4 do
 begin
  paint;
 end;
end.
```

8 Максім спрабуе ўявіць, як можна было б выкарыстоўваць робатаў у розных сітуацыях, апісаных у літаратурных творах. Напрыклад, для Тома Соера, якога цёця Полі адправіла фарбаваць плот, Максім прыдумаў робата-маляра і вырашыў, што такому робату дастаткова адной каманды: пафарбуй дошку. Алгарытм афарбоўкі плота з 20 дошак Максім запісаў так:

1. Устанавіць робата ля левага краю плота.
2. Для i = 1..20 паўтараць:
   2.1. Пафарбуй дошку.

Ці зможа робат-маляр пафарбаваць плот? У чым памылка Максіма? Выпраўце алгарытм, дадаўшы неабходную(-ыя) каманду(-ы).

§ 11. Выкарыстанне ўмоў

Умовы выкарыстоўваюцца ў правілах дарожнага руху. Так, калі гарыць зялёнае святло, то можна пераходзіць вуліцу.

Умовы таксама сустракаюцца ў фальклоры, напрыклад пры выбары шляху казачнымі героямі.

В. Васняцоў. «Віцязь на раздарожжы» (фрагмент). 1882 г.

11.1. Паняцце ўмовы

Прыняцце рашэнняў часта залежыць ад розных умоў. Калі на вуліцы дождж, то трэба ўзяць парасон; калі добра падрыхтаваўся да ўрока, то атрымаеш высокую адзнаку, інакш нізкую і інш.

Чалавек здольны разумець умовы, сфармуляваныя ў адвольнай форме. Але для таго каб Робат або іншы выканаўца мог прымаць рашэнні, трэба «навучыць» яго «разумець» умовы.

Умовай для выканаўцы з'яўляецца зразумелае яму выказванне, якое можа быць праўдзівым (выконвацца) або непраўдзівым (не выконвацца).

Выканаўца можа праверыць праўдзівасць умоў, што ўваходзяць у яго сістэму ўмоў.

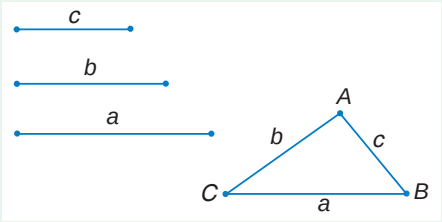
Разгледзім сістэму ўмоў для выканаўцы Робат.

`WallFromLeft`	Праўдзіва, калі злева ад Робата сцяна
`WallFromRight`	Праўдзіва, калі справа ад Робата сцяна
`WallFromUp`	Праўдзіва, калі зверху ад Робата сцяна
`WallFromDown`	Праўдзіва, калі знізу ад Робата сцяна
`FreeFromLeft`	Праўдзіва, калі злева ад Робата свабодна
`FreeFromRight`	Праўдзіва, калі справа ад Робата свабодна
`FreeFromUp`	Праўдзіва, калі зверху ад Робата свабодна
`FreeFromDown`	Праўдзіва, калі знізу ад Робата свабодна
`CellIsPainted`	Праўдзіва, калі ячэйка, у якой знаходзіцца Робат, зафарбавана
`CellIsFree`	Праўдзіва, калі ячэйка, у якой знаходзіцца Робат, не зафарбавана

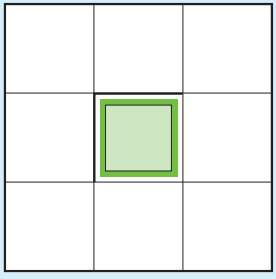
Узоры праўдзівых і непраўдзівых умоў для выканаўцы Робат паказаны ў прыкладзе 11.1.

Выкарыстоўваюцца ўмовы ў матэматыцы, напрыклад:

трохвугольнік існуе, калі для большай стараны a выконваецца няроўнасць $a < b + c$.



Прыклад 11.1. Разгледзім пачатковае становішча поля выканаўцы Робат:



У дадзеным выпадку для Робата будуць праўдзівымі наступныя ўмовы:

`WallFromLeft`
`WallFromUp`
`FreeFromRight`
`FreeFromDown`
`CellIsPainted`

Непраўдзівымі пры такім пачатковым становішчы будуць умовы:

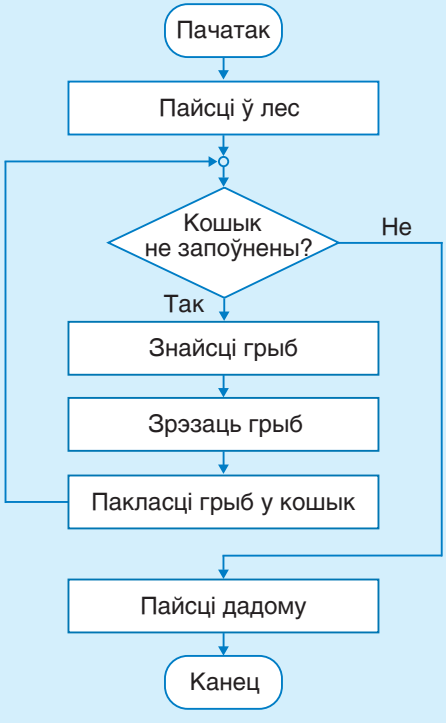
`WallFromRight`
`WallFromDown`
`FreeFromLeft`
`FreeFromUp`
`CellIsFree`

Прыклад 11.2. Збор грыбоў.

Выкарыстанне цыкла з параметрам пры складанні алгарытму рашэння гэтай задачы можа прывесці да розных вынікаў.

Кошык можа быць або напаўпустым, або не ўсе знойдзеныя грыбы ў яго змесцяцца.

Калі выкарыстоўваць цыкл з перадумовай, то ў выніку дамоў можна прынесці поўны кошык грыбоў.



11.2. Цыкл з перадумовай

Цыкл з параметрам выкарыстоўваецца пры складанні алгарытму ў тым выпадку, калі загадзя вядомая колькасць паўтарэнняў. Аднак часта да выканання цыкла колькасць паўтарэнняў невядомая.

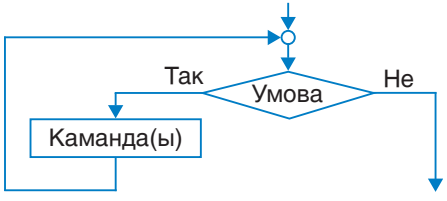
Прыклад 11.2. Вы з бацькамі пайшлі ў лес у грыбы. Вашы дзеянні можна апісаць камандамі: знайсці грыб, зрэзаць грыб, пакласці грыб у кошык. Гэтыя дзеянні будуць выконвацца ў цыкле, але вы загадзя не ведаеце, колькі грыбоў увойдзе ў кошык. Таму варта гаварыць не пра колькасць паўтарэнняў (колькасць грыбоў), а пра ўмову, пры якой вы будзеце працягваць збор грыбоў: пакуль кошык не запоўнены.

Алгарытмічная канструкцыя *цыкл з перадумовай* (*цыкл «пакуль»*) — спосаб арганізацыі цыкла, пры якім колькасць выкананняў каманд цела цыкла залежыць ад праўдзівасці або непраўдзівасці ўмовы цыкла.

Цыкл з перадумовай выкарыстоўваецца, калі колькасць паўтарэнняў цела цыкла загадзя невядомая, але вядомая ўмова працягвання працы.

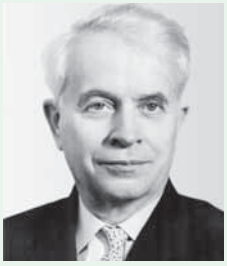
Умова цыкла вызначае, як доўга будзе выконвацца цыкл. Пакуль умова праўдзівая, выконваюцца каманды, што складаюць цела цыкла. Цыкл перастае выконвацца тады, калі ўмова становіцца непраўдзівай. Цыкл з перадумовай мае такую назву, паколькі праверка ўмовы папярэднічае выкананню каманд цела цыкла.

Алгарытмічная канструкцыя цыкла з перадумовай адлюстроўваецца на блок-схеме так:



У дадзенай канструкцыі ў прамавугольніку(-ах) запісваюцца каманды алгарытму (цела цыкла), якія паўтараюцца і здзяйсняюцца, пакуль правільная ўмова (Так). Пры гэтым пасля кожнага выканання каманд цела цыкла адбываецца праверка, ці праўдзівая ўмова. Як толькі ўмова стане непраўдзівай (Не), цыкл завяршаецца. Калі ўмова адразу непраўдзівая, то цыкл не выканаецца ні разу.

Калі ўмова ў цыкле будзе заўсёды праўдзівай (заўсёды Так), то такі цыкл не зможа завяршыцца. Такую сітуацыю называюць **зацыкліваннем**.



Расійскі акадэмік Андрэй Андрэевіч Маркаў (малодшы) (1903—1979) у сваіх даследаваннях у галіне тэорыі алгарытмаў паказаў, што ў агульным выпадку алгарытмы павінны змяшчаць прадпісанні двух відаў:

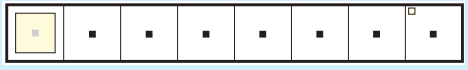
1) функцыянальныя аператары, накіраваныя непасрэдна на пераўтварэнне інфармацыі;

2) лагічныя аператары, якія вызначаюць далейшы напрамак дзеянняў.

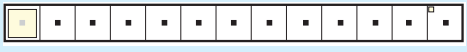
Аператар — элемент мовы, што задае поўнае апісанне дзеяння, якое неабходна выканаць. У англійскай мове дадзенае паняцце абазначаецца словам *statement*, што азначае таксама ‘выказванне’.

Калі ўжыць вышэйсказанае да камп’ютарных выканаўцаў, то прадпісанні першага віду складаюць сістэму каманд выканаўцы, а прадпісанні другога віду — сістэму ўмоў выканаўцы.

Прыклад 11.3. Адно з магчымых пачатковых становішчаў:



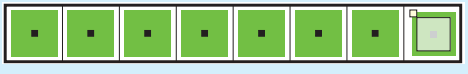
Другое магчымае пачатковае становішча:



Запішам праграму для вучэбнага камп'ютарнага выканаўцы Робат:

```
uses Robot;
begin
 Task('w2');
 while FreeFromRight do
 begin
  paint;
  right;
 end;
 paint;
end.
```

Вынік работы запісанай вышэй праграмы для першага пачатковага становішча будзе мець наступны выгляд:



Вынік работы праграмы для другога пачатковага становішча:

Для запісу цыкла з перадумовай выкарыстоўваецца каманда **while**. Фармат запісу каманды:

```
while <умова> do
begin
  цела цыкла;
end;
```

Радок **while** <умова> **do** з'яўляецца загалоўкам цыкла. Гэты радок можна прачытаць наступным чынам: «Пакуль правільная ўмова, рабі». Каманды **begin** і **end;** у дадзеным выпадку адыгрываюць ролю аператарных дужак.

Прыклад 11.3. Напішам праграму для рашэння задачы w2 з убудаванага задачніка.

Робат павінен зафарбаваць калідор пераменнай даўжыні.

У дадзенай задачы нам дакладна невядомая даўжыня калідора, але вядома, што Робат можа рухацца, пакуль справа пуста, і зафарбоўваць клеткі:

```
Пакуль справа пуста, паўтараць
  зафарбаваць;
  управа.
```

Пасля праходу ўсяго калідора Робат павінен зафарбаваць апошнюю клетку. Гэта адбываецца пасля выканання цыкла, бо для апошняй клеткі ўмова «справа пуста» ўжо не выконваецца.

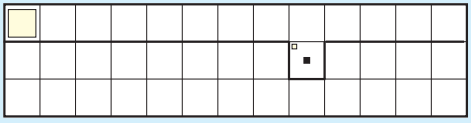
Прыклад 11.4. Напішам праграму для рашэння наступнай задачы. Робат знаходзіцца ў верхнім левым вугле поля. Знізу ад яго ўздоўж усяго поля размешчана сцяна з праходам у адну клетку. Скласці алгарытм, выканаўшы які Робат зможа прайсці праз праход і зафарбаваць клетку. Размяшчэнне праходу загадзя невядомае.

Праход не абмежаваны сцяной знізу. Робат можа рухацца ўправа, пакуль знізу ёсць сцяна:

```
Пакуль знізу сцяна, паўтараць
  управа.
```

Робат спыніцца ў той клетцы, у якой знізу няма сцяны. Пасля гэтага Робат павінен зрушыцца ўніз і зафарбаваць клетку.

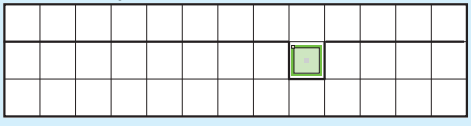
Прыклад 11.4. Адно з магчымых пачатковых становішчаў:



Праграма для выканаўцы Робат:

```
uses Robot, RobTasks[1];
begin
 Task('myrob3');
 while WallFromDown do
  right;
 down;
 paint;
end.
```

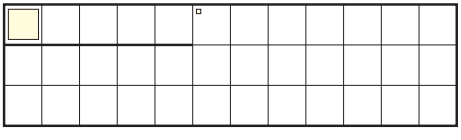
Вынік работы праграмы будзе наступным:

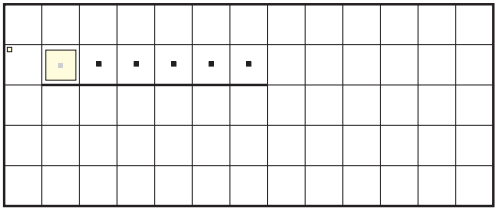


1. Што разумеюць пад умовай для выканаўцы?
2. Калі выкарыстоўваецца цыкл з перадумовай?
3. У якім выпадку ўзнікае сітуацыя зацыкліванне?

Практыкаванні

1 Напішыце праграму для рашэння задач w3 і w8 з убудаванага задачніка. Звяртайце ўвагу на пачатковае і канцавое становішча Робата.





[1] Модуль RobTasks, што змяшчае дадзенае становішча і задачу, можна спампаваць па адрасе: http://e-vedy.adu.by/course/view.php?id=423.

2 Для выканаўцы Робат быў напісаны наступны алгарытм:

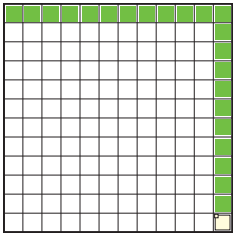
```
uses Robot;
begin
 Field( , );
 while FreeFromRight do
 begin
  paint;
  down;
  right;
  paint;
  up;
  right;
 end;
end.
```

Нарысуйце ў сшытку вынік работы дадзенага алгарытму. Якімі павінны быць памеры поля, каб Робат не ўрэзаўся ў сцяну? Вызначыце пачатковае становішча Робата.

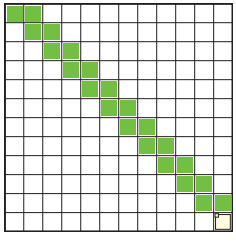
3 Складзіце алгарытм, выканаўшы які Робат нарысуе «ўзор» з задання 2 уздоўж левага краю поля выканаўцы. Якім павінен быць вертыкальны памер поля выканаўцы? (Задача myrob5 з модуля RobTasks.)

4 Робат знаходзіцца на квадратным полі невядомага памеру. Пачатковае становішча Робата — верхні левы вугал. Складзіце і выканайце алгарытм, па якім Робат перамесціцца з пачатковага становішча ў ніжні правы вугал і зафарбуе ўсе клеткі свайго шляху. На якім (на якіх) з рысункаў адлюстравана рашэнне гэтай задачы? Чаму?

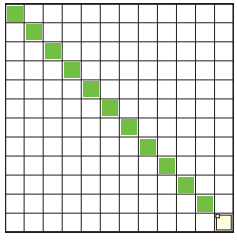
а



б

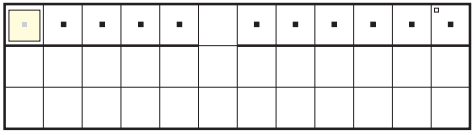


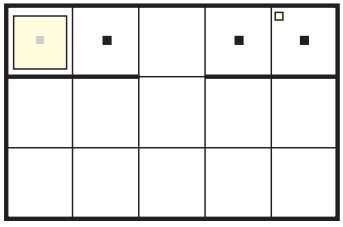
в



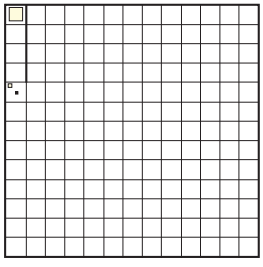
5 На полі Робата размешчаны «плот» — гарызантальная сцяна. Плот трэба «пафарбаваць» — зафарбаваць усе клеткі зверху сцяны. У «плоце» могуць быць адны «вароты» — клетка без ліній. Даўжыня «плота»

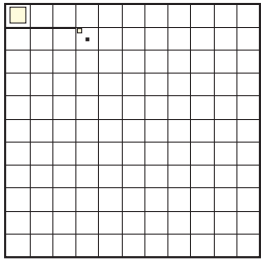
і размяшчэнне «варот» невядомыя. (Задача myrob7 з модуля RobTasks.)

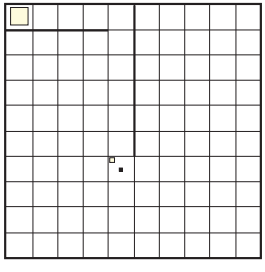


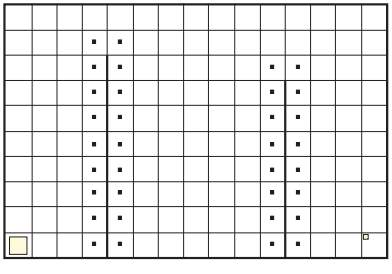


6 Па блок-схеме запішыце праграму для выканаўцы Робат. Якім будзе вынік для кожнага з прапанаваных пачатковых становішчаў? (Задача myrob8 з модуля RobTasks.)







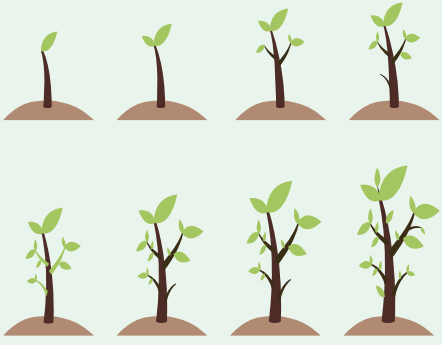


7* Рашыце задачу w10 з убудаванага задачніка. Напішыце дапаможны алгарытм для абходу адной сцяны.

§ 12. Алгарытмічная канструкцыя *галінаванне*

Паняцце галінавання выкарыстоўваецца ў розных сферах чалавечай дзейнасці.

У батаніцы пад галінаваннем парасткаў разумеюць працэс утварэння бакавых парасткаў у раслін.



Пры ўжыванні тэрміна ў пераносным сэнсе пад галінаваннем разумеюць наяўнасць некалькіх шляхоў, напрамкаў, сюжэтных ліній і г. д.

Галінаванні выкарыстоўваюцца ў дарожнай разметцы і картаграфіі.

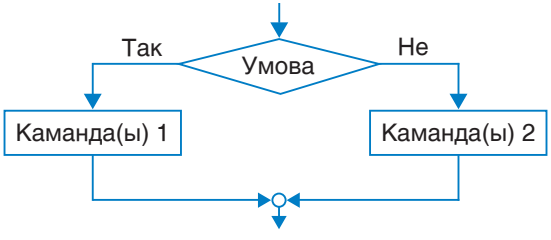
12.1. Каманда галінавання

Даволі часта на пастаўленае пытанне чалавек атрымлівае адказ «Так» або «Не». У залежнасці ад адказу ён вызначае свае дзеянні і выконвае адну ці другую каманду (групу каманд).

Робаты і іншыя тэхнічныя ўстройствы таксама могуць выконваць розныя дзеянні ў залежнасці ад умовы. Калі ўмова праўдзівая (на пытанне атрыманы адказ «Так»), то выконваюцца адны дзеянні, калі непраўдзівая, то іншыя.

Алгарытмічная канструкцыя *галінаванне* забяспечвае выкананне адной ці другой паслядоўнасці каманд у залежнасці ад праўдзівасці або непраўдзівасці некаторай умовы.

Галінаванне можа адлюстроўвацца на блок-схеме такім чынам:



У дадзенай канструкцыі ў прамавугольніку(-ах) запісваюцца каманды алгарытму. Пры такой

арганізацыі алгарытму можа выканацца **толькі** адна з дзвюх каманд (паслядоўнасцей каманд). Іншая паслядоўнасць будзе праігнаравана (прыклад 12.1).

Для запісу канструкцыі галінавання ў мове праграмірования Pascal выкарыстоўваецца каманда **if**. Фармат запісу каманды:

```
if <умова> then
begin
 каманды 1;
end
else
begin
 каманды 2;
end;
```

Радок **if** <умова> **then** з'яўляецца загалоўкам галінавання. Гэты радок можна прачытаць так: «Калі ўмова правільная, то». Пасля слова **then** запісваецца паслядоўнасць каманд 1, якая выканаецца, калі ўмова праўдзівая. Пасля слова **else** запісваецца паслядоўнасць каманд 2, якая выканаецца, калі ўмова непраўдзівая. Словы **begin** і **end**; у дадзеным выпадку адыгрываюць ролю аператарных дужак. Звярніце ўвагу, што перад словам **else** кропка з коскай не ставіцца.

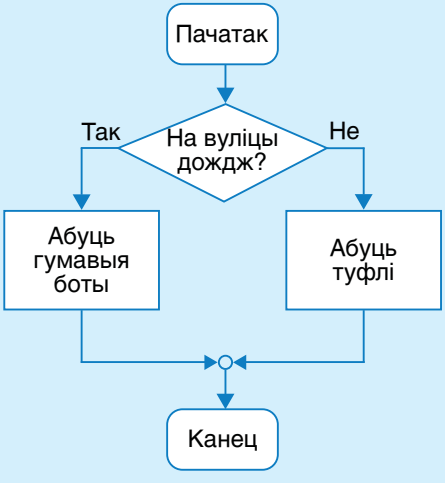
Галінаванне можа быць запісана ў **поўнай** або **скарочанай** форме.

Прыклад 12.1. Выбар абутку вясной у залежнасці ад надвор'я:

```
Калі на вуліцы дождж, то
  абуць гумавыя боты;
Інакш
  абуць туфлі.
```

У дадзеным прыкладзе ў цяперашні момант часу можа быць выканана толькі адна каманда з дзвюх: або абуць боты, або абуць туфлі.

Блок-схема дадзенага алгарытму будзе выглядаць наступным чынам:

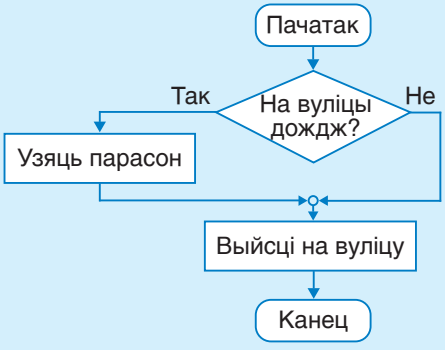


Прыклад 12.2. Выхад на вуліцу восенню.

```
Калі на вуліцы дождж, то
  ўзяць парасон;
выйсці на вуліцу.
```

Тут выкарыстоўваецца скарочаная форма каманды галінавання. Калі ўмова праўдзівая, выконваецца каманда «ўзяць парасон». Калі ўмова непраўдзівая, ніякіх дзеянняў не адбываецца. Каманда «выйсці на вуліцу» выконваецца заўсёды.

Блок-схема алгарытму:

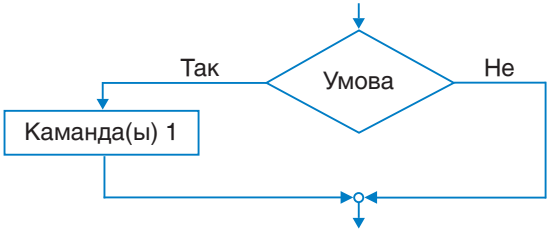


Прыклад 12.3. Маецца тры манеты, сярод якіх адна фальшывая. Фальшывая манета лягчэйшая за сапраўдныя. Знойдзем фальшывую манету за мінімальны лік узважванняў на шалях без гір:

```
Пакласці на кожную шалю манеты 1 і 2;
Калі шалі ўраўнаважаны, то
  фальшывая манета 3;
Інакш
 Калі манета 1 цяжэйшая, то
   фальшывая манета 2;
 Інакш
   фальшывая манета 1.
```

Поўная форма галінавання прадугледжвае арганізацыю выканання двух розных набораў каманд, з якіх выконваецца толькі адзін. У скарочанай форме адзін з набораў каманд (часцей па адказе «Не») прапускаецца. У гэтым выпадку, калі ўмова непраўдзівая, то ніякія дзеянні не выконваюцца.

Блок-схема скарочанай формы галінавання:



(Разгледзьце прыклад 12.2.)

На мове праграміравання Pascal каманда запішацца так:

```
if <умова> then
begin
 каманды 1;
end;
```

Алгарытм можа змяшчаць больш за адну канструкцыю галінавання (прыклад 12.3).

Прыклад 12.4. Рэшым задачу if1 з убудаванага задачніка.

Робат павінен зафарбаваць клетку, якая знаходзіцца за сцяной. У залежнасці ад становішча абход сцяны можа ажыццяўляцца па-рознаму.

Спачатку Робат павінен зрушыцца ўправа. Калі сцяна знізу, то зверху свабодна і можна абысці сцяну зверху, у адваротным выпадку Робат абыходзіць сцяну знізу.

Пасля абходу сцяны Робат зафарбоўвае клетку. Алгарытм можна запісаць так:

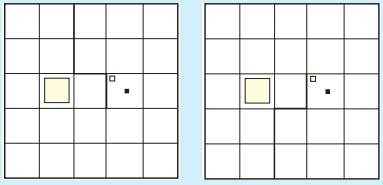
```
управа;
Калі зверху свабодна, то
  ўверх; управа; уніз;
Інакш
  уніз; управа; уверх;
зафарбаваць.
```

Прыклад 12.5. Робат знаходзіцца на невядомай клетцы поля без ліній. Ён павінен зафарбаваць клетку злева ад сябе.

Для таго каб зафарбаваць клетку злева ад сябе, Робат павінен перамясціцца ўлева, а затым зафарбаваць клетку. Аднак зрабіць гэта Робат зможа толькі тады, калі не знаходзіцца ў клетках, якія з'яўляюцца левай мяжой поля. Таму, перш чым зрушыцца ўлева, Робат павінен праверыць, ці свабодна злева.

Вынік работы дадзенай праграмы залежыць ад пачатковага становішча Робата. Таму для праверкі правільнасці работы праграмы неабходна падрыхтаваць пачатковыя становішчы, якія даюць розныя адказы на пытанне: злева пуста?

Прыклад 12.4. Магчымыя пачатковыя становішчы:



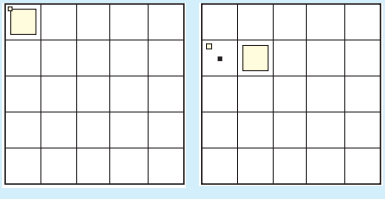
Праграма для Робата:

```
uses Robot;
begin
 Task('if1');
 right;
 if FreeFromUp then
 begin
  up; right; down;
 end
 else
 begin
  down; right; up;
 end;
 paint;
end.
```

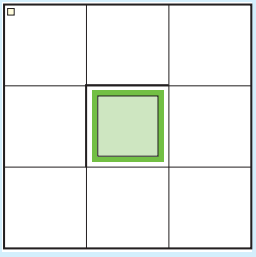
Прыклад 12.5. Праграма для выканаўцы Робат:

```
uses Robot, RobTasks;
begin
 Task('myrob9');
 if FreeFromLeft then
 begin
  left; paint;
 end;
end.
```

Магчымыя пачатковыя становішчы:



Прыклад 12.6. Разгледзім наступнае пачатковае становішча поля для выканаўцы Робат:



Праверым для Робата наступныя састаўныя ўмовы:

1. `WallFromLeft and CellIsPainted`.
2. `WallFromUp or WallFromDown`.
3. `Not (WallFromRight or FreeFromUp)`.

Першая ўмова складаецца з дзвюх простых: `WallFromLeft` (умова *A*) і `CellIsPainted` (умова *B*). Умова можа быць запісана як «*A* **І** *B*». Гэта ўмова правільная толькі тады, калі правільныя і *A*, і *B*. Умова *A* — `WallFromLeft` — праўдзівая, умова *B* — `CellIsPainted` — праўдзівая, умова *A* **І** *B* — праўдзівая.

Другая ўмова можа быць запісана як «*A* **АБО** *B*», дзе *A* — `WallFromUp`, *B* — `WallFromDown`. Умова *A* — праўдзівая, умова *B* — непраўдзівая. Значыць, умова «*A* **АБО** *B*» — праўдзівая.

12.2. Састаўныя ўмовы

У якасці ўмовы ў алгарытмах з цыкламі і галінаваннямі выкарыстоўваецца любое зразумелае выканаўцы гэтага алгарытму выказванне, якое можа быць або праўдзівым, або непраўдзівым.

Усе ўмовы, з якімі нам даводзілася дагэтуль сустракацца пры складанні алгарытмаў для Робата, былі простымі выказваннямі. Аднак можна будаваць і састаўныя ўмовы.

Састаўная ўмова — умова, якая ўтвараецца з некалькіх простых умоў, злучаных адна з адной лагічнымі аперацыямі.

З лагічнымі аперацыямі над выказваннямі вы ўжо знаёмыя. У PascalABC выкарыстоўваюцца наступныя лагічныя аперацыі:

Лагічная аперацыя	Запіс у PascalABC
Не	`Not`
І	`And`
Або	`Or`

(Разгледзьце прыклад 12.6.)

Сістэма ўмоў для Робата пабудавана так, што можна абысціся без выкарыстання лагічнай аперацыі адмаўлення.

Для ўмовы FreeFromLeft адмаўленнем будзе ўмова not

FreeFromLeft. Але ўмова «злева не свабодна» азначае, што там сцяна. Таму замест умовы not FreeFromLeft можна пісаць WallFromLeft. Адмаўленні іншых умоў паказаны ў табліцы:

Умова	Адмаўленне
`WallFromLeft`	`FreeFromLeft`
`WallFromRight`	`FreeFromRight`
`WallFromUp`	`FreeFromUp`
`WallFromDown`	`FreeFromDown`
`CellIsPainted`	`CellIsFree`

У трэцяй умове часціца not адмаўляе састаўную ўмову `WallFromRight or FreeFromUp`. Умова можа быць запісана як **НЕ** (*A* **АБО** *B*). Для таго каб вызначыць, праўдзівая ці непраўдзівая гэта ўмова, трэба спачатку вызначыць праўдзівасць умовы «*A* **АБО** *B*». Умова *A* — непраўдзівая, умова *B* таксама непраўдзівая. Таму непраўдзівай будзе і ўмова «*A* **АБО** *B*», але тады ўмова **НЕ** (*A* **АБО** *B*) будзе праўдзівай.

1. Што такое алгарытмічная канструкцыя *галінаванне*?
2. Чым адрозніваецца поўная канструкцыя галінавання ад скарочанай?
3. Што такое састаўная ўмова?
4. Якія лагічныя аперацыі можна выкарыстоўваць для запісу састаўных умоў?
5. Якімі спосабамі можна пабудаваць адмаўленне ўмовы для камп'ютарнага выканаўцы Робат?

Практыкаванні

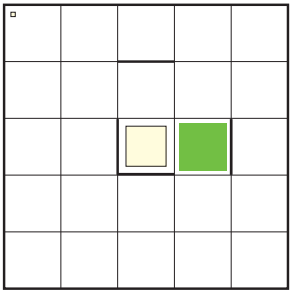
1 Вылучыце канструкцыю галінавання ва ўрыўку з паэмы А. С. Пушкіна «Руслан і Людміла» і адлюструйце яе з дапамогай блок-схемы.

У лукоморья дуб зеленый;
Златая цепь на дубе том:
И днем и ночью кот ученый
Все ходит по цепи кругом;
Идет направо — песнь заводит,
Налево — сказку говорит.
Там чудеса: там леший бродит,
Русалка на ветвях сидит...[1]

[1] Пушкин, А. С. Руслан и Людмила : поэма. — М. : Изд. Дом «Прибой». — 1996. — С. 5.

2 Для зададзенага становішча поля Робата вызначыце, якія з састаўных умоў праўдзівыя, а якія непраўдзівыя.

Пачатковае становішча	Умовы

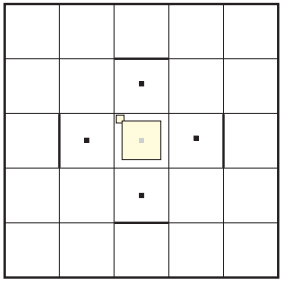
	`WallFromLeft or CellIsPainted;`
	`WallFromUp and WallFromDown;`
	`Not CellIsPainted and FreeFromRight;`
	`Not (WallFromUp or FreeFromRight);`
	`WallFromDown and CellIsFree;`
	`(WallFromUp or WallFromDown) and FreeFromRight.`

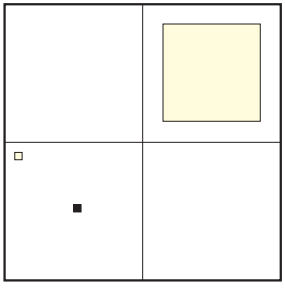
3* У заданні 2 замяніце ўмовы, якія змяшчаюць not, адпаведнымі ўмовамі без выкарыстання адмаўлення.

4 Для кожнай з непраўдзівых умоў задання 2 прыдумайце становішча, у якім дадзеная ўмова будзе праўдзівай, а для кожнай праўдзівай — становішча, у якім умова будзе непраўдзівай.

5 Змяніце праграму з прыкладу 12.5 так, каб Робат зафарбоўваў клетку справа (знізу, зверху) ад сябе. Нарысуйце ў сшытку розныя пачатковыя становішчы для праверкі дадзенай умовы.

6 Рашыце задачы if2 і if3 з убудаванага задачніка.





§ 13. Выкарыстанне асноўных алгарытмічных канструкцый для выканаўцы Робат

Паслядоўнае выкананне каманд у праграме вызначаецца структурай ***паслядоўнасць***. Для арганізацыі дзеянняў, якія паўтараюцца, у алгарытме выкарыстоўваецца каманда **цыкла**. Каманда **галінавання** дазваляе выконваць адну ці другую паслядоўнасць каманд у залежнасці ад праўдзівасці ўмовы.

Паслядоўнасць, цыкл і галінаванне — **базавыя алгарытмічныя канструкцыі**. Выкарыстоўваючы гэтыя канструкцыі як элементы нейкага «канструктара», можна складаць і распрацоўваць любыя алгарытмы.

Каманды цыкла і галінавання кіруюць парадкам выканання іншых каманд у праграме і належаць да каманд кіравання. Выкарыстанне алгарытмічнай канструкцыі *паслядоўнасць* прадугледжвае адсутнасць кіруючых канструкцый.

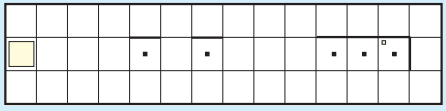
Разгледзім падрабязней прыклады алгарытмаў, якія змяшчаюць некалькі алгарытмічных канструкцый.

Перад чалавекам увесь час узнікаюць разнастайныя задачы, для якіх існуюць розныя алгарытмы рашэння. Пры ўсёй разнастайнасці алгарытмаў для іх запісу дастаткова трох алгарытмічных канструкцый (структур): *паслядоўнасць, цыкл, галінаванне*.

Гэта палажэнне было высунута ў сярэдзіне 70-х гг. XX ст. нідэрландскім вучоным Эдсгерам Вібе Дэйкстрам (1930—2002).

Яго працы зрабілі ўплыў на развіццё інфарматыкі і інфармацыйных тэхналогій. Э. Дэйкстра з'яўляецца адным з распрацоўшчыкаў канцэпцыі структурнага праграмірования, удзельнічаў у стварэнні мовы праграмірования Алгол. Вядомы сваімі дасягненнямі ў галіне матэматычнай логікі і тэорыі графаў.

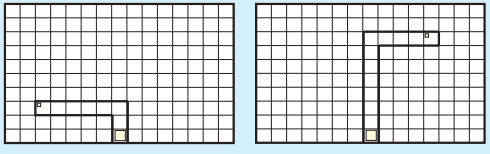
Прыклад 13.1. Адно з магчымых пачатковых становішчаў:



Праграма для Робата:

```
uses Robot;
begin
 Task('cif1');
 while FreeFromRight do
 begin
  if WallFromUp then
   paint;
  right;
 end;
 if WallFromUp then
   paint;
end.
```

Прыклад 13.2. Магчымыя пачатковыя становішчы:



Праграма для Робата:

```
uses Robot;
begin
 Task('cif17');
 while FreeFromUp do
  up;
 if FreeFromLeft then
  while FreeFromLeft do
   left
 else
  while FreeFromRight do
   right;
end.
```

Прыклад 13.1. Рэшым задачу cif1 з убудаванага задачніка.

Робат перамяшчаецца ўправа датуль, пакуль не сустрэне сцяну. На шляху ён павінен зафарбаваць клеткі, над якімі ёсць сцяна.

Для рашэння задачы Робат павінен правяраць кожную клетку на сваім шляху. Калі ўмова «зверху сцяна» выконваецца, Робат зафарбоўвае гэту клетку. Пасля праверкі клеткі Робат зрушваецца ўправа. Такія дзеянні выконваюцца ў цыкле, пакуль справа пуста.

Пасля цыкла патрэбна каманда галінавання, бо для крайняй клеткі поля каманда «справа пуста» не выконваецца і клетка ў цыкле не зафарбоўваецца. У гэтай задачы ўнутры структуры цыкла выкарыстоўваецца структура галінавання.

Прыклад 13.2. Рэшым задачу cif17 з убудаванага задачніка.

Робат павінен дайсці да канца калідора пераменнага памеру. Калідор можа зварочваць улева або ўправа.

Для рашэння задачы Робат спачатку перамяшчаецца ўверх датуль, пакуль зверху пуста. Сцяна, якая з'явілася зверху, азначае, што пачаўся паварот калідора. Калідор паварочвае ўлева, калі злева пуста, інакш калідор паварочвае ўправа. Далей Робат руха-

ецца ў тым напрамку, дзе пуста, пакуль не сустрэне сцяну.

У дадзенай задачы выкарыстоўваецца спачатку структура цыкла, а затым структура галінавання. Кожная паслядоўнасць каманд у структуры галінавання, у сваю чаргу, з'яўляецца цыклам.

Аператарныя дужкі прапушчаны, паколькі паслядоўнасць складаецца з адной каманды цыкла.

Прыклад 13.3*. Рэшым задачу cc5 з убудаванага задачніка.

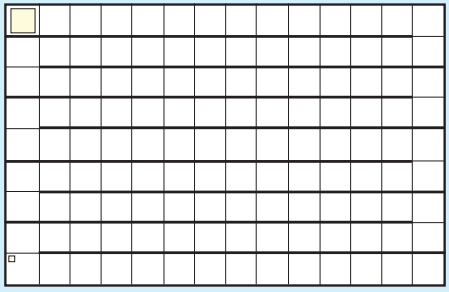
Робат знаходзіцца ў верхнім левым вугле поля і павінен перамясціцца ў ніжні левы вугал. На полі прысутнічаюць сцены, якія Робат павінен абысці. Пры гэтым ён павінен спачатку рухацца да правай мяжы поля, затым спусціцца ўніз, а потым рухацца да левай мяжы поля і спусціцца ўніз. Гэтыя дзеянні Робат павінен паўтарыць 4 разы.

У дадзенай задачы ўнутры цыкла з параметрам выкарыстоўваюцца два іншыя цыклы з перадумовай.

Структуру, калі ўнутры аднаго цыкла выконваецца іншы, называюць **укладзенымі цыкламі**.

Такім чынам, базавыя алгарытмічныя канструкцыі можна камбінаваць адну з адной.

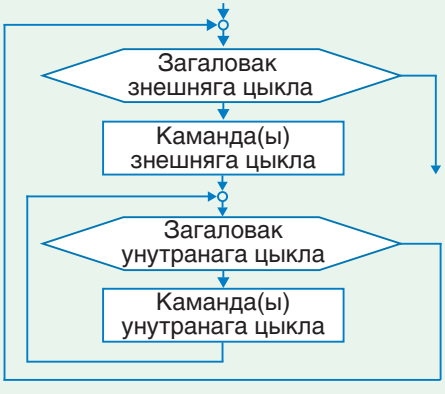
Прыклад 13.3*. Пачатковае становішча:



Праграма для Робата:

```
uses Robot;
begin
 Task('cc5');
 for var i:= 1 to 4 do
 begin
  while FreeFromRight do
   right;
  down;
  while FreeFromLeft do
   left;
  down;
 end;
end.
```

Блок-схема ўкладзеных цыклаў:



1. Назавіце базавыя алгарытмічныя канструкцыі.
2. Прывядзіце прыклады выкарыстання базавых алгарытмічных канструкцый.
3. Што такое ўкладзены цыкл?

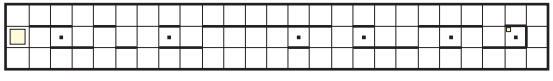
Практыкаванні

1 Якія алгарытмічныя канструкцыі выкарыстоўваюцца ў прыведзеных праграмах? Нарысуйце блок-схемы дадзеных алгарытмаў. Прапануйце прыклад пачатковага становішча, у якім алгарытм выканаецца карэктна.

а)
```
uses Robot;
begin
 while WallFromLeft do
 begin
  down;
  paint;
 end;
end.
```

б)
```
uses Robot;
begin
 while CellIsPainted do
  if FreeFromleft then
   left;
end.
```

2 Для рашэння задачы cif3 з убудаванага задачніка Міша напісаў праграму, але яна працуе няправільна. Якія памылкі дапусціў Міша?



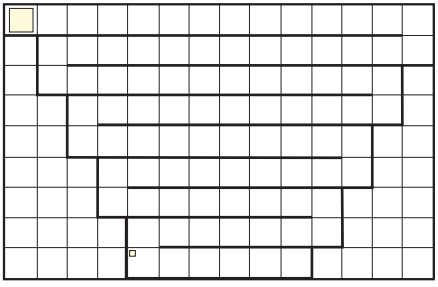
```
uses Robot;
begin
 Task('cif3');
 while WallFromRight do
 begin
  if WallFromDown or
  WallFromUp then
   paint;
  right;
 end;
 if WallFromUp and
 WallFromDown then
  paint;
end.
```

3 Выкарыстоўваючы базавыя алгарытмічныя канструкцыі, запішыце алгарытмы, якія адпавядаюць апісанням. Пабудуйце для іх блок-схемы.

1. Цела цыкла, што выконваецца пры ўмове `WallFromUp`, складаецца з дзвюх каманд: `right` і `paint`.
2. Калі ўмова `FreeFromRight` не выконваецца, то, калі клетка не зафарбавана, яе трэба зафарбаваць, а калі зафарбавана, то зрушыцца ўлева.

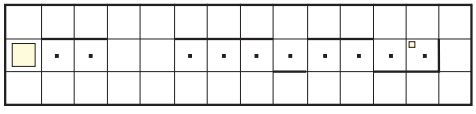
3. Праверку ўмовы `CellIsPainted` трэба выконваць датуль, пакуль знізу няма сцен. Пры выкананні ўмовы зрушыцца ўніз, пры невыкананні ўмовы зафарбаваць клетку.

4 Запоўніце пропускі ў праграме рашэння задачы cc14 з убудаванага задачніка так, каб яна працавала правільна.

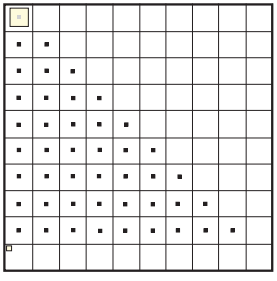


```
uses Robot;
begin
 Task('cc14');
 for var i: = 1 to 4 do
 begin
  while … do
   right;
  down;
  while … do
   left;
  down;
 end;
end.
```

5 Рашыце задачу cif2 з убудаванага задачніка, выкарыстоўваючы ўнутры цыкла каманду галінавання.



6 Рашыце задачу cc7 з убудаванага задачніка, выкарыстоўваючы ўнутры аднаго цыкла два ўкладзеныя цыклы.



7* Прыдумайце задачу для выканаўцы Робат, у якой будуць выкарыстоўвацца розныя алгарытмічныя канструкцыі.

§ 14. Мова праграмірования Паскаль

Камп'ютар (ад англ. *computer* — вылічальнік) — устройства або сістэма, здольныя выконваць зададзеную дакладна пэўную змяняемую паслядоўнасць аперацый (часцей за ўсё лікавых разлікаў).

Электронна-вылічальная машына (ЭВМ) — комплекс тэхнічных сродкаў, дзе асноўныя функцыянальныя элементы (лагічныя; якія запамінаюць; індыкацыйныя і інш.) выкананы на электронных прыборах, прызначаных для аўтаматычнай апрацоўкі інфармацыі ў працэсе рашэння вылічальных задач.

Ніклаус Вірт (нарадзіўся ў 1934 г.) — швейцарскі вучоны, спецыяліст па інфарматыцы, адзін з самых вядомых тэарэтыкаў у галіне распрацоўкі моў праграмірования, прафесар камп'ютарных навук. Стваральнік і вядучы праекціроўшчык моў праграмірования Паскаль, Модула-2, Аберон.

Жаданне спрасціць і паскорыць разнастайныя разлікі ўласціва чалавеку са старажытных часоў. Сёння камп'ютар здольны выконваць сотні мільёнаў аперацый у секунду. Для рашэння вылічальных задач патрабуецца спачатку скласці алгарытм іх рашэння, а затым запісаць яго ў выглядзе праграмы, выкарыстоўваючы якую-небудзь мову праграмірования.

Мова праграмірования вызначае набор правіл, якія акрэсліваюць знешні выгляд праграмы і дзеянні, што ажыццявіць выканаўца пад яе кіраваннем.

Мова праграмірования Паскаль (Pascal) выкарыстоўваецца для навучання праграмірованню і з'яўляецца базай для шэрага прафесійных моў праграмірования.

Існуе мноства асяроддзяў праграмірования, якія падтрымліваюць мову Паскаль: PascalABC, FreePascal, Delphi, GNU Pascal, Dev-Pascal, Rad Studio і інш. У вучэбным курсе выкарыстоўваецца асяроддзе PascalABC (з ім вы працавалі, знаёмячыся з вучэбнымі камп'ютарнымі выканаўцамі).

14.1. Каманда вываду

Дэманстраваць работу любой праграмы мае сэнс толькі тады, калі яна выводзіць якую-небудзь інфармацыю.

Праграма на мове Pascal (цела праграмы) павінна пачынацца са слова **begin**, а заканчвацца словам **end** і кропкай. Праграма, што складаецца з гэтых каманд, раздзеленых прабелам або пераводам радка, можа быць запушчана на выкананне, але яна нічога не робіць. Дабавім у яе каманду вываду прывітання:

```
begin
  write('Прывітанне!');
end.
```

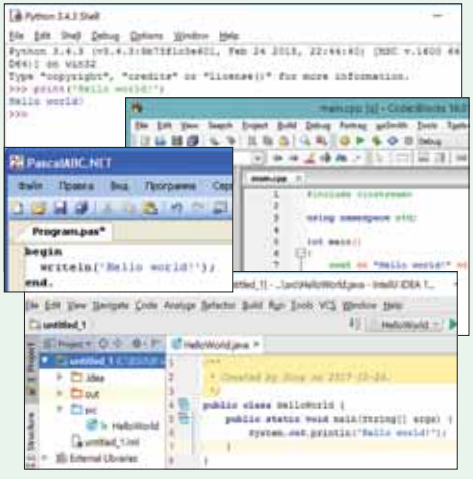
Вынік работы праграмы адлюстроўваецца ў ніжняй частцы акна праграмы PascalABC у акне вываду (прыклад 14.1).

Каманда `write( );` прызначана для **вываду даных.**

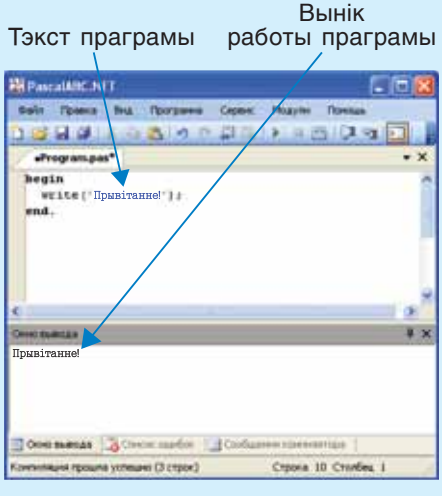
Тэкст, які трэба вывесці на экран, змяшчаюць у апострафах (адзінарнае двукоссе). Гэты тэкст не аналізуецца і выводзіцца ў тым выглядзе, у якім ён запісаны. Тэкст можна запісаць на любой мове. Тэкстам можа быць адвольны набор сімвалаў.

У адной праграме можа быць некалькі каманд вываду. Для

Па традыцыі, якая пачалася ў 1978 г. з прыкладу ў кнізе Б. Кернігана і Д. Рытчы «Мова праграміравання Сі», першая праграма на любой мове праграміравання павінна выводзіць на экран прывітанне свету:



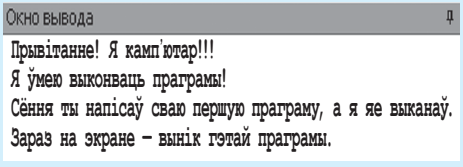
Прыклад 14.1. Акно асяроддзя PascalABC з вынікам работы праграмы:



Прыклад 14.2. Тэкст праграмы:

```
begin
  write('Прывітанне!  ');
  writeln('Я  камп'ютар!!!');
  write('Я ўмею выконваць ');
  writeln('праграмы!');
  write('Сёння  ты  ');
  write('напісаў  сваю  ');
  write('першую праграму,');
  writeln(' а я яе выканаў.');
  write('Зараз  на  экране  –');
  writeln('  вынік  гэтай
праграмы.');
end.
```

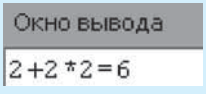
Вынік работы праграмы:



Прыклад 14.3. Тэкст праграмы:

```
begin
  write('2+2*2=');
  write(2+2*2);
end.
```

Вынік работы праграмы:



Дзве каманды `write` ў праграме можна аб'яднаць у адну, аддзяліўшы тэкст ад выразу коскай:

```
begin
  write('2+2*2=',  2+2*2);
end.
```

выводу тэксту, запісанага ў некалькі радкоў, выкарыстоўваюць каманду `writeln( )`. Спалучэнне «`ln`» (скар. ад англ. *line* — лінія, радок), запісанае ў канцы каманды, азначае, што пасля выводу трэба перавесці курсор у новы радок.

Прыклад 14.2. Выведзем на экран камп'ютара наступны тэкст: «`Прывітанне! Я камп'ютар!!! Я ўмею выконваць праграмы! Сёння ты напісаў сваю першую праграму, а я яе выканаў. Зараз на экране – вынік гэтай праграмы`».

Выкарыстоўваючы спалучэнне каманд `write` і `writeln`, тэкст можна размясціць па-рознаму.

Як вы ўжо ведаеце, тэкст у камандзе `write( )`, запісаны ў двукоссі, не аналізуецца. Калі двукоссе прапусціць, то выконваецца аналіз даных, запісаных у дужках. Так, калі ў дужках напісаць арыфметычны выраз, то спачатку вылічваецца яго значэнне, а затым выводзіцца вынік.

Прыклад 14.3. Вылічым значэнне выразу 2 + 2 * 2.

Калі запісаць выраз у двукоссі, то будзе выведзены сам выраз. Пры адсутнасці двукосся на экран будзе выведзена значэнне дадзенага выразу.

14.2. Паняцце тыпу даных

На практыцы рэдка даводзіцца пісаць праграмы, якія рашаюць толькі адну задачу. Звычайна праграмы пішуцца для рашэння цэлага класа задач, што можна сфармуляваць у агульным выглядзе.

З такімі задачамі вы ўжо сустракаліся ў курсе матэматыкі. Напрыклад, рашэнне задачы «Знайдзіце плошчу прамавугольніка» можна запісаць так: $S = a \cdot b$, дзе пераменныя a і b абазначаюць адпаведна даўжыню і шырыню прамавугольніка, а S — плошчу. Ведаючы гэту формулу, можна знайсці плошчу любога прамавугольніка.

У праграмiраванні для рашэння задач у агульным выглядзе таксама выкарыстоўваюць пераменныя. Паколькі з такімі пераменнымі будзе працаваць камп'ютар, то яны павінны захоўвацца ў яго памяці.

Інфармацыю, пададзеную ў прыдатным для апрацоўкі на камп'ютары выглядзе, называюць **данымі**.

Пераменная ў праграмiраванні — гэта найменная ячэйка памяці, якая захоўвае значэнне пераменнай.

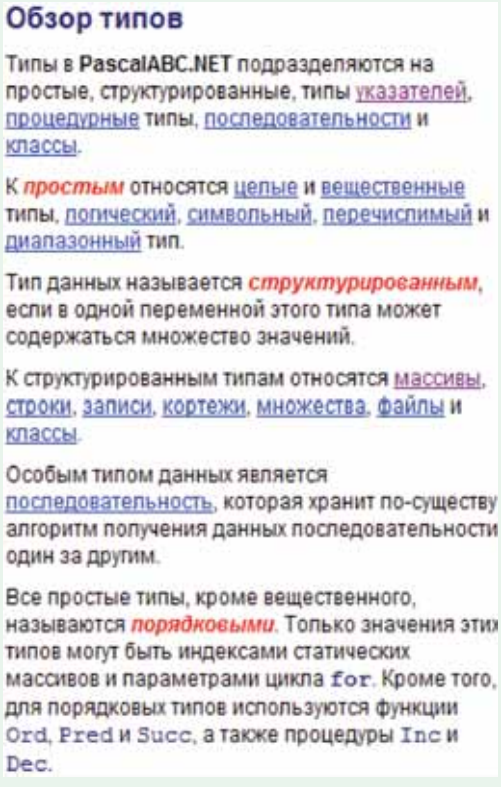
Да пачатку 1950-х гг. XX ст. праграмісты ЭВМ пры стварэнні праграм карысталіся машынным кодам. Запіс праграмы на машынным кодзе складаўся з адзінак і нулёў. Машынны код прынята лічыць мовай праграмiравання першага пакалення. Тыпы даных не выкарыстоўваліся.

Першай мовай праграмiравання, у якой з'явілася магчымасць ствараць пераменныя, лічыцца Асэмблер. У гэтай мове замест машынных кодаў сталі выкарыстоўваць каманды, запісаныя тэкстам. Асэмблер належыць да моў праграмiравання другога пакалення.

У 1957 г. з'явілася мова Фартран, якая адкрыла эру моў праграмiравання трэцяга пакалення. Яна дазволіла выкарыстоўваць розныя лікавыя тыпы даных, неабходныя для складаных разлікаў: цэлыя, рэчыўныя (рэчаісныя) і комплексныя.

Далейшае развіццё моў праграмiравання дазволіла дабавіць магчымасць работы з іншымі тыпамі даных. Сучасныя мовы праграмiравання дазваляюць працаваць з вялікай колькасцю тыпаў даных.

У асяроддзі праграміравання PascalABC рэалізавана больш за 30 розных тыпаў даных.



Обзор типов

Типы в **PascalABC.NET** подразделяются на простые, структурированные, типы указателей, процедурные типы, последовательности и классы.

К *простым* относятся целые и вещественные типы, логический, символьный, перечислимый и диапазонный тип.

Тип данных называется *структурированным*, если в одной переменной этого типа может содержаться множество значений.

К структурированным типам относятся массивы, строки, записи, кортежи, множества, файлы и классы.

Особым типом данных является последовательность, которая хранит по-существу алгоритм получения данных последовательности один за другим.

Все простые типы, кроме вещественного, называются *порядковыми*. Только значения этих типов могут быть индексами статических массивов и параметрами цикла `for`. Кроме того, для порядковых типов используются функции `Ord`, `Pred` и `Succ`, а также процедуры `Inc` и `Dec`.

Прыклад 14.4. Прыклады апісання пераменных:

```
var x: real;
var x1, y1: real;
var a_1, a_2, a_3: real;
```

Дыяпазон магчымых значэнняў тыпу `real` задаецца лікамі ў стандартным уяўленні ад $-1.8 \cdot 10^{308}$ да $1.8 \cdot 10^{308}$. Найменшы дадатны лік тыпу `real` прыбліжана роўны $5.0 \cdot 10^{-324}$. Пры вылічэннях у ліку захоўваецца да 16 лічбаў.

Камп'ютар можа апрацоўваць даныя розных тыпаў: цэлыя і рэчаісныя лікі, сімвалы, тэксты і інш.

Тып даных вызначае спосаб захоўвання даных у памяці камп'ютара, дыяпазон магчымых значэнняў даных і аперацыі, якія з гэтым тыпам даных можна выконваць.

Каб выкарыстаць якую-небудзь пераменную, яе трэба апісаць. Апісанне пераменных выконваецца да пачатку праграмы (каманды **begin**) (прыклад 14.4). Пры апісанні пераменнай вылучаецца памяць для захоўвання яе значэння. У працэсе выканання праграмы значэнне пераменнай можа змяняцца.

Для апісання пераменных выкарыстоўваецца каманда **var** (скар. ад англ. *variable* — пераменная). Фармат запісу каманды:

var <імя пераменнай>: <тып>;

Для абазначэння імя пераменнай выкарыстоўваюць літары лацінскага алфавіта, лічбы і знак «_». Першым знакам павінна быць літара або знак падкрэслівання.

Тып даных `real` у мове Pascal дазваляе працаваць з лікамі і выконваць над імі арыфметычныя дзеянні.

14.3. Аператар прысвойвання

Адной з асноўных каманд для апрацоўкі даных у праграме з'яўляецца аператар прысвойвання.

Аператар прысвойвання прызначаны для таго, каб:

- задаваць значэнні пераменным;
- вылічваць значэнні арыфметычнага выразу (вынік вылічэння будзе запісаны як значэнне пераменнай).

Фармат запісу аператара:

```
<імя пераменнай>:= <выраз>;
```

(Разгледзьце прыклад 14.5.)

У запісе арыфметычнага выразу выкарыстоўваюцца знакі матэматычных дзеянняў.

Матэматычныя аперацыі	Запіс у Pascal
+ (складанне)	+
− (адніманне)	-
· (множанне)	*
: (дзяленне)	/

Прыярытэт выканання аперацый адпавядае прынятаму ў матэматыцы: спачатку выконваюцца множанне і дзяленне, а затым складанне і адніманне. Для змянення парадку дзеянняў у выразах выкарыстоўваюць дужкі.

Прыклад 14.5. Прыклады запісу аператара прысвойвання:

```
x:= 7;
x1:= 3.5;
a_1: = 20 * (x + x1) - 32;
y:= y + 7;
```

Прыклад 14.6. Запішам аператар прысвойвання на Pascal для матэматычных выразаў:

Выраз	Запіс на Pascal
$S = 2(a + b)$	`S:=2*(a+b);`
$S = a^2$	`S:=a*a;`
$a = \frac{x + y}{3}$	`a:=(x+y)/3;`

Прыклад 14.7. Запішам аператар прысвойвання, пасля выканання якога значэнне пераменнай a павялічыцца ў 2 разы, а пераменнай b паменшыцца на 3.

У Pascal дапушчальныя каманды прысвойвання наступнага выгляду:

```
a:= a * 2;
```

Сэнс такой каманды наступны: з ячэйкі памяці здабываецца значэнне пераменнай a, затым яно памнажаецца на 2, вынік запісваецца ў тую ж ячэйку памяці. Старое значэнне пераменнай a будзе страчана.

Запіс аператара прысвойвання для змянення значэння пераменнай b наступны:

```
b:= b - 3;
```

У PascalABC.NET вызначаны аператары прысвойвання са значкамі +=, −=, *=, /=. Яны дазваляюць змяніць значэнне пераменнай. Напрыклад:

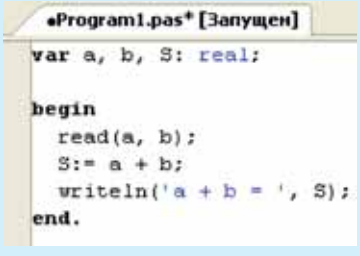
```
a *= 2;  // павялічыць a ў 2 разы;
b -= 3;  // паменшыць b на 3.
```

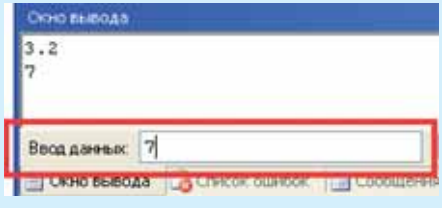
Прыклад 14.8. Увесці два лікі, знайсці і вывесці іх суму.

Тэкст праграмы:

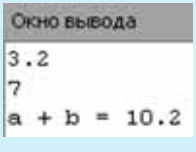
```
var a, b, S: real;
begin
 read(a, b);
 S:= a + b;
 writeln('a + b =', S);
end.
```

Увод даных:





Вынік:



Для запісу звычайнага дробу выкарыстоўваецца знак дзялення. Знак множання апускаць нельга. Цэлая частка дробавага ліку аддзяляецца ад дробавай часткі кропкай. (Разгледзьце прыклады 14.6 і 14.7 на с. 93.)

14.4. Увод даных

Пачатковыя значэнні пераменным можна задаваць не толькі з дапамогай аператара прысвойвання, але і шляхам уводу з клавіятуры. У гэтым выпадку, калі неабходны вылічэнні з новым наборам значэнняў зыходных даных, тэкст праграмы не трэба змяняць.

Каманда `read( )` прызначана для **ўводу даных.** У дужках праз коску пералічаюцца імёны пераменных, значэнні якіх неабходна ўвесці.

Увод даных адбываецца ў ніжняй частцы акна праграмы PascalABC. Для гэтага выкарыстоўваецца акно «Увод даных». Пасля націскання на кнопку «Увесці» або на клавішу «Enter» уведзеныя значэнні пераносяцца ў акно вываду. Пасля завяршэння работы праграмы ў гэтым жа акне будзе выведзены вынік (прыклад 14.8).

14.5. Структура праграмы

Усе праграмы на мове праграміравання Pascal маюць агульную структуру. У праграме можна вылучыць наступныя часткі:

- загаловак (неабавязковы);
- бібліятэкі, што падключаюцца (модулі) (калі падключаць дадатковыя бібліятэкі не трэба, раздзел адсутнічае; вядомыя бібліятэкі: `Drawman`, `Robot`, `RobTasks`);
- апісанне пераменных з вызначэннем іх тыпу;
- апісанне дапаможных алгарытмаў (калі выкарыстоўваць дапаможныя алгарытмы не трэба, раздзел адсутнічае);
- **begin ... end.** — службовыя словы, што абрамляюць цела асноўнай праграмы, у якой знаходзяцца выконваемыя каманды; **begin** пачынае выконваемую частку праграмы, а **end.** (кропка ў канцы абавязковая) яе завяршае.

У мінімальна магчымым наборы праграма складаецца з пустога цела праграмы: **begin end.** Праграма, што змяшчае ўсе раздзелы, паказана ў прыкладзе 14.9.

Для кожнага раздзела вызначана ключавое службовае слова, якім ён пачынаецца. Пры напісанні праграмы ключавыя словы вылучаюцца паўтлустым шрыфтам.

Прыклад 14.9. Праграма, якая змяшчае ўсе раздзелы (падлічваецца колькасць зафарбаваных клетак у полі Робата памерам 10 × 10):

```
//загаловак праграмы
//(неабавязкова)
program Primer;
//апісанне бібліятэк
uses Robot, RobTasks;
//апісанне пераменных
var k: integer;
//апісанне дапаможных
//алгарытмаў
procedure line;
begin
 for var i:= 1 to 9 do
 begin
  if CellIsPainted then
   k:= k + 1;
  right;
 end;
 if CellIsPainted then
  k:= k + 1;
end;
procedure back;
begin
 for var i:= 1 to 9 do
  left;
end;
//асноўная праграма
begin
//цела праграмы
 Task('myrob11');
 k:= 0;
 for var i:= 1 to 9 do
 begin
  line; back; down;
 end;
 line;
 writeln(k);
end.
```

1. Якая каманда мовы праграміравання Pascal прызначана для выведу даных?
2. Што вызначае тып даных?
3. Для чаго выкарыстоўваецца каманда прысвойвання?
4. Якая каманда мовы праграміравання Pascal прызначана для ўводу даных?
5. З якіх раздзелаў складаецца праграма на мове праграміравання Pascal?

Практыкаванні

1 Для праграмы з прыкладу 14.2 выканайце наступныя заданні (файл з праграмай можна спампаваць):

1. Замяніце ўсе каманды `writeln` на каманды `write` і выканайце праграму. Што адбылося? Растлумачце чаму.
2. Як зменіцца вынік работы праграмы, калі ў зыходным тэксце замяніць усе каманды `write` на `writeln`?
3. Змяніце праграму так, каб тэкст на экране выглядаў наступным чынам:

```
Прывітанне! Я камп'ютар!!! Я ўмею выконваць праграмы!
Ты сёння напісаў сваю першую праграму!!!
Я выканаў тваю праграму. Паглядзі на экране вынік!
```

2 Унясіце неабходныя змяненні ў праграму з прыкладу 14.3, каб дзеянні выконваліся ў тым парадку, у якім запісаны, г. зн. спачатку складанне, а потым множанне.

3 Уводзіцца ўзрост карыстальніка ў гадах. Вызначыце ўзрост карыстальніка праз 5 гадоў.

4 Напішыце праграму, у якой уводзяцца два лікі *a* і *b*. Затым першы лік памяншаецца ў 2 разы, а другі павялічваецца на 30. Выведзіце зменены значэнні пераменных.

5 Напішыце праграму для вылічэння значэння лікавых выразаў:

1. $23 + 45 \cdot 11 - 15$.
2. $\dfrac{37 + 2 \cdot 27}{41}$.
3. $\dfrac{5638 - 2347}{49} + \dfrac{123 \cdot 756}{4455}$.

§ 15. Арганізацыя вылічэнняў

Пры рашэнні любой задачы чалавеку даводзіцца выконваць наступныя дзеянні:

- вызначэнне зыходных даных (што дадзена ў задачы);
- вызначэнне вынікаў (што трэба атрымаць);
- апрацоўка зыходных даных у адпаведнасці з вядомымі правіламі так, каб атрымаць вынік.

Ужываючы дадзеныя правілы пры рашэнні задачы па праграмiраванню, атрымаем наступныя этапы рашэння задачы:

I. Вызначэнне зыходных даных.

II. Вызначэнне вынікаў.

III. Складанне алгарытму рашэння задачы.

IV. Вызначэнне тыпаў даных для пераменных, што выкарыстоўваюцца пры рэалізацыі алгарытму.

V. Напісанне праграмы.

VI. Тэсціраванне праграмы.

VII. Аналіз вынікаў.

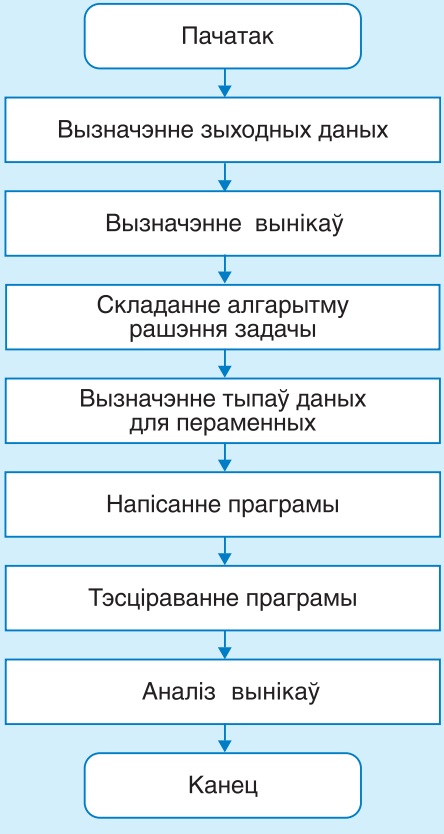
(Разгледзьце прыклад 15.1.)

Тэсціраванне праграмы — праверка правільнасці работы праграмы пры розных наборах зыходных даных.

Прыклад 15.1. Рашэнне задач па фізіцы прынята афармляць пэўным чынам.

Злева запісваецца тое, што дадзена і што трэба атрымаць, справа — паслядоўнасць дзеянняў, якая прыводзіць да рашэння задачы. Аналагічна афармляюцца рашэнні задач па хіміі, геаметрыі.

Этапы рашэння задачы па праграмiраванні можна паказаць наступным чынам:



Прыклад 15.2.

V. Праграма:

```
var  x, y, z, a:  real;
begin
 write('увядзіце  x  =  ');
 read(x);
 write('увядзіце  y  =  ');
 read(y);
 write('увядзіце  z  =  ');
 read(z);
 a:=(2*x+3*y-z)/(3+x*x);
 writeln('a  =  ',a);
end.
```

VI. Тэсціраванне праграмы.

Запусціце праграму і ўвядзіце значэнні: $x = 2$, $y = 3$, $z = 1$.

Вынік работы праграмы павінен быць наступным:

```
Окно вывода
увядзіце x = 2
увядзіце y = 3
увядзіце z = 1
a = 1.71428571428571
```

А для значэнняў $x = 2$, $y = 4$, $z = 2$ атрымаем:

```
Окно вывода
увядзіце x = 2
увядзіце y = 4
увядзіце z = 2
a = 2
```

VII. Праверка правільнасці вылічэнняў можа быць выканана на калькулятары.

15.1. Вылічэнне значэння арыфметычнага выразу

Прыклад 15.2. Дадзены пераменныя x, y, z. Напішам праграму для вылічэння значэння выразу $a = \dfrac{2x + 3y - z}{3 + x^2}$.

Этапы выканання задання:

I. Вызначэнне зыходных даных: пераменныя x, y, z.

II. Вызначэнне вынікаў: пераменная a.

III. Алгарытм рашэння задачы:

1. Увод зыходных даных.
2. Вылічэнне значэння выразу.
3. Вывад выніку.

IV. Апісанне пераменных.

Усе пераменныя, вызначаныя для рашэння задачы, маюць тып `real`.

У прыведзеным прыкладзе перад кожнай камандай уводу запісана каманда выводу з тлумачэннямі пра тое, значэнне якой пераменнай трэба ўводзіць.

Пры напісанні праграм для вылічэння значэння арыфметычнага выразу часта дапускаюць наступныя памылкі:

`2x+3;` `(x-4)(x+2);`	Прапушчаны знак *
`(2+y)/(x*x;`	Прапушчана дужка

Будзьце ўважлівымі!

15.2. Выкарыстанне мовы праграмірования для рашэння задач

Прыклад 15.3. Напішам праграму для рашэння геаметрычнай задачы. Зададзены квадрат з даўжынёй стараны a. Патрабуецца знайсці яго плошчу і перыметр.

Этапы выканання задання:

I. Вызначэнне зыходных даных: пераменная a (даўжыня стараны).

II. Вызначэнне вынікаў: пераменныя S (плошча) і P (перыметр).

III. Алгарытм рашэння задачы:

1. Увод зыходных даных.

2. Вылічэнне значэнняў плошчы ў матэматыцы выконваецца па формуле $S = a^2$, а перыметра — па формуле $P = 4a$. У праграме гэтым формулам будуць адпавядаць каманды прысвойвання:

$S := a * a$; $P := 4 * a$.

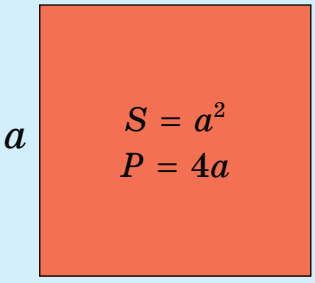
3. Вывад выніку.

IV. Апісанне пераменных:

Усе пераменныя, вызначаныя для рашэння задачы, маюць тып `real`.

Звярніце ўвагу: запіс формул у аператары прысвойвання можа адрозніваццa ад запісу матэматычных формул.

Прыклад 15.3.



V. Праграма:

```
var a,S,P: real;
begin
 write('увядзіце a = ');
 read(a);
 s:= a*a;
 p:= 4*a;
 writeln('плошча = ',S);
 writeln('перыметр = ',P);
end.
```

VI. Тэсціраванне праграмы.

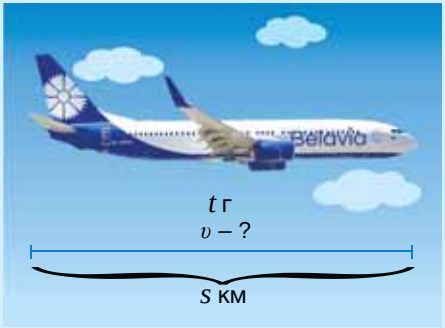
Запусціце праграму і ўвядзіце значэнне $a = 5.2$.

Вынік работы праграмы павінен быць наступным:

```
Окно вывода
увядзіце a = 5.2
плошча = 27.04
перыметр = 20.8
```

VII. Праверка правільнасці вылічэнняў можа быць выканана на калькулятары.

Прыклад 15.4.



V. Праграма:

```
var s, t, v: real;
begin
 write('увядзіце s = ');
 read(s);
 write('увядзіце t = ');
 read(t);
 v:= s / t;
 writeln('скорасць = ', v);
end.
```

VI. Тэсціраванне праграмы.

Запусціце праграму і ўвядзіце значэнні $s = 3550$ і $t = 4$.

Вынік работы праграмы павінен быць наступным:

```
Окно вывода
увядзіце s = 3550
увядзіце t = 4
скорасць = 887.5
```

VII. Праверка правільнасці вылічэнняў можа быць выканана на калькулятары.

Прыклад 15.4. Напішам праграму для рашэння фізічнай задачы. Адлегласць паміж двума гарадамі складае s км. Самалёт пралятае гэту адлегласць за t г. Вызначыце скорасць самалёта.

Этапы выканання задання:

I. Вызначэнне зыходных даных: пераменныя s (адлегласць) і t (час).

II. Вызначэнне вынікаў: пераменная v (скорасць).

III. Алгарытм рашэння задачы:

1. Увод зыходных даных.

2. Згодна з формулай адлегласці: $s = vt$. Адсюль выразім v: $v = \frac{s}{t}$.

3. Вывад выніку.

IV. Апісанне пераменных:

Усе пераменныя, вызначаныя для рашэння задачы, маюць тып `real`.

Пры напісанні праграм звяртайце ўвагу на фармаціраванне іх тэксту:

- у першай пазіцыі на экране пішуць толькі словы **var**, **begin** **end**, а астатнія са зрухам на 2—4 пазіцыі ўправа;
- калі ў праграме некалькі частак, то іх можна аддзяліць адна ад адной пустым радком.

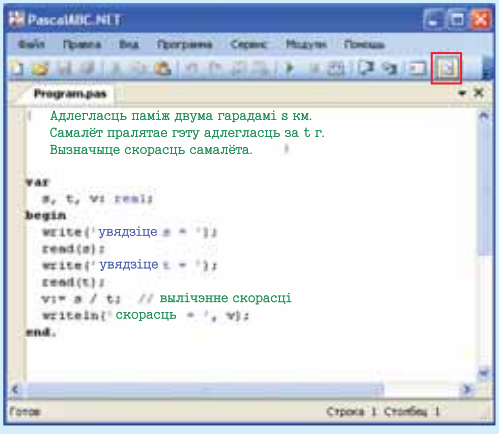
Выкананне гэтых правіл павышае чытальнасць праграмы.

У праграме можна выкарыстоўваць **каментарыі** — тэкст, які не аналізуецца пры запуску праграмы на выкананне.

Тэкст пасля знакаў // лічыцца каментарыем і вылучаецца зялёным колерам (прыклад 15.5).

У каментарыях зручна запісваць умову задачы. Тлумачэнні да каманд, запісаныя як каментарыі, патрэбныя для разумення дзеяння, выкананага камандай. У мове праграмірования Pascal каментарыі можна запісаць у некалькі радкоў. Тады тэкст, які з'яўляецца каментарыем, змяшчаюць у фігурных дужках.

Прыклад 15.5. Праграма з каментарыямі і адфармаціраваным кодам:



Значок для фармаціравання кода на панэлі інструментаў мае наступны выгляд:

1. Пералічыце этапы рашэння задачы па праграмірованню.
2. Што разумеюць пад тэсціраваннем праграмы?
3. Для чаго можна выкарыстоўваць каментарыі?

Практыкаванні

1 Дадзены x, y, z. Напішыце праграму для вылічэння значэння арыфметычных выразаў.

1. $a = \dfrac{x + y - z}{x^2 + 2}$.
2. $a = 5\dfrac{2x - z}{3 + y^2}$.
3. $a = (1 + z)\dfrac{x + \dfrac{y}{x^2 + 4}}{2 + \dfrac{1}{x^2 + 4}}$.

2 Напішыце праграму для рашэння геаметрычнай задачы.

1. Знайдзіце даўжыню акружнасці і плошчу круга зададзенага радыуса.

2*. Знайдзіце вугал пры аснове раўнабедранага трохвугольніка, калі вядомы вугал пры вяршыні.

3 Напішыце праграму для рашэння фізічнай задачы.

1. Веласіпедыст едзе з пастаяннай скорасцю v км/г. За колькі мінут ён праедзе адлегласць у s км?

2*. Аўтамабіль праходзіць першую частку шляху даўжынёй s_1 км за t_1 мін, участак шляху даўжынёй s_2 км за t_2 мін і ўчастак даўжынёй s_3 км за t_3 мін. Знайдзіце сярэднюю скорасць аўтамабіля ў км/г.

4 Напішыце праграму для рашэння хімічнай задачы.

1. У арганізме чалавека на долю атамаў кісларoду прыпадае 65 % ад масы цела. Знайдзіце масу атамаў кісларoду для сваёй масы цела.

2*. Маса атама кісларoду роўна $26.56 \cdot 10^{-27}$ (гэты лік на мове Pascal запісваецца так: 26.56E-27, літара E — англійская). Колькі атамаў кісларoду змяшчаецца ў вашым целе?

§ 16. Рэалізацыя алгарытмаў работы з цэлалікавымі данымі

У PascalABC вызначаны розныя тыпы даных для работы з цэлымі лікамі, якія дазваляюць выконваць дзеянні над данымі з розных лікавых дыяпазонаў. Чым большы дыяпазон, тым больш месца ў памяці камп'ютара адводзіцца для захоўвання пераменных.

Некаторыя цэлалікавыя тыпы даных:

Тып	Дыяпазон значэнняў
`shortint`	–128..127
`smallint`	–32768..32767
`integer`, `longint`	–2147483648..2147483647
`byte`	0..255
`word`	0..65535

16.1. Цэлалікавы тып даных

Часта пры рашэнні задач трэба працаваць з цэлымі лікамі. Для гэтага ў Pascal выкарыстоўваецца тып даных `integer`. З дапамогай пераменных гэтага тыпу задаюцца цэлыя лікі ад –2147483648 да 2147483647. Для тыпу даных `integer` вызначаны наступныя аперацыі:

Матэматычныя аперацыі	Запіс у Pascal
+ (складанне)	+
– (адніманне)	–
· (множанне)	*
цэлалікавае дзяленне	`div`
знаходжанне астачы	`mod`

Для цэлалікавых даных не вызначана аперацыя дзялення, як для рэчаісных лікаў. Пры спробе выкарыстаць аперацыю дзялення будзе выдадзена памылка (прыклад 16.1).

Для арганізацыі вылічэнняў з цэлымі лікамі вызначаны аперацыі `div` і `mod`. Гэтыя аперацыі маюць такі ж прыярытэт, як і аперацыі дзялення і множання.

Прыклад 16.2. Дадзены два цэлыя лікі *a* і *b*. Напішам праграму, якая знаходзіць цэлую частку ад дзялення *a* на *b* і астачу.

Этапы выканання задання:

I. Вызначэнне зыходных даных: пераменныя *a* і *b*.

II. Вызначэнне вынікаў: пераменныя *c* (цэлалікавая дзель) і *d* (астача).

III. Алгарытм рашэння задачы:

1. Увод зыходных даных.
2. Цэлалікавую дзель знаходзім як вынік аперацыі `a div b`, астачу — `a mod b`.
3. Вывад выніку.

IV. Апісанне пераменных.

Усе пераменныя, вызначаныя для рашэння задачы, маюць тып `integer`.

Значэнне, якое выдаецца як вынік аперацыі `mod`, можа ад-

Прыклад 16.1. Памылка выкарыстання аперацыі дзялення для цэлалікавых тыпаў даных:

•Program.pas*

```
var a, b, c: integer;
begin
  read(a, b);
  c:= a / b;
  write(c);
end.
```

Список ошибок

	Строка	Описание
1	4	Нельзя преобразовать тип real к integer

Прыклад 16.2.

V. Праграма:

```
var a, b, c, d: integer;
begin
 read(a, b);
 c:= a div b;
 d:= a mod b;
 writeln('цэлая частка = ',
 c);
 writeln('астача = ', d);
end.
```

VI. Тэсціраванне праграмы.

Запусціце праграму і ўвядзіце значэнні $a = 11$ і $b = 4$.

Вынік работы праграмы павінен быць такім, як паказана ніжэй:

Окно вывода

```
11
4
цэлая частка = 2
астача = 3
```

Вынік аперацый `div` і `mod` для розных лікаў:

a	b	a div b	a mod b
17	3	5	2
−17	3	−5	−2
17	−3	−5	2
−17	−3	5	−2

Так, a **mod** b = a − (a **div** b) * b.

Прыклад 16.3.

V. Праграма:

```
var c, m, s: integer;
begin
 write('увядзіце c = ');
 readln(c);
 {мінуты}
 m:= c div 60;
 {секунды}
 s:= c mod 60;
 write(m, ':', s);
end.
```

VI. Тэсціраванне праграмы.

Запусціце праграму і ўвядзіце значэнне $c = 137$.

Вынік работы праграмы:

```
Окно вывода
увядзіце c = 137
2:17
```

Для значэння $c = 24$ атрымаем:

```
Окно вывода
увядзіце c = 24
0:24
```

розніваццa ад матэматычнага азначэння астачы (у матэматыцы пад астачай разумеюць неадмоўны лік). Калі астача не роўна нулю, то знак ліку, які з'яўляецца вынікам аперацыі `mod`, вызначае знак дзялімага.

16.2. Выкарыстанне цэлалікавых даных для рашэння задач

Прыклад 16.3. Няхай таймер паказвае час толькі ў секундах. Напішам праграму, якая пераводзіць час у мінуты і секунды.

Этапы выканання задання:

I. Вызначэнне зыходных даных: пераменная *c* (час у секундах).

II. Вызначэнне вынікаў: пераменныя *m* (поўная колькасць мінут) і *s* (астача секунд).

III. Алгарытм рашэння задачы:

1. Увод зыходных даных.
2. Для знаходжання поўнага ліку мінут трэба знайсці цэлую частку ад дзялення зыходнага ліку секунд на 60.
3. Секунды, што засталіся, знаходзім як астачу ад дзялення зыходнага ліку секунд на 60.
4. Вывад выніку.

IV. Апісанне пераменных.

Пераменныя, вызначаныя для рашэння задачы, маюць тып `integer`.

Прыклад 16.4. Зададзены двухзначны лік. Трэба памяняць месцамі першую і другую лічбы ліку.

Этапы выканання задання:

I. Вызначэнне зыходных даных: пераменная *a* (зыходны лік).

II. Вызначэнне вынікаў: пераменная *b* (пераўтвораны лік).

III. Алгарытм рашэння задачы:

1. Увод зыходных даных.

2. Для пераўтварэння ліку неабходна выканаць наступныя дзеянні:

а) у пераменнай *a*1 захаваем другую лічбу ліку. Для вылучэння лічбы з ліку трэба знайсці астачу ад дзялення зыходнага ліку на 10 (`a mod 10`);

б) для вылучэння першай лічбы (пераменная *a*2) трэба знайсці цэлую частку ад дзялення ліку на 10;

в) шуканы лік *b* атрымаем, калі памножым *a*1 на дзесяць і да атрыманага здабытку дададзім значэнне пераменнай *a*2.

3. Вывад выніку.

IV. Апісанне пераменных.

Усе пераменныя, вызначаныя для рашэння задачы, маюць тып `integer`.

Прыклад 16.4.

V. Праграма:

```
var
  a, b, a1, a2: integer;
 begin
  write('увядзіце a = ');
  readln(a);
  {Вылучэнне апошняй
лічбы}
  a1:= a mod 10;
  {Вылучэнне першай лічбы}
  a2:= a div 10;
  b:= a1 * 10 + a2;
  write('вынік = ', b);
end.
```

VI. Тэсціраванне праграмы.

Запусціце праграму і ўвядзіце значэнне $a = 25$.

Вынік работы праграмы павінен быць наступным:

```
Окно вывода
увядзіце a = 25
вынік = 52
```

Здаўна на Русі ўжывалася сістэма мер, якая адрознівалася ад сучаснай Міжнароднай сістэмы адзінак (СІ). Напрыклад:

1 локаць = 45 см;
1 аршын = 16 вяршкоў;
1 вяршок = 4 ногці;
1 ногаць ≈ 11 мм.

Прыклад 16.5.

V. Праграма:

```
var l, m, s, x: integer;
begin
 write('увядзіце l = ');
 readln(l);
 x:= l * 45;
 {метры}
 m:= x div 100;
 {сантыметры}
 s:= x mod 60;
 write(l, 'локцяў = ');
 write(m, ' м ', s, ' см');
end.
```

VI. Тэсціраванне праграмы.

Запусціце праграму і ўвядзіце значэнне $l = 7$.

Вынік работы праграмы павінен быць наступным:

```
Окно вывода
увядзіце l = 7
7 ëîêöÿў = 3 м 15 см
```

Прыклад 16.5. У гістарычнай кнізе даўжыня адрэзу тканіны вымяралася ў локцях. Напішам праграму, якая перавядзе локці ў метры і сантыметры.

Этапы выканання задання:

I. Вызначэнне зыходных даных: пераменная l (локці).

II. Вызначэнне вынікаў: пераменныя m (метры) і s (сантыметры).

III. Алгарытм рашэння задачы:

1. Увод зыходных даных.
2. Спачатку перавядзём локці ў сантыметры. Для гэтага колькасць локцяў трэба памножыць на 45 і захаваць значэнне ў пераменнай x.
3. Для вызначэння ліку метраў знойдзем цэлую частку ад дзялення x на 100.
4. Сантыметры, што засталіся, можна знайсці як астачу ад дзялення x на 100.
5. Вывад выніку.

IV. Апісанне пераменных:

Усе пераменныя, вызначаныя для рашэння задачы, маюць тып `integer`.

?

1. Які тып даных можна выкарыстоўваць у Pascal для работы з цэлалікавымі данымі?

2. Якое максімальнае значэнне можна задаць пераменнай тыпу `integer`?

3. Якія аперацыі вызначаны для цэлалікавых даных?

Практыкаванні

1 Вася напісаў праграму, якая пераводзіць даўжыню з метраў у кіламетры і метры. Але ён не можа вырашыць, дзе трэба выкарыстоўваць `div`, а дзе `mod`. Дапамажыце яму. Адкрыйце файл і выпраўце праграму.

```
var d, m, k: integer;
begin
  write('увядзіце d = ');
  readln(d);
  k:= d … 1000;
  m:= d … 1000;
  write(d, ' м = ');
  write(k, ' км ', m, ' м');
end.
```

2 Адкажыце на пытанні для прыкладу 16.4.

1. Пры якіх значэннях пераменнай a значэнне пераменнай b будзе такім жа?
2. Ці заўсёды ў выніку выканання праграмы мы будзем атрымліваць двухзначны лік? Чаму?
3. Паспрабуйце ўвесці трохзначны лік (напрыклад, 125). Растлумачце атрыманы вынік.

3 Напішыце праграмы для рашэння задач. Выкарыстоўвайце аперацыі `div` і `mod`.

1. Зададзены двухзначны лік. Знайдзіце сярэдняе арыфметычнае лічбаў ліку.
2. Зададзены двухзначны лік. Знайдзіце рознасць паміж колькасцю дзясяткаў і адзінак.
3. Дадзена маса ў грамах. Перавядзіце яе ў кілаграмы і грамы.
4. Плошча ўчастка вымяраецца ў арах. Знайдзіце колькасць поўных км^2.

4* Для старарускай сістэмы вагі вядомыя наступныя суадносіны:

1 беркавец = 10 пудоў = 400 фунтаў = 38 400 залатнікоў.

Напішыце праграму, якая пераводзіць масу, зададзеную ў залатніках, у фунты, пуды і беркаўцы.

Глава 4
АПАРАТНАЕ І ПРАГРАМНАЕ ЗАБЕСПЯЧЭННЕ КАМП'ЮТАРА

§ 17. Сучасныя камп'ютарныя ўстройствы

Прыклад 17.1.

Настольны камп'ютар

Ноўтбук

Планшэт

17.1. Віды камп'ютараў

Развіццё вылічальнай тэхнікі прывяло да з'яўлення вялікай разнастайнасці ўстройстваў. Сучасныя камп'ютары маюць розную канструкцыю і знешні выгляд.

Сукупнасць усіх устройстваў камп'ютара называюць яго **апаратным забеспячэннем.**

Настольны камп'ютар складаецца з сістэмнага блока і падключаных да яго знешніх устройстваў. Карыстальнік сам вызначае якасны і колькасны склад устройстваў, што падключаюцца да сістэмнага блока.

У **мабільных камп'ютарах** усе неабходныя ўстройствы знаходзяцца ў адным корпусе. Пераносныя камп'ютары маюць магчымасць бесправаднога падключэння да знешніх устройстваў і сетак.

Асноўныя разнавіднасці мабільных камп'ютараў:

1. **Ноўтбукі** — паўнавартасныя камп'ютары з клавіятурай, экра-

нам, цвёрдым дыскам і магчымасцю выкарыстання шырокага спектра праграм.

2. **Планшэтныя камп'ютары (планшэты)** маюць абмежаваныя магчымасці, віртуальную клавіятуру і аперацыйную сістэму з наборам каманд.

3. **Смартфоны** — тэлефоны з некаторымі магчымасцямі камп'ютара. Сучасныя смартфоны цудоўна спраўляюцца шмат з якімі задачамі, не ўласцівымі тэлефонам. Гэта работа з электроннай поштай, стварэнне і рэдагаванне тэкставых дакументаў, прагляд фільмаў, праслухоўванне музыкі і шмат што іншае.

Для рашэння найбольш складаных задач выкарыстоўваюць **суперкамп'ютары.** Яны валодаюць вялізнай вылічальнай магутнасцю і пераўзыходзяць па сваіх характарыстыках большасць існуючых у свеце камп'ютараў. Сярод галін іх ужывання можна адзначыць матэматычнае мадэляванне, метэаралогію, авіяцыйную прамысловасць, сейсмалогію і інш. Выявы розных відаў камп'ютараў паказаны ў прыкладзе 17.1.

Камп'ютар-трансформер (ноўтбук-планшэт)

Смартфон

Суперкамп'ютар

Адзін з магчымых камп'ютараў будучыні

Прыклад 17.2. Унутраныя ўстройствы камп'ютара.

Мацярынская плата

Працэсар

Аператыўная памяць канструктыўна ўяўляе сабой набор мікрасхем, размешчаных на адной невялікай плаце. Модулі аператыўнай памяці ўстаўляюцца ў адпаведныя раздымы мацярынскай платы.

Модуль аператыўнай памяці

17.2. Прызначэнне ўстройстваў персанальнага камп'ютара

Склад устройстваў (канфігурацыя) камп'ютара можа змяняцца ў залежнасці ад задач, якія неабходна рашаць.

Базавая канфігурацыя настольнага камп'ютара змяшчае наступныя функцыянальныя блокі: сістэмны блок, манітор, клавіятуру, мыш. У мабільных камп'ютарах гэтыя ўстройствы інтэграваны ў адзінае цэлае.

У **сістэмным блоку** размяшчаюцца: мацярынская плата, блок сілкавання, устройствы памяці, карты расшырэнняў (відэакарта, гукавая карта, сеткавая карта).

Усе кампаненты камп'ютара звязаны паміж сабой самай вялікай пячатнай платай. Гэту плату называюць **мацярынскай платай**. На ёй устаноўлены працэсар.

Працэсар — найважнейшае ўстройства камп'ютара, яго мозг. Ён апрацоўвае інфармацыю, выконваючы вылічэнні.

Устройствы памяці прызначаны для захоўвання інфармацыі. Памяць камп'ютара бывае ўнутраная і знешняя.

Унутраная памяць знаходзіцца ўнутры камп'ютара і прызначана для захоўвання пра-

грам і іх даных у працэсе работы камп'ютара.

Знешняя памяць прызначана для доўгачасовага і энерганезалежнага захоўвання праграм і даных. Да аднаго камп'ютара можна падключыць некалькі ўстройстваў знешняй памяці.

Унутраная памяць падзяляецца на аператыўную і пастаянную.

Аператыўная памяць (RAM) служыць для захоўвання праграм і даных, з якімі камп'ютар працуе ў дадзены момант.

Абмен данымі паміж працэсарам і аператыўнай памяццю выконваецца за вельмі кароткія прамежкі часу. Пры выключэнні электрасілкавання ўся інфармацыя знікае з аператыўнай памяці.

Пастаянная памяць (ROM) — энерганезалежная памяць для захоўвання праграм кіравання работай і тэсціравання ўстройстваў камп'ютара.

Акрамя праграмы першапачатковага тэсціравання камп'ютара, у пастаяннай памяці захоўваецца BIOS (базавая сістэма ўводу-выводу). Даныя ў паста-

Прыклад 17.3. Устройствы знешняй памяці.

Вінчэстар, які змяшчаецца ўнутры сістэмнага блока (на рысунку накрыўка вінчэстара знята).

Знешні вінчэстар, які падключаецца да партоў сістэмнага блока.

Для пераносу даных выкарыстоўваюць:

Аптычныя дыскі	Флэш-памяць
	

Прыклад 17.4. Перыферыйныя ўстройствы.

Відэапраектар

Дакумент-камера

Вэб-камера

Мікрафон

янную памяць заносяцца пры вырабе камп'ютара.

Асноўным устройствам доўгачасовага захоўвання інфармацыі з'яўляецца **вінчэстар (цвёрды дыск).**

Вінчэстар знаходзіцца ўнутры сістэмнага блока, але належыць да знешніх устройстваў памяці. Існуюць вінчэстары, якія могуць падключацца да сістэмнага блока.

Вінчэстар можна ўмоўна падзяліць на некалькі **лагічных дыскаў (раздзелаў).** Абслугоўванне аднаго лагічнага раздзела не закранае іншыя раздзелы.

Акрамя вінчэстара, да ўстройстваў знешняй памяці належаць аптычныя дыскі і флэш-памяць.

Устройствы, якія не ўваходзяць у сістэмны блок, называюць **перыферыйнымі.**

Перыферыйныя ўстройствы ўводу-вываду падключаюцца да **партоў (раздымаў)** мацярынскай платы або да карт расшырэнняў. Звычайна яны выводзяцца на заднюю панэль камп'ютара.

З прызначэннем клавіятуры, мышы, манітора, прынтара і сканера вы пазнаёміліся ў 6-м класе.

Разгледзім прызначэнне іншых перыферыйных устройстваў.

Відэапраектар прызначаны для праектавання відарыса на вялікі экран.

Дакумент-камера дазваляе атрымліваць лічбавы відарыс любых прадметаў.

Вэб-камера — малапамерная лічбавая відэа- або фотакамера, здольная ў рэальным часе фіксаваць відарысы, прызначаныя для далейшай перадачы па сетцы Інтэрнэт.

Для ўводу гукавой інфармацыі выкарыстоўваюць **мікрафон**, а для ўзнаўлення — акустычныя сістэмы (**гукавыя калонкі** і **навушнікі**). Часам мікрафон і навушнікі аб'ядноўваюцца ў адно ўстройства — **гарнітуру**.

(Разгледзьце прыклады 17.2—17.4 на с. 110—113).

Калонкі

Навушнікі

Гарнітура

? 1. Што такое апаратнае забеспячэнне камп'ютара?
2. Якія канструкцыі камп'ютараў вам вядомыя?
3. Назавіце мабільныя камп'ютары.
4. Для чаго прызначаны працэсар?
5. Якая бывае памяць камп'ютара?
6. Для чаго прызначана аператыўная памяць?
7. Для чаго прызначана пастаянная памяць?
8. Назавіце ўстройствы знешняй памяці.
9. Пералічыце ўстройствы ўводу інфармацыі.
10. Назавіце ўстройствы выводу інфармацыі.

Практыкаванні

1 Вызначыце аб'ём памяці цвёрдага дыска камп'ютара, да якога вы маеце доступ у вашай навучальнай установе або дома. Які аб'ём займае інфармацыя, што захоўваецца на ім? Запішыце даныя ў сшытак.

2 Якім, на ваш погляд, можа быць камп'ютар будучыні? Напішыце эсэ на гэту тэму.

3 З дапамогай графічнага рэдактара Paint стварыце відарыс камп'ютара будучыні.

§ 18. Аперацыйная сістэма

18.1. Віды аперацыйных сістэм

Аперацыйная сістэма (АС) — комплекс праграм, які дазваляе карыстальніку мець зносіны з камп'ютарам, кіруе ўстройствамі камп'ютара, праграмамі і інфармацыяй, што захоўваецца ў памяці камп'ютара.

Камп'ютары працуюць пад кіраваннем розных аперацыйных сістэм. Часам на адным камп'ютары ўстанаўліваюць некалькі АС. Найбольш распаўсюджаныя сем'і АС для настольных камп'ютараў і ноўтбукаў: Windows, Mac OS, Linux. Шмат якія смартфоны і планшэты працуюць пад кіраваннем АС Android (прыклады 18.1 і 18.2).

Пры ўключэнні камп'ютара аперацыйная сістэма пачынае

Прыклад 18.1.

Значкі найбольш распаўсюджаных АС	
	Windows 10
	Mac OS
	Linux
	Android

аўтаматычна загружацца з дыска ў аператыўную памяць камп'ютара і застаецца там да выключэння камп'ютара. АС увесь час знаходзіцца ў рабочым стане. Загрузка АС з доўгачасовай памяці камп'ютара ў аператыўную называецца **загрузкай камп'ютара.**

АС выводзіць на экран манітора запрашэнне да работы ў якой-небудзь форме. У адказ карыстальнік дае каманду на выкананне канкрэтнага дзеяння. Калі такая каманда вядомая АС і ў дадзены момант часу можа быць выканана, то АС яе выконвае, калі не — карыстальніку выдаецца адпаведнае паведамленне. Пасля гэтага АС чакае наступную каманду карыстальніка. Такі рэжым работы называецца **дыялогавым рэжымам.**

У сучасных АС дыялогавы рэжым графічны.

У **графічным рэжыме** карыстальнік можа задаваць каманды АС, выбіраючы іх з розных меню. Пры такой арганізацыі дыялога камандам АС адпавядаюць пэўныя значкі (невялікія карцінкі).

Выбар каманд часта ажыццяўляюць з дапамогай мышы. Так,

Прыклад 18.2. Графічныя рэжымы АС.

Windows 7

Windows 10

Mac OS

Linux

АС Android

Першая аперацыйная сістэма UNIX была распрацавана ў 1969 г. у падраздзяленні Bell Labs кампаніі AT & T. З тых часоў была створана вялікая колькасць розных UNIX-сістэм.

У 1981 г. карпарацыя IBM размясціла запыт на стварэнне аперацыйнай сістэмы, якая павінна была выкарыстоўвацца ў новай сям'і камп'ютараў IBM PC. Microsoft выкупіла права на аперацыйную сістэму 86-DOS у Seattle Computer Products і пачала работу па яе мадыфікацыі па патрабаванні IBM.

напрыклад, арганізаваны дыялог карыстальніка з аперацыйнымі сістэмамі сям'і Windows.

АС дазваляе карыстальніку працаваць з іншымі праграмамі. Яна знаходзіць праграму ў доўгачасовай памяці (на дыску), загружае яе ў аператыўную памяць і прымушае працэсар выконваць каманды дадзенай праграмы. У аператыўную памяць можа быць загружана некалькі праграм. Такі рэжым работы называецца **шматзадачным**. АС таксама арганізуе і кантралюе работу ўсіх устройстваў камп'ютара.

На цвёрды дыск АС устанаўліваецца з загрузачнага дыска.

Прыклад 18.3.

Значкі Рабочага стала	
	Камп'ютар
	Карзіна
	Сеткавае акружэнне
	Мае дакументы
	Праваднік

18.2. Элементы графічнага карыстальніцкага інтэрфейсу

Карыстальніцкі інтэрфейс — сукупнасць сродкаў і спосабаў узаемадзеяння чалавека і камп'ютара.

Сучасныя АС маюць графічны карыстальніцкі інтэрфейс. Асноўнымі элементамі графічнага інтэрфейсу з'яўляюцца **вокны** і **меню**.

Разгледзім карыстальніцкі інтэрфейс АС Windows 7. Пасля загрузкі АС Windows 7 на экране

камп'ютара з'яўляецца Рабочы стол з ярлыкамі і значкамі (прыклад 18.3).

Рабочы стол — асноўнае акно графічнага інтэрфейса АС, якое займае ўсю прастору экрана.

Унізе Рабочага стала знаходзіцца **Панэль задач.** На ёй размешчана кнопка **Пуск**, кнопкі выконваемых праграм і адкрытых акон дакументаў, індыкатар мовы і часу (даты і часу), індыкатар сеткавых падключэнняў, рэгулятар гучнасці гуку (прыклад 18.4). Кнопка **Пуск** выклікае **Галоўнае меню.** З дапамогай Галоўнага меню карыстальнік можа атрымаць доступ да ўсіх праграм, установленых на камп'ютары.

Ярлык уяўляе сабой спасылку на аб'ект (файл, каталог) і заўсёды змяшчае стрэлку . Двайная пстрычка левай клавішай мышы запускае праграму, на якую спасылаецца ярлык. **Значок** без стрэлкі выкарыстоўваецца для абазначэння самога аб'екта — папкі або файла (выгляд значкоў некаторых дакументаў паказаны ў прыкладзе 18.5).

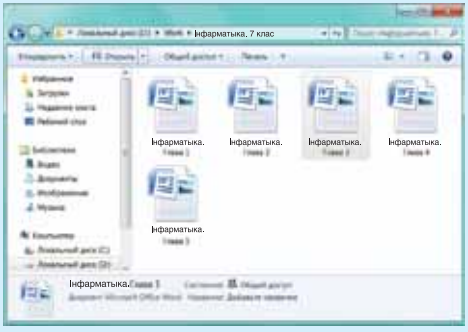
Прыклад 18.4.

Элементы Панэлі задач	
	Кнопка **Пуск**
	Значкі загружаных праграм і адкрытых папак
	Індыкатар даты і часу
	Індыкатар сеткавых падключэнняў
	Рэгулятар гучнасці гуку
	Індыкатар мовы

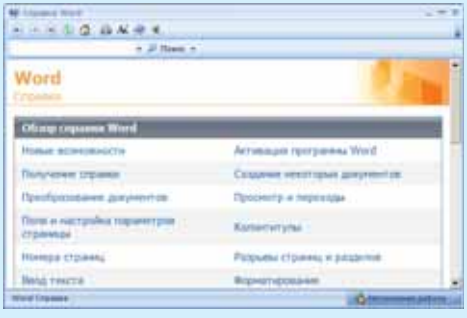
Прыклад 18.5.

Значкі праграм і адпаведных дакументаў	
Значок праграмы	Значок дакумента
Тэкставы рэдактар Блакнот	
	
Power Point	
	
PascalABC.NET	
	

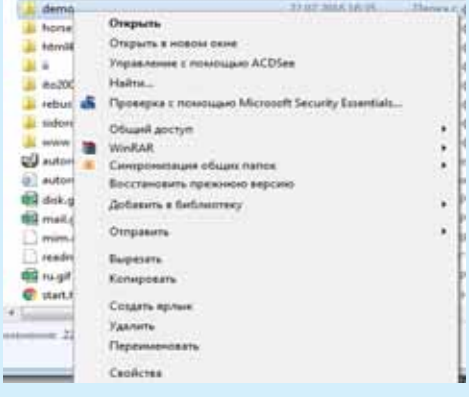
Прыклад 18.6. Дыялогавае акно «Адкрыццё дакумента».



Прыклад 18.7. Акно даведачнай сістэмы MS Word.



Прыклад 18.8. Кантэкставае меню папкі.



У працэсе работы на Рабочым стале могуць раскрывацца дыялогавыя вокны, вокны папак, вокны праграм і вокны даведачнай сістэмы.

Дыялогавыя вокны прызначаны для арганізацыі дыялогу карыстальніка з камп'ютарам (прыклад 18.6).

Акно праграмы з'яўляецца пасля запуску праграмы.

У **акне даведачнай сістэмы** можна атрымаць даведку пра работу праграмы (прыклад 18.7). Адкрываюцца такія вокны пасля націскання на клавішу F1 у працэсе работы з праграмай.

У 6-м класе вы даведаліся, што кіраваць работай праграм дазваляюць розныя меню.

Меню — спіс для выбару каманд.

Доступ да ўсіх каманд, магчымых для дадзенага аб'екта, можна атрымаць з дапамогай **кантэкставага меню** (прыклад 18.8).

Кантэкставае меню выклікаецца пстрычкай правай клавішай мышы. Але ёсць і іншыя спосабы:

- спецыяльнай клавішай на клавіятуры:

- спалучэннем клавіш Shift + F10.

18.3. Асноўныя элементы файлавай сістэмы

Патрэбная нам інфармацыя захоўваецца ў камп'ютары ў выглядзе файлаў. Працуюць з імі з дапамогай **файлавай сістэмы.**

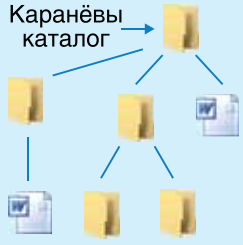
Файлавая сістэма прызначана для арганізацыі выканання аперацый над файламі і папкамі (каталогамі).

Кожная АС падтрымлівае пэўныя файлавыя сістэмы. Аб'ектамі любой файлавай сістэмы з'яўляюцца файлы, папкі і дыскі.

Структура файлавай сістэмы Windows уяўляе сабой сістэму ўкладзеных папак (прыклад 18.9). У кожнай папцы могуць захоўвацца іншыя папкі і файлы. Пра папкі або файлы, што знаходзяцца ў іншай папцы, гавораць, што яны ўкладзены ў гэту папку. Структуры, пабудаваныя на прынцыпах укладзенасці (падпарадкавання), называюцца **іерархічнымі.** Файлавая сістэма АС Windows з'яўляецца іерархічнай.

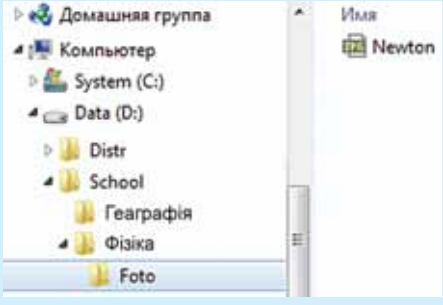
Файлавая сістэма дазваляе ствараць, перайменоўваць і выдаляць файлы, пераносіць і капіраваць файлы з аднаго носьбіта на другі, шукаць файлы, што захоўваюцца на розных носьбітах, запускаць праграмы на выкананне.

Прыклад 18.9. Структуру файлавай сістэмы можна ўявіць як дрэва з перавернутай кронай:



Прыклад 18.10. Шлях да файла `Newton.jpg`:

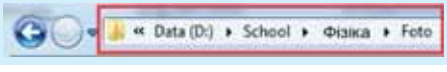
`D:\School\Фізіка\Foto\`



Пры пераходзе з папкі `Foto` ў папку `Фізіка` адбываецца пад'ём на адзін узровень іерархіі. Пры адваротным перамяшчэнні — спуск.

Поўнае імя файла `D:\School\Фізіка\Foto\Newton.jpg` азначае: 1) файл `Newton.jpg` укладзены ў папку `Foto`; 2) папка `Foto` ўкладзена ў папку `Фізіка`; 3) папка `Фізіка` ўкладзена ў папку `School`, што знаходзіцца на дыску `D`:

Адлюстраванне шляху да файла `Newton.jpg` у адрасным радку:



Прыклад 18.11. Сямікласнік Вадзім працаваў з каталогам E:\Mat\Form\1_ch\. Спачатку ён узняўся на адзін узровень уверх, затым яшчэ раз уверх, потым апусціўся ў каталог Prog, у якім знаходзіўся файл z1.pas. Які шлях да гэтага файла?

Рашэнне

Вадзім працаваў з каталогам E:\Mat\Form\1_ch\. Узняўшыся на адзін узровень уверх, ён апынуўся ў каталогу E:\Mat\Form\. Узняўшыся яшчэ раз уверх, ён апынуўся ў E:\Mat\. Пасля, апусціўшыся ў каталог Prog, Вадзім апынуўся ў E:\Mat\Prog\. Гэта і ёсць шлях да файла z1.pas.

Прыклад 18.12.

Значкі файлавых менеджараў для АС Windows	
	Правадзік
	Total Commander
	Far
	WinSCP
	Q-Dir

Адны папкі стварае карыстальнік; іншыя, такія як Мой камп'ютар або Кошык, ствараюцца аўтаматычна пры ўстаноўцы аперацыйнай сістэмы. Каб знайсці файл у файлавай структуры, трэба паказаць шлях да файла.

Шлях да файла — паслядоўнасць папак, пачынаючы ад самай верхняй і заканчваючы той, у якой непасрэдна захоўваецца файл. Шлях да файла разам з імем файла называюць **поўным імем файла.**

Шлях пачынаецца з **каранёвай папкі** (імені дыска) і змяшчае паслядоўнасць імёнаў папак, у якія ўкладзены файл. Дыскі называюцца вялікімі літарамі англійскага алфавіта з двукроп'ем пасля літары. Імёны дыскаў пачынаюцца з C:. Пасля імені кожнай папкі ставіцца адваротны слэш (гл. прыклад 18.10 на с. 119 і прыклад 18.11).

Для работы з файламі і папкамі выкарыстоўваюць праграмы, якія называюць **файлавымі менеджарамі.**

Для кожнай АС створаны розныя файлавыя менеджары. У АС Windows папулярныя Правадзік, Total Commander, Far і інш. (прыклад 18.12).

18.4. Тыпавыя аперацыі з файламі і папкамі

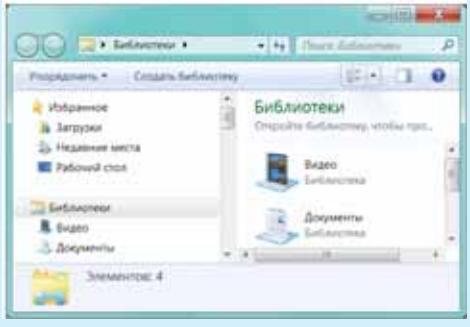
Праграма **Правадні́к** дазваляе карыстальніку ствараць і выдаляць файлы і каталогі (папкі), капіраваць і пераносіць іх з аднаго носьбіта на іншы, а таксама пераймяноўваць файлы і папкі (спосабы запуску Правадніка паказаны ў прыкладзе 18.13; змяніць адлюстраванне інфармацыі ў Правадніку можна з дапамогай спосабаў, паказаных у прыкладзе 18.14). Дзеянні па капіраванні, пераносе і выдаленні файлаў аналагічныя дзеянням па капіраванні, пераносе і выдаленні тэкставых або графічных фрагментаў.

Пасля **капіравання** атрымліваюць два аднолькавыя файлы. У папцы-крыніцы трэба выбраць аб'ект для капіравання і ў кантэкставым меню аб'екта выканаць каманды **Правка → Копировать.** Пасля гэтага неабходна выбраць папку-прыёмнік і ў яе кантэкставым меню выканаць каманды **Правка → Вставить.** У якасці крыніцы і прыёмніка можа быць выбрана адна і тая ж папка.

Дзеянні па **перамяшчэнні** файла (папкі) аналагічныя дзеянням па капіраванні. Спачатку ў

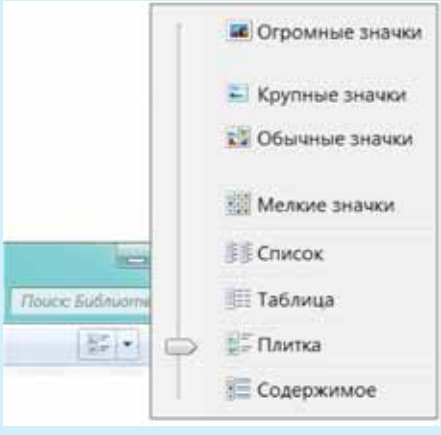
Прыклад 18.13. Розныя спосабы запуску Правадніка:

- ярлык праграмы Правадні́к на Рабочым стале;
- кнопка **Пуск (Все программы → Стандартные → Проводник);**
- кантэкставае меню кнопкі **Пуск.**



Акно файлавага менеджара Правадні́к

Прыклад 18.14. Спосаб адлюстравання інфармацыі ў акне Правадніка выбіраецца з дапамогай меню **Вид** або кнопкай **Изменить представление** ( ) на Панэлі інструментаў.



Прыклад 18.15. «Гарачыя» клавішы, якія выкарыстоўваюцца ў Правадніку:

Выразаць	Ctrl + X
Капіраваць	Ctrl + C
Уставіць	Ctrl + V
Вылучыць усё	Ctrl + A

Прыклад 18.16.

Дадатковыя аперацыі з файламі, якія падтрымліваюцца файлавымі менеджарамі:

Файлавы менеджар / Аперацыя			
Архівацыя файлаў	Так	Так	Так
Колеравае вылучэнне імёнаў файлаў	Не	Так	Так
Устаўка змесціва буфера абмену ў выглядзе файла	Частко-ва	Так	Так

папцы-крыніцы трэба выбраць аб'ект для перамяшчэння і ў кантэкставым меню аб'екта выканаць каманды **Правка** → **Вырезать.** Пасля гэтага выбіраецца папка-прыёмнік і ў кантэкставым меню папкі-прыёмніка выконваецца каманда **Правка** → **Вставить.**

Непатрэбныя файлы і папкі могуць быць выдалены. Для гэтага іх вылучаюць, а затым націскаюць клавішу Delete на клавіятуры або выконваюць каманду **Файл** → → **Удалить.** Пасля выдалення аб'екты звычайна змяшчаюцца ў **Карзіну.** Карзіна прызначана для часовага захоўвання выдаленых аб'ектаў. Выдаленыя з Карзіны аб'екты аднавіць з дапамогай аперацыйнай сістэмы немагчыма.

Для выканання аперацый адразу з некалькімі аб'ектамі іх трэба вылучыць. Калі аб'екты размешчаны побач, то іх вылучаюць так: пстрыкаюць мышшу на першым аб'екце і, утрымліваючы клавішу Shift, пстрыкаюць на апошнім. Калі аб'екты, што вылучаюцца, не размешчаны побач, то іх вылучаюць, утрымліваючы клавішу Ctrl.

Для работы ў Правадніку можна выкарыстоўваць «гарачыя»

клавішы (прыклад 18.15), што дазваляе паскорыць выкананне некаторых дзеянняў.

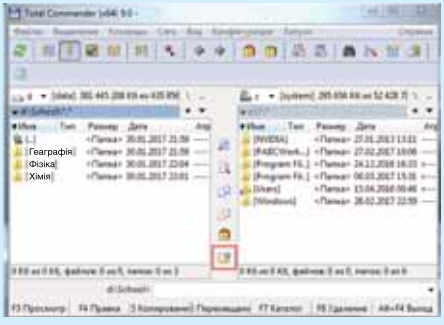
Унутры адкрытай папкі можна стварыць новую папку. Для гэтага трэба націснуць кнопку **Новая папка** ў меню акна Правадніка. З'явіцца папка з імем «Новая папка» (прыклад 18.16). Гэта імя можна памяняць на іншае.

Каб **перайменаваць** файл або папку, можна выкарыстаць адпаведны пункт кантэкставага меню файла або папкі.

З працэсам стварэння новай папкі ў файлавым менеджары Total Commander пазнаёмцеся самастойна, разгледзеўшы прыклад 18.17.

Прыклад 18.17. Стварэнне папкі ў Total Commander:

- адкрыць папку, унутры якой будзе знаходзіцца новая папка;
- націснуць кнопку **Создать каталог** або клавішу F7:



- у акне **Создать новый каталог (папку)**, якое адкрыецца, увесці імя новай папкі:

1. Што такое аперацыйная сістэма?
2. Якія асноўныя функцыі выконвае АС?
3. Што такое значок?
4. Што такое ярлык?
5. Для чаго выкарыстоўваецца кнопка **Пуск**?
6. Што такое файлавая сістэма?
7. Якія аперацыі дазваляе выконваць файлавая сістэма?
8. Якія праграмы называюць файлавымі менеджарамі?
9. Як запусціць файлавы менеджар?

Практыкаванні

1 Настаўнік працаваў у папцы `E:\Для ўрокаў\7 клас\Заданні\`. Потым перайшоў на ўзровень вышэй, увайшоў у папку `Тэма 4` і выдаліў з яе файл `Абагульненне.docx`. Якое поўнае імя файла, што выдаліў настаўнік?

2 Змяніце фон Рабочага стала. Для гэтага выканайце наступныя дзеянні:

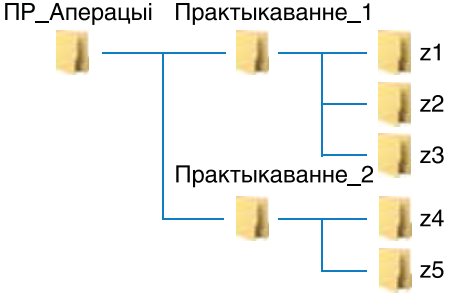
1. Пстрыкніце правай клавішай мышы па Рабочым стале і ў кантэкставым меню выберыце **Персонализация.**
2. У акне, якое адкрыецца, выберыце адну з тэм.

Пакажыце настаўніку вынік сваёй работы.

3 Выкарыстоўваючы кнопку **Пуск**, загрузіце на Рабочы стол наступныя праграмы з падменю Стандартныя: Блакнот, Калькулятар, Paint, WordPad. Размясціце вокны з праграмамі Каскадам, Стосам, Побач, выкарыстоўваючы кантэкставае меню Панэлі задач. Пакажыце настаўніку сваю работу, пасля гэтага закрыйце ўсе вокны.

4 Разнясіце значкі па розных вуглах Рабочага стала. Выстрайце значкі. Выкарыстоўваючы кантэкставае меню Рабочага стала, адсартуйце значкі па тыпе, даце, памеры. Пакажыце настаўніку вынік сваёй работы.

5 Стварыце дрэва папак наступнай структуры:



1. У папцы `z1` стварыце файл — дакумент MS Word з імем `text1.docx`, у якім адкажыце на пытанне 8 пасля гэтага параграфа (рубрыка «Пытанні і заданні для праверкі ведаў»).
2. У папцы `z2` стварыце відарыс смайла ў графічным рэдактары Paint, захаваўшы яго пад імем `ris1.bmp`.

3. Скапіруйце файл `ris1.bmp` з папкі `z2` у папку `z4`.
4. Скапіруйце папку `Практыкаванне _ 1` у папку `z5`. Перайменуйце папку `z5` у папку з імем `Copy`. Скапіраваныя папкі `z1`, `z2`, `z3` перайменуйце ў `Copy1`, `Copy2`, `Copy3` адпаведна. Выдаліце папку `Copy3`.

§ 19. Лакальная камп'ютарная сетка

Жаданне перадаваць інфармацыю ад аднаго камп'ютара да іншага, забяспечыць карыстальнікам сумесны доступ да тэхнічных устройстваў, праграмнага забеспячэння і інфармацыйных рэсурсаў камп'ютараў выклікала неабходнасць аб'яднання камп'ютараў у адзіную сетку.

Камп'ютарная сетка — аб'яднанне камп'ютараў, якое забяспечвае сумеснае выкарыстанне сеткавых рэсурсаў.

Камп'ютары, размешчаныя на невялікіх адлегласцях, могуць быць аб'яднаны ў **лакальную сетку**. Гэта, як правіла, сетка адной арганізацыі, навучальнай установы і інш. (прыклад 19.1).

Паводле спосабу арганізацыі лакальныя камп'ютарныя сеткі падзяляюцца на **аднарангавыя** і **сеткі з вылучаным серверам**.

У аднарангавых сетках усе камп'ютары раўнапраўныя. Сетка з вылучаным серверам мае адзін высокапрадукцыйны камп'ютар, які кіруе работай усёй сеткі. Гэты камп'ютар называецца **серверам**. Ён дае магчымасць астатнім камп'ютарам сеткі, якія называюцца **кліентамі**, сумесна

Прыклад 19.1.

Схема лакальнай сеткі

Прыкладам лакальнай камп'ютарнай сеткі з'яўляецца сетка ў кабінеце інфарматыкі. Яна існуе для таго, каб навучэнцы маглі працаваць з аднымі і тымі ж інфармацыйнымі рэсурсамі і выкарыстоўваць агульны прынтар.

Адзіных правілаў паводзін карыстальнікаў у лакальнай сетцы не існуе. Тым не менш адзначым некаторыя агульныя патрабаванні:

- не перадавайце іншым карыстальнікам ваша імя і пароль для ўваходу ў сетку;
- па магчымасці захоўвайце інфармацыю на дыску вашага камп'ютара, а не на дысках агульнага карыстання.

Прыклад 19.2. Да апаратнага забеспячэння работы лакальнай сеткі належаць сеткавыя платы (карты) і спецыяльны кабель. Сеткавымі платамі павінны быць аснашчаны ўсе камп'ютары сеткі. Яны прызначаны для прыёму і перадачы інфармацыі ў сеткі.

Сеткавая карта

Сеткавы кабель

Сеткавы порт

У бесправадных лакальных сетках выкарыстоўваецца пункт доступу, а на кожным камп'ютары павінна быць установлена спецыяльная бесправадная сеткавая плата тыпу Wi-Fi.

выкарыстоўваць яго рэсурсы і можа кіраваць іх работай.

Паводле спосабу падключэння камп'ютарныя сеткі могуць быць **праваднымі** і **бесправаднымі.**

Для арганізацыі работы камп'ютараў у лакальнай сетцы неабходна адпаведнае апаратнае (прыклад 19.2) і праграмнае забеспячэнне.

Праграмную падтрымку работы камп'ютараў у лакальнай сетцы выконвае аперацыйная сістэма.

Камп'ютары аб'ядноўваюць у сеткі для сумеснага выкарыстання сеткавых рэсурсаў. **Сеткавымі рэсурсамі (рэсурсамі сеткі)** камп'ютараў могуць з'яўляцца:

1) тэхнічныя ўстройствы (мадэмы, прынтары, дыскаводы і інш.);

2) праграмнае забеспячэнне (аперацыйныя сістэмы, розныя рэдактары і інш.);

3) інфармацыйныя рэсурсы (файлы з інфармацыяй).

Для доступу да сеткавых рэсурсаў часта бывае трэба ўвесці імя карыстальніка і пароль.

Карыстальнік, на камп'ютары якога знаходзіцца рэсурс (файл, дыск, папка, устройства), з'яўляецца яго ўладальнікам і мае поўны доступ да гэтага рэсурсу. Уладальнік рэсурсу можа дазволіць

іншым карыстальнікам сеткі доступ да свайго дыска, папкі, файла.

Прагляд даступных сеткавых рэсурсаў ажыццяўляецца ў папцы **Сетка**. У ёй адлюстроўваюцца агульныя рэсурсы сеткі, да якой падключаны камп'ютар (камп'ютары, папкі, файлы, прынтары).

Найважнейшай характарыстыкай работы лакальнай сеткі з'яўляецца скорасць перадачы інфармацыі ў ёй — колькасць інфармацыі, якая перадаецца за адзінку часу. Скорасць перадачы інфармацыі па сетцы звычайна вымяраецца ў біт/с. (Разгледзьце рашэнне задачы ў прыкладзе 19.3.)

Прыклад 19.3. Вызначыце аб'ём файла камп'ютарнай прэзентацыі, калі перадача яго па сетцы адбываецца за 5 с пры скорасці 1 024 000 біт/с. Запішыце атрыманы вынік у кілабайтах.

Рашэнне

$1\,024\,000$ біт/с $\cdot$ 5 с =
$= (2^{10} \cdot 10^3 \cdot 5)$ біт =
$= 2^{10} \cdot (2^3 \cdot 5^4)$ біт $= 2^{13} \cdot 5^4$ біт.

Перавядзём біты ў кілабайты:

1 байт = 8 біт, або 2^3 біты;

1 Кбайт = 1024 байты,
або 2^{10} байты;

1 Кбайт $= 2^3 \cdot 2^{10} = 2^{13}$ біты;

$(2^{13} \cdot 5^4) / 2^{13} = 5^4 = 625$ Кбайт.

1. Для чаго камп'ютары аб'ядноўваюць у сеткі?
2. Якая сетка называецца лакальнай?
3. З дапамогай чаго камп'ютары аб'ядноўваюцца ў лакальную сетку?
4. Якія бываюць лакальныя сеткі?
5. Назавіце сеткавыя рэсурсы.

Практыкаванні

1 Вызначыце аб'ём відэафайла, калі перадача яго па сетцы доўжылася 1 мін 20 с пры скорасці $80 \cdot 10^{20}$ біт/с. Запішыце атрыманы вынік у мегабайтах.

2 Вызначыце аб'ём гукавога файла, калі перадача яго па сетцы доўжылася 0,5 с пры скорасці 155 мегабіт/с. Запішыце атрыманы вынік у байтах.

3 Вызначыце скорасць перадачы інфармацыі па сетцы, калі архіўны файл аб'ёмам 1,5 Гбайт перадаваўся 2 мін.

4 Вызначыце скорасць перадачы інфармацыі па сетцы, калі файл з камп'ютарнай гульнёй аб'ёмам 3,2 Гбайт перадаваўся 3 мін 45 с.

§ 20. Архівацыя

У 40-х гг. XX ст. вучоныя, якія працавалі ў галіне інфармацыйных тэхналогій, прыйшлі да высновы, што можна распрацаваць такі спосаб захоўвання даных, пры якім прастора для захоўвання будзе расходавацца больш эканомна. Аднымі з першых алгарытмаў па сціскан-ні даных з'яўляюцца алгарыт-мы Шэнана — Фано і Хафмана.

Клод Элвуд Шэнан (1916—2001) — амерыканскі інжы-нер і матэматык, заснавальнік тэорыі інфармацыі. Большасць базавых паняццяў тэорыі сціс-кання інфармацыі была распра-цавана Клодам Шэнанам.

Роберт Марыа Фано (1917—2016) — італьянска-амерыканскі вучоны, вядомы працамі ў га-ліне тэорыі інфармацыі. Ён незалежна ад Клода Шэнана вынайшаў алгарытм сціскання інфармацыі.

20.1. Праграмы-архіватары

Для рацыянальнага захоўван-ня інфармацыі на камп'ютарных носьбітах выкарыстоўваюцца праграмы-архіватары, якія дазваля-юць прадставіць інфармацыю ва ўпакаваным выглядзе. Архівы ства-раюць у наступных выпадках:

- неабходна стварыць рэзервовыя копіі найбольш важных файлаў;
- патрабуецца вызваліць месца на дыску;
- неабходна перадаць файлы па электроннай пошце;
- плануецца перанесці вялікую колькасць файлаў на іншы но-сьбіт;
- трэба ахоўваць інфармацыю ад несанкцыянаванага доступу — запароліць яе.

Упакоўваць файлы і размяш-чаць іх у спецыяльных архівах дазваляюць **праграмы-архіватары. Архіўны файл (архіў)** захоў-вае ва ўпакаваным выглядзе іншыя файлы (адзін або некалькі), якія пры неабходнасці можна дастаць з архіва ў першапа-чатковай форме.

Праграмы-архіватары могуць выконваць наступныя функцыі:

- змяшчэнне зыходных файлаў у архіў;

- даставанне файлаў з архіва;
- выдаленне файлаў з архіва;
- прагляд зместу архіва;
- праверка архіва.

Памер архіва звычайна меншы за памер усіх файлаў, якія ў яго ўваходзяць. Для пераўтварэння інфармацыі праграмы-архіватары выкарыстоўваюць розныя алгарытмы, таму памеры архіваў, што змяшчаюць адны і тыя ж файлы, але створаны з дапамогай розных архіватараў, могуць адрознівацца.

Інфармацыя ў архіве захоўваецца ў закадзіраваным выглядзе, таму для прагляду змесціва архіўнага файла трэба выкарыстаць праграму-архіватар. Каб працаваць з файлам, яго неабходна дастаць з архіва. Робяць гэта, выкарыстоўваючы тую ж праграму, з дапамогай якой ствараўся архіў, або з дапамогай іншай праграмы, што распазнае дадзены тып архіва.

Дэвід Хафман (1925—1999) распрацаваў метад пабудовы мінімальна-залішніх кодаў. Вучоны ўнёс важны ўклад у інфарматыку і ў мноства іншых галін ведаў (па большай частцы ў электроніку). У 1952 г. стварыў алгарытм кадзіравання, вядомы як алгарытм, або код, Хафмана.

Вялікія архівы даных захоўваюць у спецыяльных сховішчах інфармацыі — датацэнтрах.

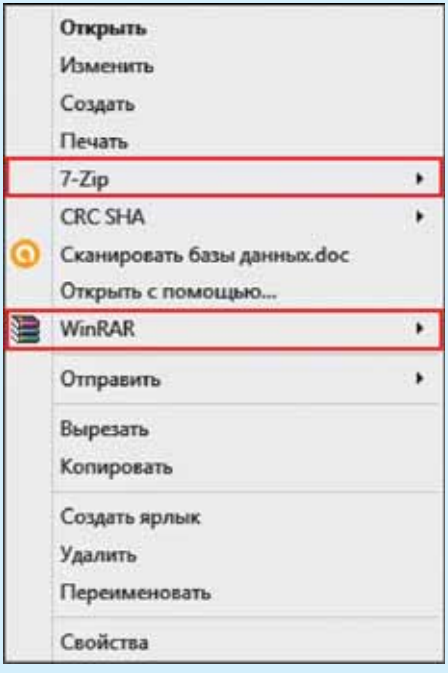
20.2. Стварэнне архіваў і даставанне файлаў з архіва

Сёння існуе вялікая колькасць праграм-архіватараў: WinRar, 7-Zip, WinZip і інш. (прыклад 20.1). Архіўныя файлы маюць расшырэнні, што адпавядаюць праграмам, з дапамогай якіх іх ствараліе: .rar, .7z, .zip. Пры

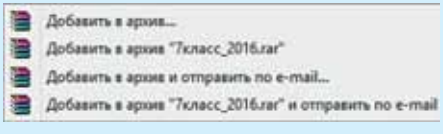
Прыклад 20.1.

Значкі праграм-архіватараў			
	WinRar		WinZip
	7-Zip		WinAce

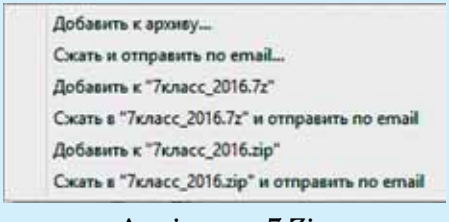
Прыклад 20.2. Кантэкставае меню з выбарам архіватара:



Прыклад 20.3. Каманды меню архіватара па дабаўленні файлаў у архіў:



Архіватар WinRar



Архіватар 7-Zip

праглядзе спіса файлаў у Правадніку архіўныя файлы адзначаюцца значкамі  (.rar),  (.7z),  (.zip).

Пры ўстаноўцы праграм-архіватараў дзеянні па стварэнні архіваў і па даставанні файлаў з архіва дабаўляюцца ў кантэкставае меню любога аб'екта файлавай сістэмы.

Для стварэння архіўнага файла неабходна:

1. Адкрыць Праваднік.
2. Вылучыць файлы.
3. Пстрыкнуць правай клавішай мышы.
4. Выбраць праграму-архіватар (прыклад 20.2).
5. Выбраць адну з каманд:

а) «Добавить в архив (к архиву)»;

б) «Добавить в архив (к архиву)» з прапанаваным імем (прыклад 20.3).

Архіў з прапанаваным імем ствараецца ў той жа папцы. Калі выбрана каманда «Добавить в архив (к архиву)», то карыстальніку трэба задаць імя архіва і вызначыць папку, у якой ён будзе захоўвацца.

Для даставання файлаў з архіва трэба:

1. Адкрыць Праваднік.
2. Выбраць архіўны файл.

3. Пстрыкнуць правай клавішай мышы.

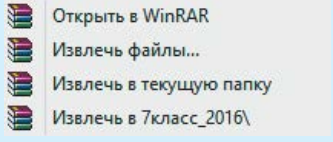
4. Выбраць адну з каманд:

а) «Извлечь в текущую папку (Распаковать здесь)»;

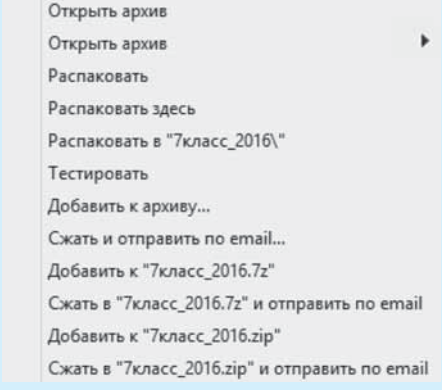
б) «Извлечь файлы... (Распаковать)» (прыклад 20.4).

Пры выбары каманды «Извлечь в текущую папку (Распаковать здесь)» файлы з архіва будуць змешчаны ў тую ж папку, у якой знаходзіўся архіў. Пры выбары каманды «Извлечь файлы... (Распаковать)» карыстальнік павінен вызначыць імя папкі, у якую дастануцца файлы.

Прыклад 20.4. Каманды меню па даставанні файлаў з архіва:



Архіватар WinRar



Архіватар 7-Zip

1. Які файл называюць архіўным?
2. Для чаго прызначаны праграмы-архіватары?
3. Як заархіваваць файл(-ы)?
4. Як дастаць файл(-ы) з архіва?

Практыкаванні

1 Стварыце рысунак у графічным рэдактары Paint. Захавайце файл на дыску. Заархівуйце гэты файл. Параўнайце памеры зыходнага і архіўнага файлаў.

2 Заархівуйце графічны файл рознымі архіватарамі. Параўнайце памеры атрыманых архіваў.

3 Заархівуйце файлы розных тыпаў: рысункі, тэксты, праграмы.

4* Параўнайце памеры зыходных файлаў і іх архіваў розных тыпаў. Якія файлы сціскаюцца лепш?

5 Дастаньце файлы з архіва, названага настаўнікам, у сваю папку.

§ 21. Праграмнае забеспячэнне

21.1. Класіфікацыя праграмнага забеспячэння

Праграмнае забеспячэнне (ПЗ) — сукупнасць усіх камп'ютарных праграм.

Камп'ютар разглядаюць як адзіную сістэму, што складаецца з апаратнага забеспячэння, праграмнага забеспячэння і інфармацыйных рэсурсаў. ПЗ камп'ютара ўвесь час змяняецца, удасканальваецца, дапаўняецца.

Праграмнае забеспячэнне камп'ютара паводле прызначэння:

1) сістэмнае;
2) прыкладное;
3) інструментальнае.

Сістэмнае ПЗ — праграмы для забеспячэння работы камп'ютара і камп'ютарных сетак. Сістэмнае ПЗ дазваляе карыстальніку ажыццяўляць кіраванне і кантроль над работай камп'ютара і камп'ютарнай сеткі, а таксама забяспечвае магчымасць выканання іншых праграм. Да сістэмнага праграмнага забеспячэння належаць ужо знаёмыя вам аперацыйныя сістэмы, файлавыя менеджары, архіватары.

Прыкладное ПЗ — праграмы для рашэння задач пэўнага класа

Прыклад 21.1. Прыкладныя праграмы.

Рэдактары апрацоўваюць інфармацыю, пададзеную ў тэкставай, графічнай, гукавой, лікавай форме. Напрыклад: тэкставы рэдактар MS Word (), графічны рэдактар Paint (), графічны рэдактар Inkscape ().

Мультымедыйныя праграмы спалучаюць магчымасць работы з відэафрагментамі, гукам, анімацыяй, статычнымі рысункамі і гіпертэкстам. Напрыклад: праграма для стварэння прэзентацый MS Power Point ().

Камунікацыйныя праграмы прызначаны для падтрымкі карыстальніцкага інтэрфейсу пры рабоце ў сетцы. Напрыклад, браўзеры: Internet Explorer (), Mozilla Firefox (), Google Chrome ().

Праграмы-перакладчыкі перакладаюць інфармацыю з адной натуральнай мовы на іншую. Існуюць праграмы-слоўнікі для перакладу асобных слоў ABBYY Lingvo (), праграмы для перакладу тэкстаў Promt () і анлайнавыя сэрвісы на Google, Yandex і інш.

прадметнай вобласці. Прыкладное ПЗ самае шматлікае (прыклад 21.1). Для наймення прыкладных праграм выкарыстоўваюць тэрмін **дадаткі**. Да прыкладных праграм належаць:

- праграмы агульнага прызначэння (патрабуюцца практычна кожнаму карыстальніку);
- праграмы спецыяльнага прызначэння (прызначаны для прафесійнага выкарыстання ў розных сферах дзейнасці);
- камп'ютарныя гульні.

Інструментальнае ПЗ прызначана для стварэння іншага ПЗ (прыклад 21.2). З інструментальным ПЗ працуюць праграмісты.

Праграмы паводле іх прававога статусу можна падзяліць на групы:

- платныя;
- якія свабодна распаўсюджваюцца;
- умоўна-бясплатныя;
- пробныя (ацэначныя);
- дэманстрацыйныя.

Прыклады ПЗ, якое распаўсюджваецца свабодна: АС Linux, графічны рэдактар Inkscape, антывірусныя праграмы AVAST і AVG, асяроддзе праграмірання PascalABC.NET.

Прыклад 21.2. Да інструментальнага праграмнага забеспячэння належыць ужо вядомае вам асяроддзе праграмірання PascalABC.NET ().

21.2. Шкодныя праграмы і спосабы аховы ад іх

Шкодныя праграмы — спецыяльна напісаныя праграмы, здольныя нанесці шкоду інфармацыі, што захоўваецца на камп'ютары, або вывесці камп'ютар са строю.

Паводле спосабу распаўсюджвання шкодныя праграмы падзяляюцца на камп'ютарныя вірусы, сеткавыя чарвякі і траянскія праграмы.

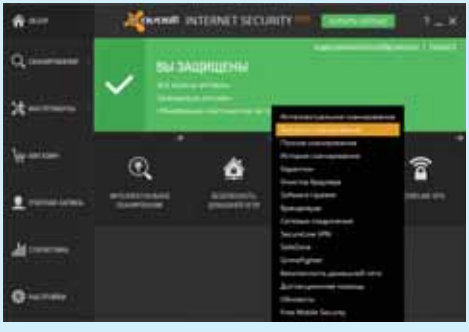
Камп'ютарныя вірусы могуць распаўсюджвацца самастойна,

Правобраз сучасных вірусаў — праграма «Дарвін». У 1962 г. інжынеры з амерыканскай кампаніі стварылі гульню з такой назвай. Сэнс яе складаўся ў выдаленні ўсіх копій праграмы суперніка і захопе поля бітвы. Праграмы-вірусы ўзніклі больш як праз дваццаць гадоў.

Рэгулярнае архіваванне і рэзервовае капіраванне файлаў дазволіць мінімізаваць шкоду ад віруснай атакі.

Прыклад 21.3. Інтэрфейс папулярных антывірусных праграм.

Антывірус Касперскага



AVAST

Norton AntiVirus

дабаўляючы свой код да іншых файлаў.

Сеткавыя чарвякі не змяняюць файлы на дысках, а распаўсюджваюцца ў камп'ютарнай сетцы — пранікаюць у аперацыйную сістэму камп'ютара, знаходзяць адрасы іншых камп'ютараў ці карыстальнікаў і рассылаюць па гэтых адрасах свае копіі.

Траянскія праграмы — гэта шкодныя праграмы, якія самі не распаўсюджваюцца, а, маскіруючыся пад папулярную праграму, схіляюць карыстальніка перапісаць шкодніка і ўстанавіць яго на свой камп'ютар самастойна.

Большасць вірусаў распрацоўваецца, каб зрабіць шкоду карыстальнікам, якія працуюць з аперацыйнымі сістэмамі сям'і Windows.

Пры заражэнні камп'ютара вірусам важна яго выявіць.

Прыметы заражэння:

1) марудная работа камп'ютара;

2) завісанні і збоі ў рабоце камп'ютара;

3) змяненне памераў файлаў;

4) змяншэнне памеру свабоднай аператыўнай памяці;

5) значнае павелічэнне колькасці файлаў на дыску;

6) знікненне файлаў і папак або скажэнне іх змесціва.

Для барацьбы са шкоднымі праграмамі выкарыстоўваюць праграмныя сродкі антывіруснай аховы: Антывірус Касперскага, Norton AntiVirus, AVAST, Dr.Web, AVG і інш. (прыклады 21.3 і 21.4).

Сканіраванне камп'ютара ў пошуках шкодных праграм звычайна выконваецца аўтаматычна пры кожным уключэнні. Пры сканіраванні антывірусная праграма шукае вірус шляхам параўнання кода праграм з кодамі вядомых ёй вірусаў, што захоўваюцца ў базе даных.

Адным з асноўных спосабаў барацьбы са шкоднымі праграмамі з'яўляецца своечасовая прафілактыка (прадухіленне заражэння).

Каб прадухіліць заражэнне камп'ютара, трэба выконваць наступныя рэкамендацыі:

1) не запускайце праграмы, атрыманыя з Інтэрнэту, без праверкі на наяўнасць у іх віруса;

2) правярайце ўсе знешнія носьбіты на наяўнасць вірусаў, перш чым капіраваць або адкрываць змешчаныя на іх файлы;

3) устанавіце антывірусную праграму і рэгулярна карыстайцеся ёю для праверкі камп'ютараў.

Прыклад 21.4.

Значкі папулярных антывірусных праграм	
	Антывірус Касперскага
	AVAST
	Norton AntiVirus
	Dr.Web
	AVG

Самым разбуральным вірусам за ўсю гісторыю іх існавання лічаць вірус «ILOVEYOU». Ён быў разасланы на паштовыя скрынкі з Філіпін у 2000 г.; у тэме ліста змяшчаўся радок «ILoveYou», а да ліста быў прыкладзены скрыпт «LOVE-LETTER-FOR-YOU.TXT.vbs». Пры адкрыцці ўкладання вірус рассылаў копію самога сябе ўсім кантактам у адраснай кнізе Windows, а таксама на адрас, пазначаны як адрас адпраўшчыка. Ён таксама здзяйсняў шэраг шкодных змяненняў у сістэме карыстальніка.

Вірус пашкодзіў больш за 3 мільёны камп'ютараў ва ўсім свеце. Шкода, якую чарвяк нанёс сусветнай эканоміцы, была настолькі вялікай, што вірус увайшоў у Кнігу рэкордаў Гінеса.

1. Што такое праграмнае забеспячэнне?
2. На якія класы можна падзяліць праграмнае забеспячэнне ў залежнасці ад прызначэння?
3. Якія праграмы называюцца шкоднымі?
4. Назавіце віды шкодных праграм.
5. Якія прыметы паказваюць на тое, што камп'ютар заражаны?
6. Што неабходна рабіць, каб прадухіліць заражэнне камп'ютара?
7. Назавіце праграмы антывіруснай аховы.

Практыкаванні

1 Пералічыце назвы вядомых вам праграм:

1. Платныя.
2. Якія свабодна распаўсюджваюцца.

2 Запішыце ў сшытку назвы антывірусных праграм, якія ўстаноўлены ў вас дома, у школьным камп'ютарным кабінеце, у вашых сяброў.

3 Вызначыце, да якога класа праграмнага забеспячэння належаць праграмы, значкі якіх паказаны на рысунках.

а 	*б* 	*в* 	*г* 
д 	*е* 	*ж* 	*з* 
і 	*к* 	*л* 	*м* 

Глава 5
РАБОТА З ВЕКТАРНАЙ ГРАФІКАЙ

§ 22. Паняцце вектарнай графікі

Адзін з напрамкаў выкарыстання камп'ютара — стварэнне і апрацоўка графічных відарысаў. Напрыклад, схем, чарцяжоў, рысункаў, фатаграфій.

Камп'ютарная графіка — галіна інфарматыкі, якая вывучае метады і сродкі стварэння і апрацоўкі відарысаў з дапамогай апаратнага і праграмнага забеспячэння камп'ютара.

У залежнасці ад спосабу ўяўлення ў памяці камп'ютарныя відарысы можна падзяліць на два віды: растравую і вектарную графіку.

Растравая графіка — відарысы, што ўяўляюць сабой сукупнасць пікселяў, афарбаваных у розныя колеры.

Вектарная графіка — відарысы ў выглядзе геаметрычных фігур (графічных прымітываў), апісаных матэматычнымі формуламі.

(Разгледзьце прыклад 22.1.)

Асноўныя галіны выкарыстання камп'ютарнай графікі: навуковая, дзелавая, канструктарская, ілюстрацыйная сферы.

Адрозненні ва ўяўленні графічнай інфармацыі ў растравым і вектарным выглядзе існуюць для графічных файлаў і спосабаў іх апрацоўкі. На экран манітора графічную інфармацыю можна вывесці толькі ў растравым выглядзе.

Прыклад 22.1.

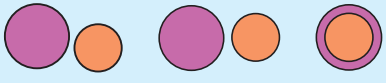
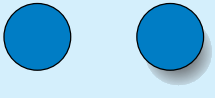
Вектарны відарыс

Вектарны відарыс можна параўнаць з аплікацыяй, якая складаецца з кавалачкаў каляровай паперы, наклееных (накладзеных) адзін на адзін. Аднак, у адрозненне ад аплікацыі, у вектарным відарысе лёгка мяняць форму і колер складовых частак.

Прыклад 22.2.

Графічныя прымітывы	
Лінія	
Прамавугольнік	
Эліпс	
Трохвугольнік, многавугольнік	
Зорка	

Прыклад 22.3. Цікавай асаблівасцю вектарных рэдактараў з'яўляецца магчымасць змянення формы нарысаваных ад рукі крывых. Таксама ў вектарных рэдактараў ёсць сродкі размяшчэння аб'ектаў адносна адзін аднаго і каманды спецэфектаў.

Змяненне формы крывых	
	
Узаемнае размяшчэнне аб'ектаў	
	
Выкарыстанне спецэфектаў	
Цень	
Аб'ём	

Графічныя прымітывы — простыя геаметрычныя фігуры: прамавугольнік, акружнасць, эліпс, лінія і г. д. (прыклад 22.2). З дапамогай матэматычных формул апісваюцца форма, колер і прасторавае становішча графічных прымітываў, што складаюць відарыс.

Графічны прымітыў — незалежны аб'ект, які можна рэдагаваць.

Становішча і форма графічных прымітываў задаюцца ў сістэме графічных каардынат, якая звязана з экранам. Пачатак каардынат размешчаны ў верхнім левым вугле экрана. Вось *OX* накіравана злева направа, вось *OY* — зверху ўніз. Каардынатная сетка супадае з сеткай пікселяў.

Добрыя якасці вектарных відарысаў:

1) невялікі памер графічнага файла;

2) пераўтварэнні без скажэнняў;

3) рысаванне ажыццяўляецца хутка і проста;

4) незалежнае рэдагаванне частак відарыса;

5) высокая дакладнасць прарысоўкі.

(Прачытайце прыклад 22.3.)

Аднак у вектарнай графіцы практычна немагчыма дасягнуць фотарэалістычнасці.

Для апісання колеру відарысаў выкарыстоўваюцца розныя колеравыя мадэлі.

Пад **колеравай мадэллю** разумеюць спосаб апісання колеру.

Колеравыя мадэлі апісваюць колеравыя адценні з дапамогай змешвання некалькіх асноўных колераў. Кожны колер можна раскласці на адценні і супаставіць яму набор лікаў — колеравых кардынат.

Асноўныя колеры разбіваюцца на адценні па яркасці — ад цёмнага да светлага. Кожнаму адценню прысвойваецца лікавае значэнне (напрыклад, самаму цёмнаму — 0, самаму светламу — 255).

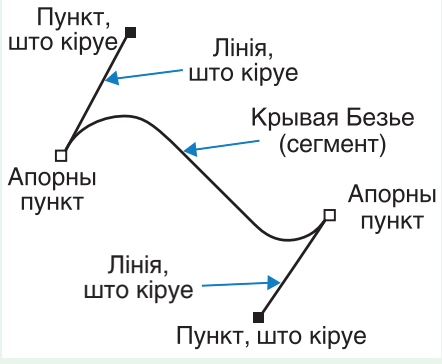
Адна з найбольш распаўсюджаных колеравых мадэлей называецца RGB (прыклад 22.4). Кожны колер у гэтай мадэлі ўяўляе сабой складанне трох асноўных колераў: чырвонага (Red), зялёнага (Green) і сіняга (Blue). Менавіта на такой мадэлі пабудавана ўзнаўленне колеру сучаснымі маніторамі і тэлевізарамі.

У паліграфіі выкарыстоўваецца колеравая мадэль, якая атрымала назву CMYK (пры-

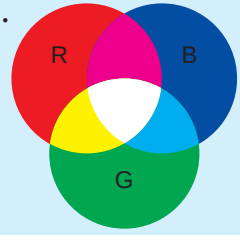
У мінулым інжынеры, ствараючы рысункі вялікіх дэталяў у натуральную велічыню, выкарыстоўвалі тонкія планкі, каб правесці крывыя па зададзеных пунктах. Гэтыя планкі называліся сплайнамі (гнуткімі лякаламі).

Зараз у вектарных рэдактарах таксама выкарыстоўваюцца сплайнавыя крывыя — крывыя Безье. Сваю назву яны атрымалі ў гонар французскага матэматыка П'ера Безье (1910—1999).

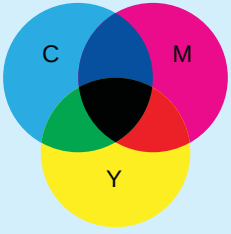
Вучоны прапанаваў апісваць крывую, абапіраючыся на вяршыні многавугольніка, які змяшчае яе ў сабе:



Прыклад 22.4. Колеравая мадэль RGB.



Прыклад 22.5. Колеравая мадэль CMYK.



Прыклад 22.6. Галіны выкарыстання вектарнай графікі:

- прамысловае праектаванне;
- візуалізацыя трохвымерных аб'ектаў;
- архітэктура і будаўніцтва;
- ландшафтны дызайн;
- пабудова графікаў паверхняў;
- паліграфія, рэклама.

клад 22.5). Асноўныя колеры ў ёй — блакітны, пурпурны, жоўты. Дадзеную колеравую мадэль часта ўжываюць для прынтараў.

Ужо вядомы вам графічны рэдактар Paint прызначаны для работы з растравай графікай. Растравыя графічныя рэдактары выкарыстоўваюць не столькі для стварэння відарысаў, колькі для іх апрацоўкі. Вектарныя рэдактары арыентаваны на стварэнне відарысаў. Вектарная графіка можа ўключаць у сябе і відарысы растравай графікі.

Вектарныя графічныя рэдактары дазваляюць захоўваць відарысы ў розных вектарных фарматах, сярод якіх можна вылучыць універсальныя графічныя фарматы і фарматы асобных вектарных рэдактараў.

Адным з недахопаў вектарнай графікі з'яўляецца праграмная залежнасць. Відарыс, створаны ў адным вектарным рэдактары, як правіла, не пераўтвараецца ў фармат іншай праграмы без хібнасцей.

Праграмы вектарнай графікі знайшлі ўжыванне ў галіне тэхнічнага рысавання, чарцёжна-графічных і афарміцельскіх работ, графічнага і паліграфічнага дызайну (прыклад 22.6).

Вядомыя вектарныя рэдактары: CorelDraw, Adobe Illustrator, Inkscape (прыклад 22.7). Вектарныя графічныя рэдактары дазваляюць выконваць разнастайныя аперацыі над графічнымі аб'ектамі.

Нягледзячы на разнастайнасць вектарных графічных рэдактараў, асноўныя прыёмы работы з вектарнымі відарысамі застаюцца нязменнымі.

Прыклад 22.7. CorelDraw і Adobe Illustrator — платныя праграмы. Рэдактарам Inkscape можна карыстацца бясплатна (http://www.inkscape.org).

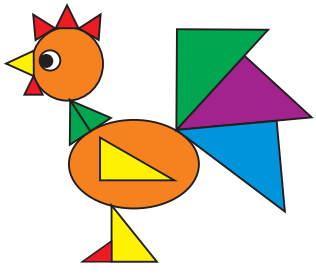
Значкі вектарных графічных рэдактараў	
	CorelDraw
	Adobe Illustrator
	Inkscape

1. Якая графіка называецца вектарнай?
2. Што такое колеравая мадэль?
3. Як узнаўляецца колер у колеравай мадэлі?
4. На якой мадэлі заснавана ўзнаўленне колеру маніторамі?
5. Што такое графічны прымітыў?
6. Як называюць праграму, што дазваляе працаваць з вектарнай графікай?

Практыкаванні

1 Прывядзіце прыклады графічных прымітываў.

2 Вызначыце, з якіх графічных прымітываў складзены відарыс пеўня.

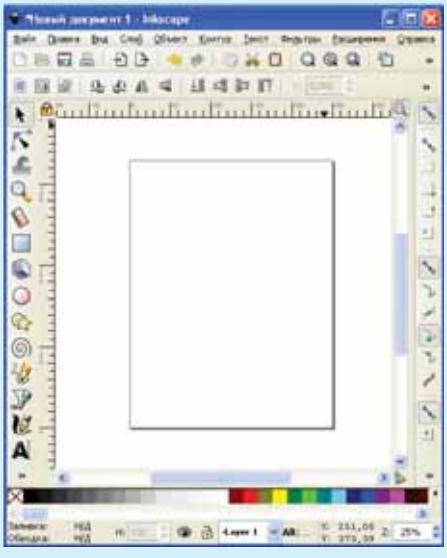


3 З дапамогай графічных прымітываў складзіце ў сшытку відарысы:

1. Доміка.
2. Кветкі.
3. Кошкі.

§ 23. Інтэрфейс вектарнага графічнага рэдактара Inkscape

Прыклад 23.1. Акно вектарнага графічнага рэдактара Inkscape.



Прыклад 23.2. Вектарны рэдактар Inkscape можа працаваць з файламі розных фарматаў, напрыклад:

.svg — фармат, які выкарыстоўвае рэдактар Inkscape;

.eps — фармат, які забяспечвае высокую якасць рысунка;

.png — растравы фармат відарысаў, які падтрымлівае празрыстасць фону;

.bmp — несціснуты растравы фармат відарысаў;

.pdf — фармат абмену дакументаў ад Adobe, які можа змяшчаць любыя спалучэнні: тэкст, шрыфты, растравую і вектарную графіку.

Разгледзім тэхналогію работы з вектарнай графікай на прыкладзе рэдактара Inkscape (прыклад 23.1).

Рэдактар мае ўбудаваны падручнік на рускай мове і зручны інтэрфейс, што дазваляе лёгка і хутка асвоіць асноўныя прыёмы работы з вектарнай графікай.

Асноўную частку акна рэдактара Inkscape займае **палатно**, на якім карыстальнік стварае і рэдагуе відарысы. На палатне вылучана **старонка**.

Перамяшчацца па палатне можна з дапамогай палос пракруткі. Павелічэнне або змяншэнне маштабу старонкі ажыццяўляецца з дапамогай клавіш «+» ці «−» на клавіятуры. Межы адлюстраванай на палатне старонкі вызначаюць межы відарыса для друку або захоўвання.

Шмат якія дзеянні ў рэдактары Inkscape можна выканаць некалькімі спосабамі:

- праз пункты меню;
- пры дапамозе кнопак на панэлях;
- з дапамогай камбінацый клавіш.

Праз меню **Правка** можна адмяніць апошняе дзеянне і паўтарыць адмененае дзеянне. Таксама даступная гісторыя дзеянняў.

Запуск праграмы ажыццяўляецца з дапамогай меню **Пуск: Все программы** → **Inkscape** — або двайной пстрычкай па ярлыку  на Рабочым стале. Для кожнага дакумента рэдактар Inkscape адкрывае асобнае акно.

Каб захаваць відарыс у рэдактары Inkscape, у пункце меню **Файл** трэба выбраць **Сохранить как** і націснуць кнопку **Сохранить**. Файл будзе захаваны ва ўласным фармаце рэдактара Inkscape — .svg. Відарыс, які мае дадзены фармат, неабходна адкрываць самім рэдактарам Inkscape, а выкарыстанне іншых праграм можа прывесці да некарэктнага выніку. Асноўныя фарматы файлаў, з якімі можна працаваць у Inkscape, пералічаны ў прыкладзе 23.2. Як захаваць відарыс у іншым фармаце, апісана ў прыкладзе 23.3.

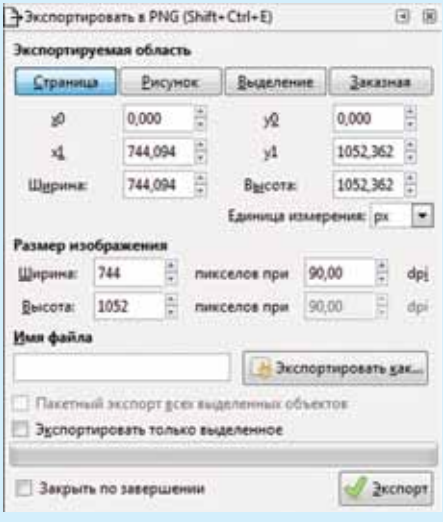
Для загрузкі відарыса ў пункце меню **Файл** трэба выбраць **Открыть**.

Падрабязней пра элементы інтэрфейсу вектарнага рэдактара Inkscape можна даведацца з матэрыялаў *Дадатку* (гл. с. 170).

Прыклад 23.3. Для захавання якога-небудзь відарыса ў фармаце, які адрозніваецца ад уласнага, неабходна пры захаванні выбраць адпаведны тып файла.

Каб захаваць рысунак у фармаце .png, трэба адкрыць акно «Экспортировать в PNG». Для гэтага неабходна:

1) у пункце меню **Файл** выбраць **Экспортировать в PNG**:



2) у акне **Экспортировать в PNG** націснуць кнопку 

; у акне, якое адкрыецца, выбраць папку для захавання файла і ўвесці імя;

3) у акне **Экспортировать в PNG** націснуць кнопку 

.

Можна вылучыць частку відарыса, і ў файл PNG будзе захавана толькі вылучанае.

1. Як запусціць вектарны рэдактар Inkscape?
2. Які фармат файла з'яўляецца ўласным фарматам рэдактара Inkscape?
3. Як павялічыць (паменшыць) маштаб старонкі?
4. З файламі якіх фарматаў можа працаваць вектарны рэдактар Inkscape?
5. Як захаваць частку відарыса ў файле фармату PNG?

Практыкаванні

1 Загрузіце вектарны рэдактар Inkscape. Адкрыйце ўбудаваны ў яго падручнік. Для гэтага ў галоўным меню выканайце каманду **Справка** → **Учебник** → **Inkscape: Основы**. Кіруючыся матэрыялам раздзелаў **Перемещение по холсту** і **Изменение масштаба**, выканайце:

1. Вертыкальнае перамяшчэнне па палатне; гарызантальнае перамяшчэнне па палатне.
2. Павелічэнне і памяншэнне маштабу палатна.

2 Унясіце змяненні ў інтэрфейс рэдактара.

1. Змяніце арыентацыю палатна на Альбом. Для гэтага выканайце каманду галоўнага меню рэдактара **Файл** → → **Свойства документа.** У дыялогавым акне **Свойства документа** ва ўкладцы **Страница** выберыце:

2. Адлюструйце сетку. Для гэтага выканайце каманду галоўнага меню рэдактара **Вид** → **Сетка страницы.**
3. Схавайце лінейкі і палітру колераў, выканаўшы каманду галоўнага меню рэдактара **Вид** → **Показать или скрыть** і зняўшы «птушкі» ў адпаведных пунктах. Паўтарыўшы каманду, вярніце адлюстраванне лінеек і палітры колераў.
4. Прагледзьце гісторыю выкананых вамі дзеянняў з дапамогай каманды галоўнага меню рэдактара **Правка** → → **История действий.**

§ 24. Стварэнне і рэдагаванне вектарнага відарыса

Работу ў вектарных графічных рэдактарах можна ўявіць як стварэнне фігур (аб'ектаў) патрэбнай формы і заданне ім пэўных залівак і абводак (абрысаў) (прыклад 24.1).

Разгледзім тэхналогію стварэння фігур у рэдактары Inkscape.

Пачаць работу над стварэннем вектарнага відарыса неабходна з вызначэння:

1) набору фігур, з якіх будзе складацца відарыс;

2) стылю і колеру абводкі;

3) стылю заліўкі;

4) узаемнага размяшчэння фігур.

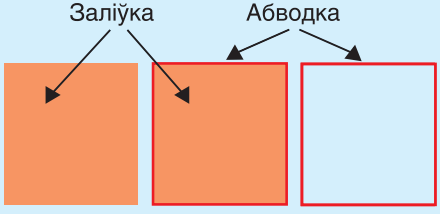
24.1. Стварэнне фігур

Прамавугольнікі ствараюць з дапамогай інструмента «Прамавугольнік»: . Прамавугольнік будуюць, перамяшчаючы мыш па палатне і ўтрымліваючы левую кнопку мышы. Калі пры гэтым утрымліваць клавішу Ctrl клавіятуры, то атрымаецца квадрат. Калі пры рысаванні прамавугольніка ўтрымліваць клавішу Shift, то пачатковы пункт будзе адпавядаць не вяршыні, а цэнтру прамавугольніка (прыклад 24.2).

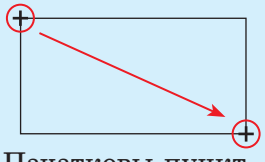
Прыклад 24.1. Асноўныя інструменты стварэння фігур у вектарным рэдактары Inkscape:

	Прамавугольнікі і квадраты
	Кругі, эліпсы і дугі
	Зоркі і многавугольнікі

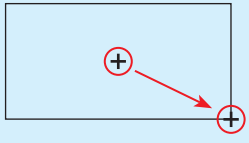
Узоры заліўкі і абводкі:



Прыклад 24.2. Рысаванне прамавугольніка.

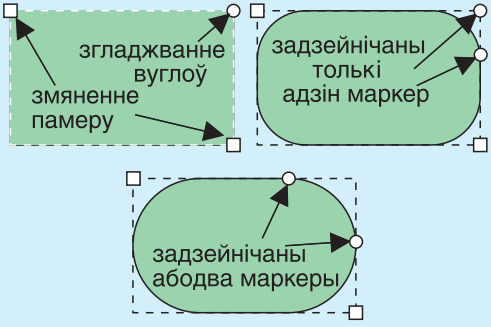


Пачатковы пункт — канец дыяганалі

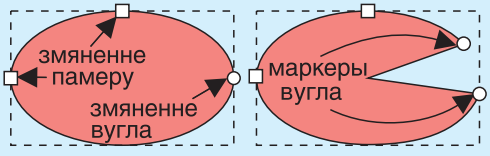


Пачатковы пункт — цэнтр прамавугольніка (пры націснутай клавішы Shift)

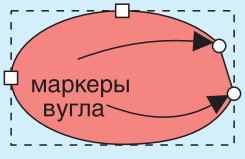
Прыклад 24.3. Выкарыстанне інструмента «Прамавугольнік».



Прыклад 24.4. Выкарыстанне інструмента «Кругі, эліпсы і дугі».



Калі пацягнуць бачны круглы маркер уніз, то атрымаецца сектар і з'явіцца іншы маркер, які можна перамяшчаць уверх.



Калі паспрабаваць перамясціць круглы маркер унутр эліпса, то рэжым рысавання сегмента зменіцца на дугу. Для таго каб зноў пераключыцца на сегмент, трэба перамясціць адзін з круглых маркераў за межы контуру эліпса.

Квадратныя маркеры ў процілеглых вуглах служаць для змянення памеру прамавугольніка, а круглы маркер у верхнім правым вугле кіруе згладжваннем вуглоў (прыклад 24.3).

Згладжванне (закругленне) вугла — замена вугла на частку акружнасці або эліпса, упісанага ў гэты вугал.

Замяніць вугал часткай акружнасці можна, перацягнуўшы круглы маркер уніз. Пры гэтым з'яўляецца другі маркер. Каб замяніць вугал часткай эліпса, трэба перамяшчаць абодва маркеры.

Зняць згладжванне можна, націснуўшы на кнопку  на панэлі ўласцівасцей інструмента «Прамавугольнік» (кнопка з'яўляецца, калі інструмент актыўны). На гэтай жа панэлі можна задаць шырыню і вышыню прамавугольніка з дапамогай лікавых параметраў.

Для рысавання эліпсаў і кругоў выбіраюць інструмент «Кругі, эліпсы і дугі»:  . Эліпсы і кругі можна пераўтварыць у сегменты і дугі (прыклад 24.4). Рысуюцца эліпс і круг гэтак жа, як і прамавугольнік. Пры ўтрыманні клавішы Shift пачатковы пункт рысавання будзе адпавядаць цэнтру эліпса.

Для рысавання зорак і многавугольнікаў выкарыстоўваюць інструмент «Зоркі і многавугольнікі»: (прыклад 24.5). Пачатковы пункт адпавядае цэнтру зоркі, канчатковы — адной з яе вяршынь. Вызначыць тып фігуры (зорка ці многавугольнік) можна толькі на панэлі ўласцівасцей інструмента, на якой ёсць два значкі з адпаведным рысункам многавугольніка і зорачкі:

Там жа можна змяняць і колькасць вуглоў. Па змаўчанні рысуюцца пяцівугольныя зоркі.

Для змянення зоркі ці многавугольніка ёсць два маркеры вузлоў. Верхні вузел служыць для кіравання становішчам канца зоркі або вуглом многавугольніка. Другі маркер (базавы вузел) кантралюе становішча ўнутранай вяршыні зоркі.

Для змянення формы зоркі выкарыстоўваюцца клавішы:

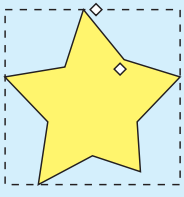
- Shift — кіруе згладжваннем вуглоў зоркі (прыклад 24.6);
- Alt — усе вяршыні зоркі або многавугольніка будуць перамяшчацца ў адвольным парадку (прыклад 24.7);
- Ctrl — перамяшчае вузлы зоркі або многавугольніка строга па прамой лініі.

Зорка — від плоскіх нявыпуклых многавугольнікаў, які не мае адназначнага матэматычнага азначэння. Правільнай называецца зорка, у якой усе ўнутраныя вуглы роўныя і ўсе знешнія вуглы роўныя. Словы, якія заканчваюцца на *-грама*, абазначаюць *n*-канцовую зорку (зорчаты многавугольнік). Актаграма — васьміканцовая зорка, гексаграма — шасціканцовая і г. д.

Прыклад 24.5. Выкарыстанне інструмента «Зоркі і многавугольнікі» для рысавання.

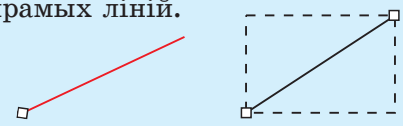
Прыклад 24.6. Выкарыстанне інструмента «Зоркі і многавугольнікі» для згладжвання вуглоў.

Прыклад 24.7. Выкарыстанне інструмента «Зоркі і многавугольнікі» для перамяшчэння вяршынь.



Прыклад 24.8. Выкарыстанне інструмента «Аловак».

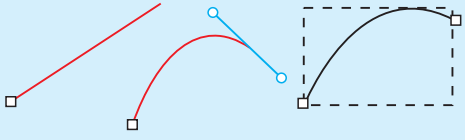
Прыклад 24.9. Рысаванне прамых ліній.



Пачатак рысавання

Заканчэнне рысавання

Прыклад 24.10. Рысаванне крывых ліній.



1. У пачатку лініі пстрыкнуць па ёй левай кнопкай мышы. За курсорам будзе цягнуцца прамая. Націснуць на левую кнопку мышы ў пункце, дзе павінен знаходзіцца канец лініі.

2. Не адпускаючы левую кнопку мышы, перанесці курсор убок. З'явіцца прамая лінія, да якой як датычная будзе прылягаць крывая.

3. Адпусціць левую кнопку мышы і пстрыкнуць правай. З'явіцца крывая, а прамая знікне. Можна працягваць праводзіць лінію, не націскаючы на правую кнопку мышы.

Для рысавання адвольных крывых выкарыстоўваецца інструмент «Аловак»:  . Каб яго выкарыстаць, трэба пстрыкнуць у пачатковым пункце на палатне і, не адпускаючы кнопку мышы, цягнуць паказальнік, праводзячы лінію.

Каб атрымаць замкнуты контур, дастаткова вярнуцца да першапачатковага пункта. Калі паказальнік мышы набліжаецца да яго, пункт змяняе колер на чырвоны. У гэты момант пункт мышы неабходна адпусціць.

Для інструмента «Аловак» устанаўліваецца ступень згладжвання:  . Чым вышэйшае значэнне згладжвання, тым больш гладкай будзе крывая (прыклад 24.8).

Для стварэння крывых і прамых ліній прызначаны інструмент «Крывая Безье і прамыя лініі»:  . Рысуючы прамую, трэба пстрыкнуць левай кнопкай мышы па месцы, дзе павінен знаходзіцца пачатковы пункт. Затым неабходна адвесці курсор у іншае месца. Пры гэтым за курсорам будзе ісці прамая лінія ад яе пачатку. Завяршаецца рысаванне лініі пстрычкай правай кнопкай мышы або двайной пстрычкай левай кнопкай мышы ў пункце заканчэння лініі. Лінія

зменіць чырвоны колер на чорны, і на яе канцы з'явіцца квадратны маркер (прыклад 24.9).

Стварэнне крывой лініі апісана ў прыкладзе 24.10.

24.2. Рэдагаванне фігур

Інструментам «Вылучыць і трансфармаваць» ( ) вылучаюць фігуры (прыклад 24.11). Вылучаныя фігуры можна: выдаляць клавішай Delete; перамяшчаць у рабочай вобласці (з дапамогай Ctrl строга па гарызанталі або па вертыкалі); маштабаваць (Ctrl дазваляе захаваць прапорцыі); адлюстроўваць гарызантальна (клавіша H) і вертыкальна (клавіша V); паварочваць; нахіляць.

Пасля вылучэння вакол фігуры з'яўляюцца восем двунакіраваных стрэлак, і аб'ект можна выдаляць, перамяшчаць і маштабаваць.

Група фігур вылучаецца пры націснутай клавішы Shift. Пстрычка па вылучаным у групе аб'екце выключае яго з вылучэння. Esc здымае ўсякае вылучэнне. Ctrl + A вылучае ўсе аб'екты.

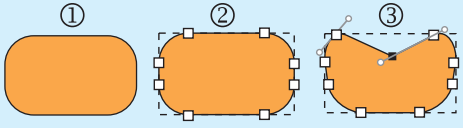
Форму аб'екта мяняюць з дапамогай інструмента «Рэдагаваць вузлы контуру»: (прыклад 24.12).

П. Безье распрацаваў метад вычэрчвання крывых у 1960 г., працуючы ў аўтамабілебудаўнічай фірме «Рэно». Адначасова гэта ж распрацоўка ўзнікла падчас даследавання П. дэ Кастэльжо з кампаніі «Сітраэн».

Прыклад 24.11. Выкарыстанне інструмента «Вылучыць і трансфармаваць».

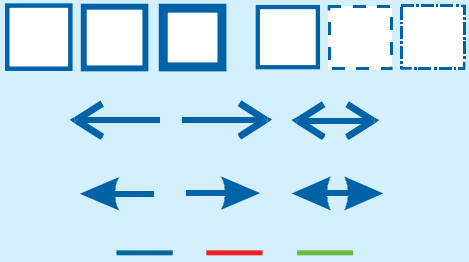
Калі выканаць пстрычку па ўжо вылучанай фігуры, выгляд стрэлак зменіцца:

Прыклад 24.12. Выкарыстанне інструмента «Рэдагаваць вузлы контуру».

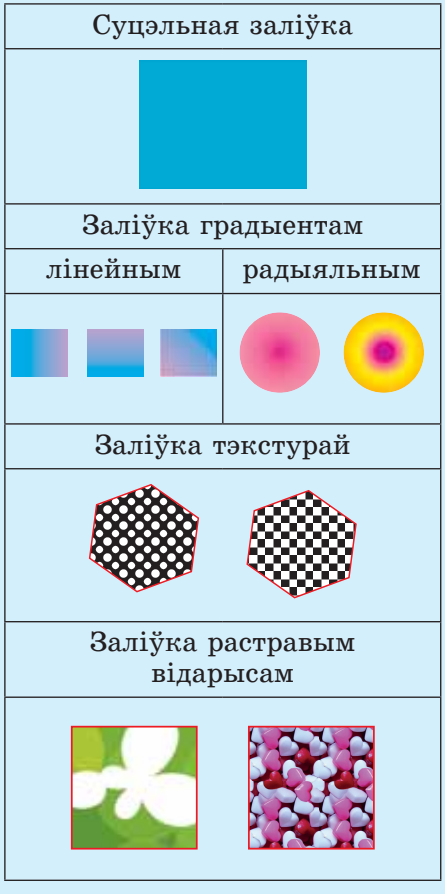


1. Нарысаваць графічны прымітыў.
2. Выбраць інструмент і настройку (пераўтварыць у контур).
3. Перамяшчаючы вузлы, змяніць контур. Пры неабходнасці дабавіць ( ) або выдаліць ( ) вузлы.

Прыклад 24.13. Стылі абводкі фігур і стрэлкі.



Прыклад 24.14. Спосабы заліўкі.



24.3. Абводка і заліўка

Абводка — лінія, якая абмяжоўвае фігуру.

Для абводкі можна вызначаць:

1) таўшчыню;

2) стыль (суцэльны, пункцірны, штрыхпункцірны);

3) стрэлкі (маркеры);

4) колер.

(Разгледзьце прыклад 24.13.)

Заліўкай называецца афарбоўка ўнутранай вобласці фігуры.

Заліўка фігуры можа быць зроблена адным з ніжэйпералічаных спосабаў:

1) суцэльным колерам;

2) градыентам;

3) тэкстурай (выбіраецца з набору дэкаратыўных залівак, у якіх выкарыстоўваюцца загадзя створаныя аўтарамі праграмы загатоўкі);

4) растравым відарысам з файла.

(Разгледзьце прыклад 24.14.)

Колеравы градыент — плаўны пераход аднаго колеру ў іншы.

У рэдактары Inkscape змяніць абводку і заліўку фігур можна ў дыялогавым акне **Заливка и**

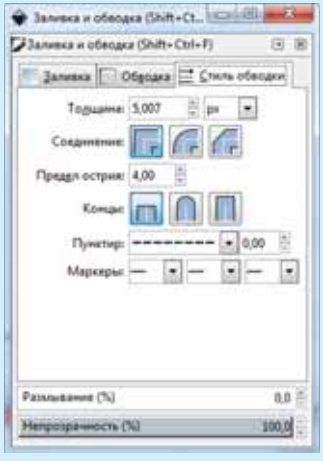
обводка або з дапамогай палітры. Акно адкрываецца па камандзе галоўнага меню рэдактара **Объект → Заливка и обводка** (прыклад 24.15). У гэтым акне можна выбраць колеравую мадэль і настроіць колер і яго празрыстасць.

У палітры колер заліўкі змяняецца пры дапамозе пстрычкі мышшу па патрэбным колеры. Для змянення колеру абводкі пстрычку неабходна рабіць пры націснутай клавішы Shift.

Аб'екту можна назначыць адсутнасць абводкі і заліўкі з дапамогай чырвонага крыжыка ў левай частцы палітры. Градыентную заліўку можна выкарыстоўваць і ў дачыненні да абводкі.

Пры пераходзе ад суцэльнай заліўкі да градыента ствараецца настройка градыента. Толькі што створаная настройка градыента выкарыстоўвае папярэдні колер суцэльнай заліўкі фігуры, які пераходзіць з непразрыстага ў празрысты.

Прыклад 24.15. Акно «Заливка и обводка».



Змяненне стылю абводкі

Змяненне заліўкі

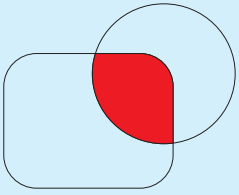
24.4. Работа з колерам

Задаваць колер у акне **Заливка и обводка** зручна, мяняючы лічбавыя значэнні асноўных колераў (прыклад 24.16).

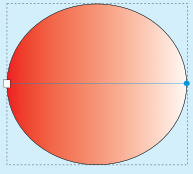
Прыклад 24.16. Настройка колеру ў колеравай мадэлі RGB:

Лічбавыя значэнні	Колер
R = 115, G = 200, B = 155	

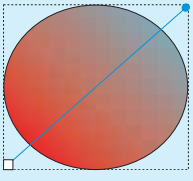
Прыклад 24.17. Выкарыстанне інструмента «Заліўка».



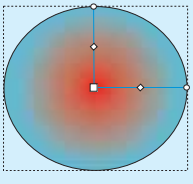
Прыклад 24.18. Выкарыстанне інструмента «Градыент».



Для лінейнага градыента ўсе апорныя пункты размешчаны на адной прамой.



Апорныя пункты могуць размяшчацца і за межамі аб'екта.



Радыяльны градыент мае цэнтральны апорны пункт, які з'яўляецца пачатковым. З яго выходзяць дзве лініі градыента.

Для залівання замкнутай вобласці выкарыстоўваецца інструмент «Заліўка»:  (прыклад 24.17).

Пры рысаванні часта ўзнікае неабходнасць выбару колеру на ўчастках палатна. Для гэтага выкарыстоўваецца інструмент «Піпетка»:  .

Для стварэння градыентных залівак прызначаны інструмент «Градыент»:  . Колеры градыента настройваюцца пры дапамозе апорных пунктаў (прыклад 24.18). Колькасць пунктаў адпавядае колькасці колераў у градыенце. Апорныя пункты можна дабаўляць або выдаляць, папярэдне выбраўшы  або  на панэлі інструмента. Дабаўляецца апорны пункт двайной пстрычкай мышы па лініі градыента. Выдаляецца вылучаны апорны пункт з дапамогай клавішы Delete.

Стан апорнага пункта пры пэўным колеры:

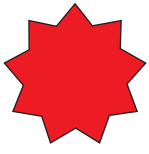
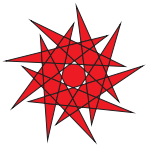
- **белы** — невылучаны (неактыўны);
- **сіні** — вылучаны (актыўны, выбраны);
- **чырвоны** — наведзены (пад паказальнікам мышы) або змяняемы (калі выконваецца дзеянне).

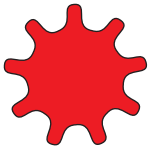
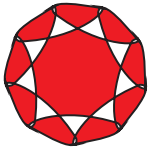
1. З дапамогай якой клавішы можна нарысаваць правільную фігуру?
2. Як згладзіць вуглы прамавугольніка ў рэдактары Inkscape?
3. Якія фігуры можна нарысаваць з дапамогай інструмента ?
4. Для чаго служыць інструмент ?
5. Які інструмент дазваляе рэдагаваць вузлы фігуры?
6. Што называецца абводкай вектарнага відарыса?
7. У чым складаецца працэс заліўкі фігуры?
8. Што такое колеравы градыент?
9. Як настроіць градыент у рэдактары Inkscape?

Практыкаванні

1 Нарысуйце некалькі розных фігур і выкарыстайце ў дачыненні да іх маштабаванне, адбіццё, паварот, нахіл.

2 Стварыце відарысы зорак.

№	Відарыс	Прыёмы стварэння і рэдагавання
1		Інструмент  
2		Выканаць п. 1 і змяніць форму, перамяшчаючы верхні вузел зоркі да цэнтра
3		Выканаць п. 1 і змяніць форму, перамяшчаючы базавы вузел зоркі ўправа
4		Выканаць п. 1 і змяніць форму, перамяшчаючы базавы вузел зоркі ўлева і ўверх

5		Выканаць п. 1 і змяніць форму, перамяшчаючы верхні вузел зоркі пры націснутай клавішы Alt
6		Выканаць п. 1 і змяніць форму, перамяшчаючы верхні вузел зоркі ўправа пры націснутай клавішы Shift
7		Выканаць п. 1 і змяніць форму, перамяшчаючы верхні вузел зоркі ўлева пры націснутай клавішы Shift

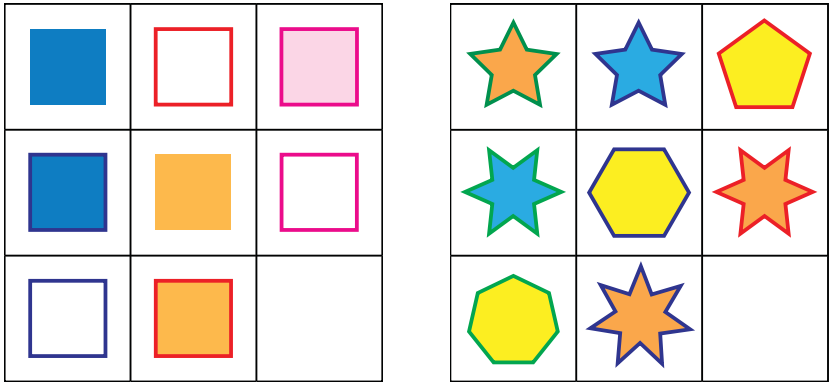
3 Нарысуйце пяць фігур: прамавугольнік, квадрат, авал, круг і правільны шасцівугольнік. Выкарыстайце ў дачыненні да кожнай фігуры адпаведныя заліўкі: суцэльную (з празрыстасцю і без), градыент (лінейны і радыяльны), з ужываннем тэкстуры.

4 Выканайце прыклад:

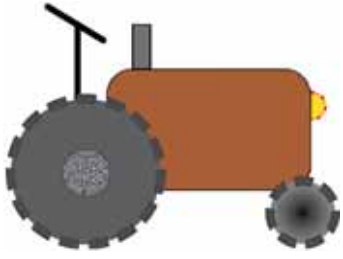
Нарысуйце малінавы квадрат без абводкі. На ўкладцы **Заливка** ўстанавіце R = 255, G = 0, B = 225, A = 255, размываннe — 0, непразрыстасць — 55. На ўкладцы **Обводка** — без абводкі (×).

Нарысуйце фіялетавы прамавугольнік з чорнай абводкай так, каб ён перакрываў частку квадрата. На ўкладцы **Заливка** ўстанавіце R = 100, G = 0, B = 105, A = 255, размываннe — 0, непразрыстасць — 100. На ўкладцы **Стиль обводки** — таўшчыня 15 px.

5 Адкрыйце файл і дапоўніце відарысы фігурамі, якіх не хапае.



6 Стварыце відарыс:

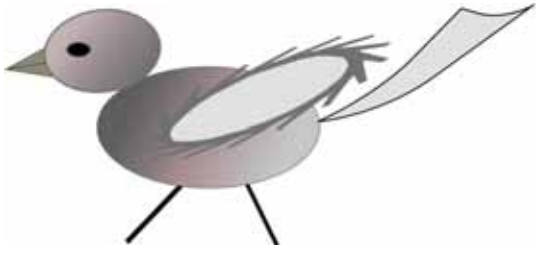


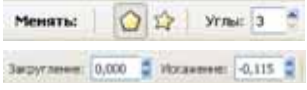
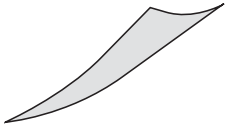
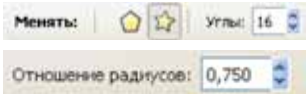
У гэтым вам дапаможа табліца.

Элемент відарыса	Інструмент і настройкі	Абводка	Заліўка
		Суцэльная, колер чорны	Аднародная, колер цёмна-шэры
	 пера-ключыцца на 	Штрыхпункцірная, колер чырвоны, таўшчыня 4 px	Аднародная, колер жоўты (FFFF00)

Элемент відарыса	Інструмент і настройкі	Абводка	Заліўка
		Суцэльная, колер чорны	Аднародная, колер карычневы (A05A2C)
		Пункцірная, колер чорны, таўшчыня 15 px	Градыент радыяльны (пераход ад чорнага да шэрага)
	 (рэжым — прамыя лініі)	Суцэльная, колер чорны	Няма
		Суцэльная, колер чорны, канцы круглыя	
		Пункцірная, колер чорны, таўшчыня 15 px	Аднародная, колер шэры (4D4D4D)
		Суцэльная, колер чорны	Тэкстура «пясок»

7 Стварыце відарыс:



<table>
<tr><th>Элемент відарыса</th><th>Інструмент і настройкі</th><th>Абводка</th><th>Заліўка</th></tr>
<tr><td>  </td><td rowspan="3">  </td><td rowspan="3">Суцэльная, колер шэры, таўшчыня 3 px</td><td>Лінейны градыент, два колеры</td></tr>
<tr><td>  </td><td>Лінейны градыент, тры колеры</td></tr>
<tr><td>  </td><td>Аднародная, колер чорны</td></tr>
<tr><td>  </td><td>Для трохвугольніка:
 
 
Для лініі: 
(рэжым — прамыя лініі)</td><td>Для трохвугольніка: суцэльная, колер шэры, таўшчыня 3 px</td><td>Для трохвугольніка: аднародная, колер 918A6F</td></tr>
<tr><td>  </td><td> 
 
Змяніць форму:
  </td><td>Суцэльная, колер чорны, таўшчыня 3 px</td><td>Аднародная, колер шэры</td></tr>
<tr><td>  </td><td> 
 
Змяніць становішча маркераў вузлоў</td><td>Суцэльная, колер цёмна-шэры, таўшчыня 7 px</td><td>Аднародная, колер шэры</td></tr>
</table>

§ 25. Аперацыі над аб’ектамі вектарнага відарыса

25.1. Капіраванне, выраўніванне і ўзаемнае размяшчэнне аб’ектаў

Стварыць копію аб’екта ў рэдактары Inkscape можна, як і ў іншых праграмах, выкарыстаўшы буфер абмену. У гэтым выпадку копія з’явіцца побач з арыгіналам. У рэдактары Inkscape ёсць дадатковыя аперацыі капіравання аб’ектаў — дубліраванне і кланіраванне (прыклад 25.1).

Прыклад 25.1. Дубліраванне і кланіраванне фігур.

Арыгінал Дубль Клон

Пасля рэдагавання арыгінала атрымаем:

Арыгінал Дубль Клон

Каб стварыць дубль аб’екта, трэба выбраць пункт меню **Правка** → **Продублировать** або націснуць  на панэлі інструментаў. Створаная копія змяшчаецца зверху арыгінала. Як бачна з прыкладу 25.1, **дубль** з’яўляецца самастойным аб’ектам. Пры змяненні арыгінала ён не мяняецца.

Клон ствараецца па камандзе **Правка** → **Клоны** → **Создать клон** або з дапамогай кнопкі  на панэлі інструментаў. Пры стварэнні клон, як і дубль, змяшчаецца над арыгіналам. **Клоны** змяняюцца разам са змяненнем арыгінала.

Прыклад 25.2. Стварэнне дубля і клона.

Для стварэння дубляў і клонаў можна выкарыстоўваць інструмент «Распыляльнік»:  . Гэты інструмент рысуе аб’ектамі з буфера абмену. Распыляльнік выконваецца ў трох рэжымах:

 — стварае дублі аб’ектаў;

 — стварае клоны аб’ектаў;

 — усе распыленыя аб’екты аб’ядноўваюцца ў адзін аб’ект.

Яшчэ адзін спосаб стварэння дубля або клона, акрамя вышэйпералічаных, прыведзены ў прыкладзе 25.2.

Для размяшчэння пэўным чынам адносна адзін аднаго аб’ектаў вектарнага відарыса выкарыстоўваюцца аперацыі выраўноўвання.

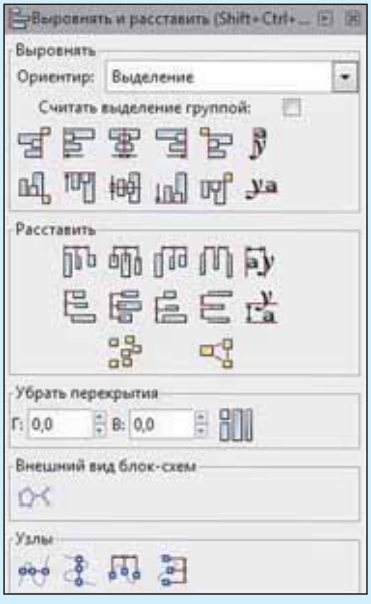
Выраўноўваннем называецца такое размяшчэнне ўсіх вылучаных аб’ектаў, пры якім вызначаныя пункты аб’ектаў размяшчаюцца на адной прамой.

Выраўноўванне аб’ектаў можна ажыццявіць з дапамогай акна **Выровнять и расставить.** Акно можна адкрыць, выканаўшы каманду **Объект** → **Выровнять и расставить** або націснуўшы на кнопку **Выровнять и расставить объекты** ( ) на панэлі інструментаў (прыклад 25.3).

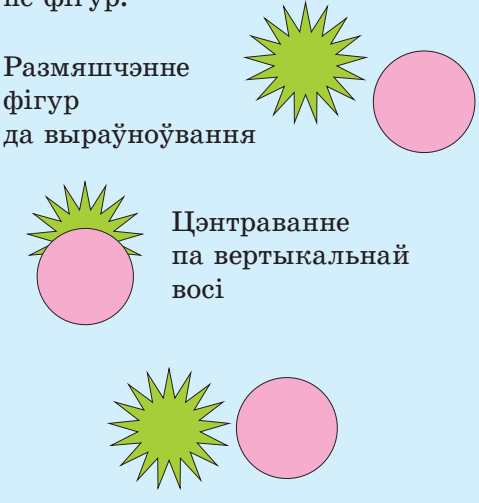
На панэлі **Выровнять** у першым радку выбіраецца спосаб выраўноўвання фігур па вертыкалі, а ў другім — па гарызанталі (прыклад 25.4).

На панэлі **Расставить** вызначаецца спосаб размеркавання аб’ектаў на некаторай адлегласці адзін ад аднаго. У першым радку паказваецца размеркаванне аб’ектаў

Прыклад 25.3. Акно **Выровнять и расставить.**



Прыклад 25.4. Выраўноўванне фігур.

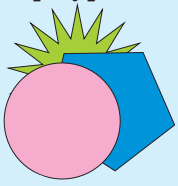


Прыклад 25.5. Размеркаванне ліній.

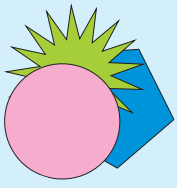
Размяшчэнне ліній да расстаноўкі

Адлегласці паміж лініямі выраўнаваны па гарызанталі

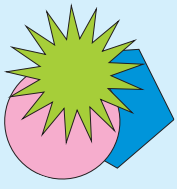
Прыклад 25.6. Парадак размяшчэння фігур.



Змяняем становішча салатавай фігуры:



Падняць



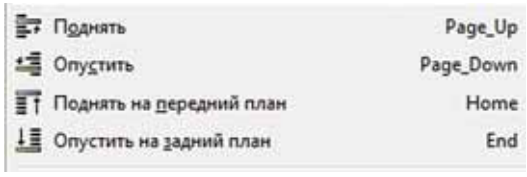
Падняць на пярэдні план

Групоўка прадухіляе выпадковыя змяненні становішча аб'екта адносна іншых аб'ектаў. Пры групоўцы можна ствараць укладзеныя групы, групуючы наяўныя групы.

Аперацыі аб'яднання і перасячэння аб'ектаў дазваляюць ствараць розныя няправільныя формы.

па вертыкалі, а ў другім — па гарызанталі (прыклад 25.5).

Для змянення **парадку размяшчэння** аб'ектаў іх неабходна вылучыць. Затым выкарыстаць каманды меню **Объект:**



Каманды **Поднять на передний план** і **Опустить на задний план** паставяць вылучаныя аб'екты на самую верхнюю або самую ніжнюю пазіцыю. Дзве іншыя каманды — **Поднять** і **Опустить** — апусцяць ці падніму ць вылучаныя аб'екты на адзін узровень адносна найбліжэйшага нявылучанага аб'екта (прыклад 25.6).

25.2. Групоўка. Аб'яднанне і перасячэнне аб'ектаў

У Inkscape некалькі аб'ектаў можна аб'яднаць у групу. Пры перамяшчэнні і трансфармацыі група паводзіць сябе як адзін аб'ект.

Каб згрупаваць некалькі аб'ектаў, трэба вылучыць іх усе і выбраць у меню **Объект** → → **Сгруппировать** або націснуць Ctrl + G. Каб разгрупаваць адну

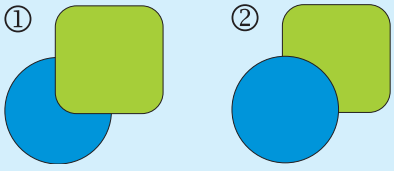
або некалькі груп, трэба выбраць у меню **Объект → Разгруппировать** ці выканаць двайную пстрычку мышшу па групе. Самі групы таксама можна аб’ядноўваць у групы. Падвойная пстрычка адмяняе толькі групаванне верхняга ўзроўню.

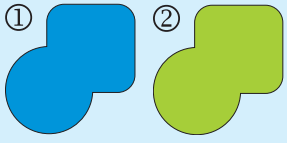
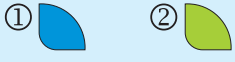
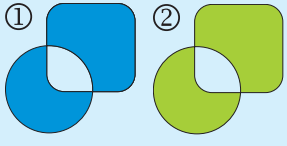
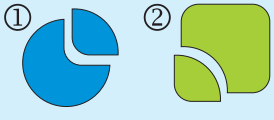
Не абавязкова разбіваць групу, калі трэба адрэдагаваць адзін аб’ект з яе. Дастаткова пстрыкнуць па аб’екце з націснутай Ctrl (або Shift + Ctrl, калі трэба адабраць некалькі аб’ектаў), і можна будзе працаваць з аб’ектам у групе асобна.

Пры стварэнні відарысаў часта ўзнікае неабходнасць выканаць над аб’ектамі лагічныя аперацыі аб’яднання і перасячэння. У Inkscape гэтыя аперацыі выконваюцца праз меню **Контур**:

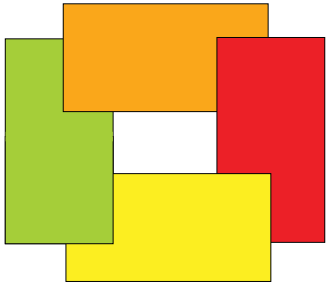
У прыкладзе 25.7 паказаны вынік выкарыстання некаторых аперацый аб’яднання для двух аб’ектаў, размешчаных у розным парадку.

Прыклад 25.7. Выкарыстанне аперацый аб’яднання.



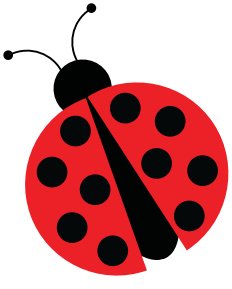
Аперацыя	Вынік аперацыі
Сума	
Рознасць	
Перасячэнне	
Выключэнне з АБО	
Падзяліць	

1. Якія аперацыі над аб'ектамі вектарнага відарыса можна выканаць у рэдактары Inkscape?
2. Што адбудзецца, калі ў дачыненні да аб'ектаў выкарыстаць спачатку выраўноўванне па вертыкальнай восі, а потым па гарызантальнай?
3. Што адбываецца пры групоўцы аб'ектаў?
4. Ці важна размяшчэнне аб'ектаў пры выкананні аперацый аб'яднання?
5*. Ці можна, мяняючы толькі ўзаемнае размяшчэнне аб'ектаў, стварыць наступны рысунак?



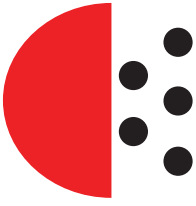
Практыкаванні

1 Стварыце відарыс божай кароўкі, як паказана на рысунку:

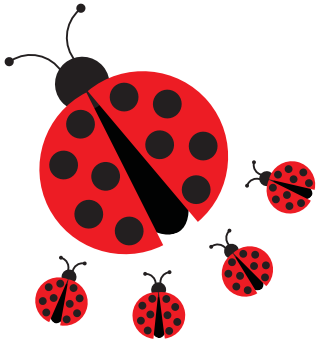


Для гэтага падрыхтуйце элементы рысунка, выкарыстоўваючы інструменты **Эліпс** і **Крывая Безье**:

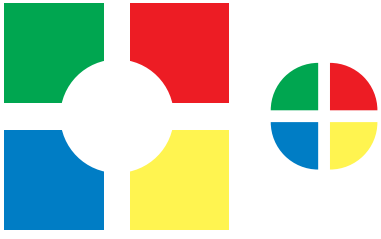
Выкарыстайце ў дачыненні да створаных элементаў аперацыі, напісаныя ў табліцы.

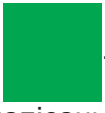
Зыходныя аб’екты	Аперацыі	Вынік
	Прадубліраваць аб’ект і паменшыць памер дубля. Стварыць 4 дублі зменена га аб’екта	
	Размясціць чорныя кругі ў чырвоным паўкрузе і згрупаваць аб’екты	
	Прадубліраваць аб’ект і паменшыць памер дубля	
	Злучыць чорны круг з дугой і згрупаваць аб’екты	
	Прадубліраваць аб’ект і адлюстраваць дубль па гарызанталі. Выканаць павароты	
		
	Прадубліраваць аб’ект і змяніць памеры і прапорцыі дубля. Выканаць паварот	

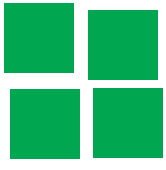
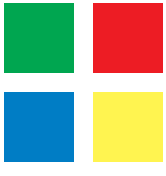
2 Выкарыстоўваючы рысунак з задання 1, стварыце відарыс:

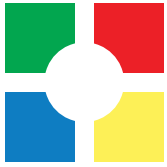


3* Стварыце відарыс, выкарыстоўваючы аперацыі над аб'ектамі:



Для гэтага стварыце элемент рысунка, ужываючы інструмент **Прамавугольнік**:  . Выкарыстайце ў дачыненні да створаных элементаў аперацыі, напісаныя ў табліцы.

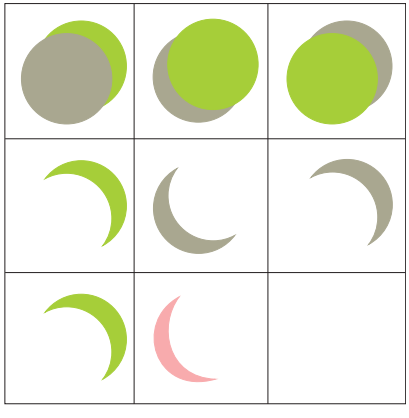
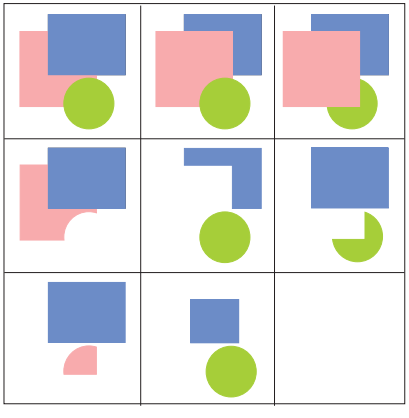
Аперацыі	Вынік
Дубліраванне + перамяшчэнне	
Выраўноўванне + змяненне колеру	

Аперацыі	Вынік
Нарысаваць белы круг, выраўнаваць па цэнтры і стварыць тры яго дублі	
Вылучаючы папарна адзін квадрат і круг, выкарыстаць аперацыю перасячэння	

4 Нарысуйце адзін з відарысаў:



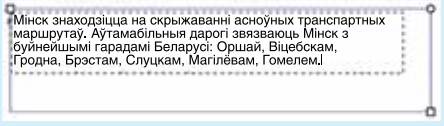

5* Адкрыйце файл і дапоўніце відарысы фігурамі, якіх не хапае:

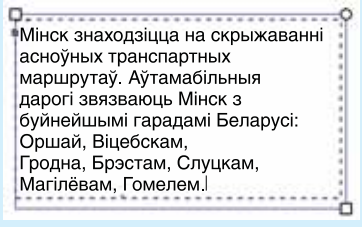



6 Выкарыстоўваючы відарысы з задання 4, прыдумайце і нарысуйце сюжэтны рысунак.

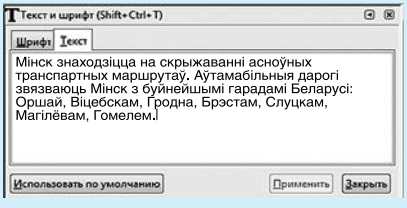
§ 26. Работа з тэкстам

Прыклад 26.1. Стварэнне простага тэксту.





Уводзіць тэкст зручна ў тэкставым рэдактары. Яго выклік выконваецца па камандзе **Текст** → → **Текст и шрифт**. Для ўводу тэксту неабходна перайсці на ўкладку **Текст**.



Тэкст можна ўставіць праз буфер абмену з іншага тэкставага рэдактара (Блакнота і інш.).

Просты тэкст можна пераўтварыць у мастацкі: **Текст** → → **Преобразовать в текст**. Перад пераўтварэннем тэкст павінен быць вылучаны. Калі пасля гэтага вылучыць тэкст, то тэкставая рамка з вузламі знікне. Гэта ўжо мастацкі тэкст.

Для стварэння тэксту выкарыстоўваецца інструмент «Тэкст»: A. У вектарных рэдактарах існуюць дзве магчымасці работы з тэкстам — просты тэкст і мастацкі тэкст.

Просты тэкст выкарыстоўваецца для адлюстравання на рысунках вялікіх тэкставых фрагментаў і можа падзяляцца на абзацы. Просты тэкст можна фармаціраваць гэтак жа, як і ў звычайным тэкставым рэдактары.

Для стварэння простага тэксту неабходна:

1) выбраць інструмент A;

2) устанавіць параметры тэксту на кантэкставай панэлі кіравання;

3) абрысаваць з дапамогай мышы тэкставую рамку і ўвесці ў яе тэкст.

Тэкставая рамка ўяўляе сабой суцэльную прамавугольную рамку. Тэкст у ёй абмяжоўваецца пункцірнай рамкай. Пры дапамозе маркера ў правым ніжнім вугле можна змяняць памеры тэкставай рамкі. Пры гэтым тэкст падганяецца пад памеры рамкі (прыклад 26.1).

Мастацкі тэкст выкарыстоўваецца для стварэння мастацкіх

надпісаў. Створаныя надпісы з'яўляюцца графічнымі аб'ектамі. Да іх можна ўжываць розныя эфекты, напрыклад змяніць форму літар, стварыць цень, размясціць тэкст па патрэбнай траекторыі. Для стварэння мастацкага тэксту неабходна пстрыкнуць мышшу ў любым пункце палатна. Пасля гэтага ў дадзеным пункце пачне міргаць курсор, запрашаючы да ўводу тэксту. Пераход на новы радок адбываецца пры націсканні на клавішу Enter.

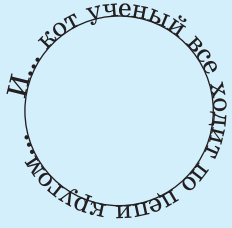
Пры любым спосабе стварэння тэксту ствараецца тэкставы аб'ект, які можна перамяшчаць, паварочваць і г. д. з дапамогай інструмента  .

Для рэдагавання тэкставага аб'екта неабходна выбраць інструмент  і пстрыкнуць па аб'екце.

Шмат цікавых магчымасцей работы з мастацкім тэкстам прапануе меню **Текст**. Разгледзім некаторыя з іх:

1. **Разместить по контуру** (прыклад 26.2). Перад выкарыстаннем гэтай каманды трэба вылучыць тэкст і контур.

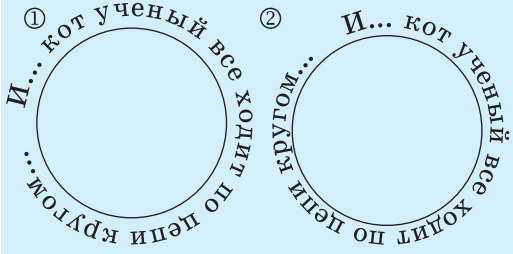
Прыклад 26.2. Размяшчэнне тэксту[1] па акружнасці.



Пасля звязвання тэксту і контуру можна рэдагаваць кожны з гэтых аб'ектаў, не парушаючы сувязі паміж імі:

1. Прыўзняць тэкст над контурам.

2. Павярнуць контур (пры гэтым тэкст паварочваецца разам з контурам).

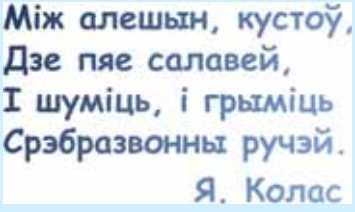


3. Зняць контур (зрабіць празрыстым).

[1] Тэкст у прыкладзе 26.2 цытуецца па: Пушкин, А. С. Руслан и Людмила : поэма. — М. : Изд. Дом «Прибой». — 1996. — С. 5.

Прыклад 26.3. Размяшчэнне тэксту ўнутры контуру.

Прыклад 26.4. Выкарыстанне градыента да тэксту[1].



Прыклад 26.5. Выкарыстанне эфекту да тэксту.

2. Заверстать в блок, г. зн. змясціць у контур (прыклад 26.3). Парадак работы з блокамі такі ж, як і з контурамі.

Тэкст можна размаляваць у розныя колеры (цалкам або часткова). Таксама да тэксту можна выкарыстаць градыент (прыклад 26.4). Пры гэтым у далейшым тэкст можна рэдагаваць звычайным чынам. Акрамя таго, літары тэксту можна пераўтварыць у контур, выканаўшы каманду **Контур → Окантурить объект.** Пасля гэтага тэкст ужо нельга рэдагаваць, але ў дачыненні да яго літар можна выкарыстоўваць разнастайныя эфекты. У прыкладзе 26.5 ужыта каманда **Расширения → Изменение контура → Дрожание узлов.**

?

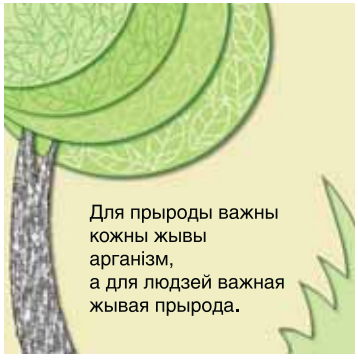
1. Якія магчымасці работы з тэкстам існуюць у рэдактары Inkscape?
2. Якія дзеянні можна выконваць з тэкставым аб'ектам?
3. Ці можна рэдагаваць мастацкі тэкст?
4. Якім чынам можна пераўтварыць літары тэксту ў контуры?
5. Ці можна рэдагаваць тэкст пасля пераўтварэння яго літар у контуры?
6. Якія дзеянні неабходна выконваць перад выкарыстаннем у дачыненні да тэксту эфектаў?

[1] Цытуецца па: Колас, Я. Ручэй // Лазарук, М. А., Логінава, Т. У. Беларуская літаратура : вучэб. дапам. для 7 кл. агульнаадукацыйных устаноў з беларус. і рус. мовай навучання. — Мінск : «Нацыянальны інстытут адукацыі». — 2010. — С. 28.

Практыкаванні

1 Нарысуйце паштоўку. Афармеце яе мастацкім тэкстам, размешчаным па контуры.

2 Нарысуйце плакат. Выкарыстайце розныя віды залівак і ўстаўце просты тэкст.

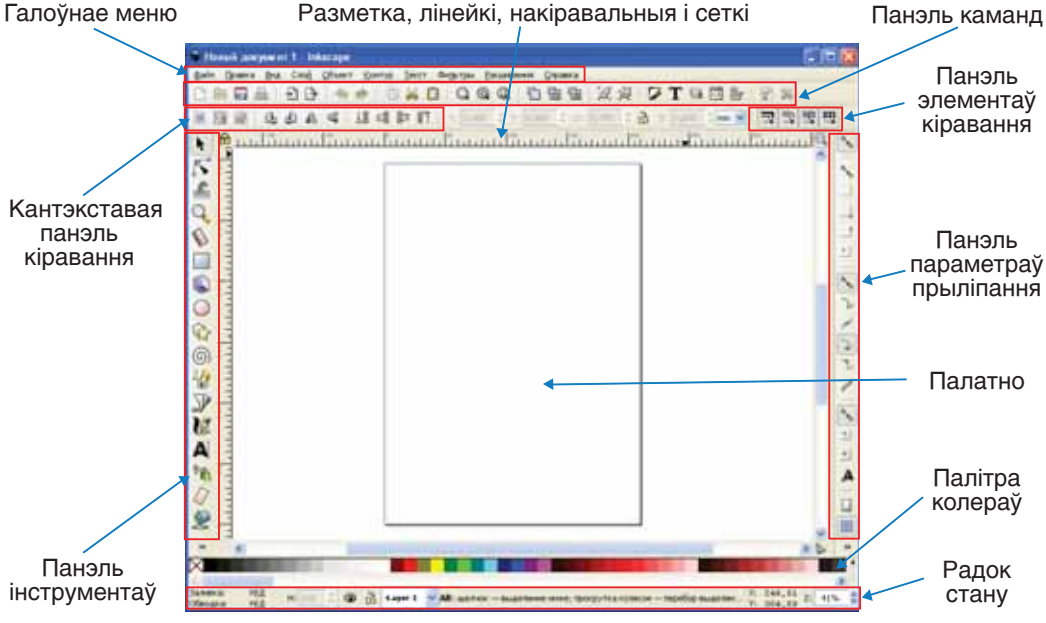


3 Прыдумайце і намалюйце плакат на адну з тэм:

1. «Мыйце рукі перад ядой».
2. «Маё шчаслівае дзяцінства».
3. «Мы ў адказнасці за тых, каго прыручылі».
4. «Беражыце электраэнергію».
5. «Правілы паводзін у камп'ютарным класе».

ДАДАТАК

Акно інтэрфейсу Inkscape можна падзяліць на дзесяць асноўных абласцей:



Галоўнае меню

Файл Правка Вид Слой Объект Контур Текст Фильтры Расширения Справка

Змяшчае асноўныя функцыі работы з праграмай: работа з файламі; функцыі рэдагавання і прагляду; функцыі рэдактара работы з тэкстам, фільтрамі, аб'ектамі і контурамі. Уключае дадаткі і даведачную інфармацыю.

Панэль каманд

Змяшчае значкі-іконкі, якія выклікаюць пэўныя каманды. Гэтыя каманды таксама даступныя ў галоўным меню або па камбінацыі клавіш. Панэль каманд прызначана для лёгкага

доступу да функцый рэдактара, якія найбольш выкарыстоўваюцца. З яе ў адзін клік можна адкрыць новы або існуючы дакумент, надрукаваць яго, загрузіць відарыс, адмяніць папярэднія каманды, маштабаваць, адкрыць дыялогавае акно для настройкі ўласцівасцей дакумента і г. д. Кожны значок пры навядзенні курсора мышы адлюстроўвае сваю функцыю з дапамогай падказак, якія пры гэтым усплываюць.

Калі ўсе значкі панэлі каманд не змяшчаюцца на экране, то доступ да іх можа быць ажыццёўлены праз кнопку з дзвюма стрэлачкамі ўнізе панэлі. У выніку пстрычкі па гэтай кнопцы ў выглядзе меню з'явяцца ўсе астатнія каманды панэлі, значкі якіх не змяшчаюцца на ёй.

Панэль інструментаў

Складаецца з вертыкальнага рада кнопак, размешчанага ў левай частцы акна рэдактара. Гэта асноўны элемент для работы ў вектарным рэдактары Inkscape. Ён змяшчае набор графічных інструментаў для стварэння і рэдагавання фігур.

У акне інструментаў прысутнічаюць:

- інструменты, прызначаныя для работы з геаметрычнымі фігурамі;
- інструменты, прызначаныя для свабоднай трансфармацыі фігур і ліній;
- інструменты, прызначаныя для работы з тэкстам;
- інструменты, прызначаныя для работы з колерам (заліўка і градыенты).

Кантэкставая панэль кіравання

У залежнасці ад таго, які інструмент выбраны ў акне інструментаў, выгляд кантэкставай панэлі змяняецца. У ёй адлюстроўваюцца настройкі і параметры актыўнага інструмента.

У залежнасці ад сітуацыі змяненне гэтых параметраў можа адразу паўплываць на выбраны аб'ект. Пры выкарыстанні інструментаў  ,  і  вярнуцца да зыходных параметраў можна з дапамогай кнопкі  кантэкставай панэлі гэтых інструментаў.

Палатно

Палатно, або канва, з'яўляецца галоўнай рабочай вобласцю праграмы. Гэта асноўная частка інтэрфейсу, паколькі менавіта тут карыстальнік стварае і рэдагуе рысункі. Размешчана палатно пасярэдзіне акна праграмы і падобна да відарыса чыстага белага ліста паперы.

Нягледзячы на тое што межы адлюстраванай на палатне старонкі вызначаюць межы відарыса для друку або захоўвання, пры рысаванні памер старонкі зусім не абмяжоўвае вобласць для відарыса. Вы можаце зрабіць межы старонкі і цені гэтых меж нябачнымі. Настроіць бачнасць меж старонкі можна ва ўласцівасцях дакумента.

Разметка, лінейкі, накіравальныя і сеткі

Лінейка разметкі размешчана ўверсе і злева ад палатна. Дзяленні лінейкі разметкі вызначаюць адлегласці ў некаторых адзінках (па змаўчанні ў пікселях). Змяніць настройку адзінак вымярэння можна ва ўласцівасцях дакумента (**Файл → Свойства документа**).

Калі паказальнік мышы знаходзіцца над палатном, на лінейцы з'яўляюцца два чорныя трохвугольнікі, якія адлюстроўваюць каардынаты курсора адносна ніжняга левага вугла старонкі. Гэтыя каардынаты X і Y можна ўбачыць у радку стану (у ніжнім правым вугле акна праграмы), побач з параметрам маштабу Z.

Камбінацыя клавіш Ctrl + R дазваляе схаваць або адлюстраваць лінейкі разметкі. Таксама гэта можна зрабіць у галоўным меню **Вид → Показать или скрыть → Линейки**.

Накіравальныя ствараюцца ў Inkscape карыстальнікам для палягчэння рысавання або для пабудовы фігур. Накіравальныя дазваляюць устанавіць становішча некаторых інструментаў дакладна па іх. Ужыванне накіравальных палягчае карыстальнікам выраўноўванне аб'ектаў, якія ствараюцца з дапамогай мышы. Каб устанавіць накіравальныя, пстрыкніце паказальнікам мышы на гарызантальнай або вертыкальнай лінейцы і, утрымліваючы кнопку мышы націснутай, перацягніце накіравальную, што з'явілася, у той пункт палатна, дзе яна павінна быць, пасля гэтага адпусціце кнопку мышы. З дапамогай гарызантальнай лінейкі ствараюцца гарызантальныя накіравальныя, з дапамогай вертыкальнай — вертыкальныя.

Сетка можа аказацца карыснай, каб не выкарыстоўваць вялікую колькасць накіравальных. Адлюстраваць сетку можна з дапамогай галоўнага меню **Вид → Сетка**.

Панэль параметраў прыліпання

Панэль параметраў прыліпання дазваляе лёгка наладзіць параметры прыліпання аб'екта. Функцыі гэтай панэлі зручныя для правільнага і дакладнага размяшчэння аб'ектаў. Панэль параметраў прыліпання размешчана вертыкальна па правым краі рабочай вобласці акна.

Палітра колераў

Забяспечвае хуткі доступ да колераў, яна ж дазваляе назначыць колеры для фігур. Адлюстроўваецца ў ніжняй частцы акна праграмы або можа быць адкрыта ў выглядзе асобнага акна

(**Вид** → **Образцы цветов** або камбінацыя клавіш Shift + + Ctrl + W).

Радок стану

Знаходзіцца ў самым нізе акна. Радок стану адлюстроўвае (злева направа):

- колер заліўкі і абводкі аб'екта;
- магчымасць хуткай работы са слаямі і пераключэння паміж імі;
- вобласць паведамленняў;
- індыкатар каардынат паказальніка мышы;
- кіраванне маштабам.

Вучэбнае выданне
Котаў Уладзімір Міхайлавіч
Лапо Анжаліка Іванаўна
Вайцеховіч Алена Мікалаеўна

ІНФАРМАТЫКА

Вучэбны дапаможнік для 7 класа
ўстаноў агульнай сярэдняй адукацыі
з беларускай мовай навучання

Заг. рэдакцыі *Г. А. Бабаева*. Рэдактар *К. І. Даніленка*. Мастацкія рэдактары *А. М. Багушэвіч, Н. І. Рэзановіч*. Вокладка *Н. У. Кузьмянковай*. Тэхнічнае рэдагаванне і камп'ютарная вёрстка *І. І. Дубройскай*. Карэктары *В. С. Бабеня, В. С. Казіцкая, А. П. Тхір, Г. В. Алешка*.

Падпісана ў друк 27.11.2017. Фармат $70\times90\ ^{1}/_{16}$. Папера афсетная. Гарнітура школьная. Друк афсетны. Умоўн. друк. арк. 12,87. Умоўн.-выд. арк. 8,8. Тыраж 15 800 экз. Заказ .

Выдавецкае рэспубліканскае ўнітарнае прадпрыемства «Народная асвета» Міністэрства інфармацыі Рэспублікі Беларусь. Пасведчанне аб дзяржаўнай рэгістрацыі выдаўца, вытворцы, распаўсюджвальніка друкаваных выданняў № 1/2 ад 08.07.2013. Пр. Пераможцаў, 11, 220004, Мінск, Рэспубліка Беларусь.

ААТ «Паліграфкамбінат імя Я. Коласа». Пасведчанне аб дзяржаўнай рэгістрацыі выдаўца, вытворцы, распаўсюджвальніка друкаваных выданняў № 2/3 ад 04.10.2013. Вул. Каржанеўскага, 20, 220024, Мінск, Рэспубліка Беларусь.

(Назва і нумар установы адукацыі)

Вучэбны год	Імя і прозвішча навучэнца	Стан вучэбнага дапаможніка пры атрыманні	Адзнака навучэнцу за карыстанне вучэбным дапаможнікам
20 /			
20 /			
20 /			
20 /			
20 /			
20 /			