Інна Володіна
Володимир Володін
Юрій Столяров

«Основи інформатики та обчислювальної техніки» – доволі молодий навчальний предмет, який вивчають та викладають у школах України трохи більше 20 років. Дійсно, за цей час склалися певні традиції та методики навчання інформатики, вченими та викладачами було розроблено велику кількість навчальних програм, що передбачають вивчення інформатики з різного віку, у класах та школах різного профілю. Але кожен викладач намагається знайти свій власний шлях у цій досить складній та цікавій справі – викладанні курсу «Основи інформатики та обчислювальної техніки», такий шлях, який привів би викладача та його учнів до спільних успіхів та максимально можливих результатів. Крім того, цей шлях має бути цікавим та зрозумілим для всіх, хто піде ним, урок за уроком, опановуючи найсучаснішу науку.

1. Різні підходи до вивчення курсу «Основи інформатики та обчислювальної техніки»

Расхождение взглядов может служить превосходной общей платформой.
Лешек Кумор, польский афорист

За роки існування цього навчального предмету у школі виділилися не лише різноманітні програми, цікаві підручники та сучасні методики викладання інформатики, а й актуальні проблеми, що зустріли викладачі під час роботи з існуючими технологіями. Як навчальний предмет, «Основи інформатики та обчислювальної техніки» складається з двох великих частин – «Комп'ютерні технології» та «Комп'ютерні науки», які в більшості навчальних програм та підручників розглядалися як дві незалежні частини курсу. Крім того, різні програми та підручники приділяли цим частинам різну увагу.

Наприклад, більшість створених програм 95% навчального часу пропонували приділити опануванню комп'ютерних технологій, а на вивчення основ алгоритмізації залишали лише по кілька годин на навчальний рік. Або навпаки, більшість навчального часу рекомендували приділити вивченню основ алгоритмізації і програмування, оскільки саме ці розділи є більш складними для учнів, і лише незначну частину годин відводили на знайомство із сучасними інформаційними технологіями.

І той, і інший підходи ми вважаємо нераціональними, оскільки, на наш погляд, ці частини є дійсно частинами одного цілого. Тому ми пропонуємо у навчальному процесі опановувати ці складові курсу у рівних частинах. Крім того, такий підхід дозволить вивчати «Комп'ютерні науки» та «Комп'ютерні технології» як єдине ціле, чергуючи теми різних розділів інформатики та демонструючи учням тісний взаємозв'язок між ними. Отже, досить важливим аспектом нашої робити ми вважаємо ***визначення послідовності тем, що вивчаються у курсі інформатики, встановлення взаємозв'язків між окремими темами та розділами інформатики, планування навчальної діяльності під час вивчення всього курсу інформатики***.

2. Спіральний принцип роботи з курсом

Наши знания есть сумма того, чему мы научились, и того, что мы забыли.
Мария Эбнер-Эшенбах, австрийская писательница

Більшість сучасних програм, за якими відбувається вивчення інформатики, пропонують опанувати дітям різного віку досить велику кількість теоретичних понять та сформувати ще більшу кількість практичних навичок, виділяючи на це обмаль навчального часу. Тому реальні результати дітей принципово відрізняються від запланованих теоретичних показників. І

головна проблема полягає не у тому, що учні не можуть засвоїти пропонований програмою навчальний матеріал.

На наш погляд, на прогнозований результат можна очікувати лише тоді, коли навчальний процес буде організовано так, що до навчального матеріалу учні будуть ***повертатися багаторазово, але маючи різний рівень знань***, а, отже, відповідно, і вимог до них. Тобто, доцільно процес вивчення теоретичного матеріалу та формування практичних навичок ***побудувати спірально***, щоб учні поверталися до вже відомого їм матеріалу багаторазово, але кожен раз з різних тем курсу, вже маючи нові знання та навички. Такий підхід дозволить виграти час для формування практичних навичок, адже, навіть закінчивши вивчати тему на уроках, учні продовжують працювати з її основними питаннями при роботі над наступними темами та розділами інформатики.

Ми вважаємо доцільним поділити курс «Основи інформатики та обчислювальної техніки» на декілька етапів:

1 етап – пропедевтичний курс «Основи інформатики», призначений для учнів 7 класів (але цей курс може викладатися з відповідною корекцією і молодшим учням). На цьому етапі відбувається початкове знайомство учнів з сучасними комп'ютерними технологіями та основами алгоритмізації, відбувається різнопланова підготовка дітей до життя в сучасному інформаційному суспільстві. Наша програма пропедевтичного курсу «Основи інформатики. 7 клас» має відповідний гриф Міністерства освіти України, а розроблений та виданий видавництвом «Гімназія» навчальний посібник з відповідними навчальними матеріалами повністю відповідає розробленій концепції та навчальній програмі.

2 етап – друге коло вивчення курсу «Основи інформатики та обчислювальної техніки», призначене для учнів 8 та 9 класів. За ці два роки поглиблюються та систематизуються знання, отримані учнями у 7 класі, та вивчаються нові теми курсу. Обов'язковим аспектом при роботі з цим курсом є повернення до питань, розглянутих за попередній рік. Але на цьому етапі навчальна ситуація має змінитися, пропонуючи учням як засвоєння базових знань курсу, так і навчаючи їх самостійній навчальній діяльності з інформатики. За ці два роки формується державний стандарт знань курсу «Основи інформатики та обчислювальної техніки». Для роботи на цьому етапі ми пропонуємо вашій увазі авторську програму, календарні плани та навчальний посібник «Основи інформатики. 8 клас», виданий видавництвом «Генеза», а також набір навчальних матеріалів, розміщених на сайті http://svitinfo.com/book.

3 етап – профільне вивчення курсу «Основи інформатики і обчислювальної техніки», призначене для учнів 10-12 класів. Саме у цих класах відбувається допрофесійна орієнтація учнів. Тому доцільно продовжити вивчення інформатики, використовуючи систему спеціальних курсів, які будуть враховувати напрямок навчання як класу, так і окремих учнів. Теми спецкурсів на протязі 10-12 класів можуть змінюватися, але має залишатися базовий курс, який є обов'язковим для вивчення учнями будь-якого профільного класу. У ці роки особлива увага приділяється індивідуальній творчій роботі старшокласників.

3. Перші кроки

Працюючи у ліцеї та викладаючи інформатику дітям різного віку у класах різного профілю навчання, ми прийшли до висновку, що особливу увагу потрібно приділити початку опанування курсу «Основи інформатики та обчислювальної техніки», адже саме на цьому етапі знайомства з інформатикою і починають проявлятися уподобання дитини, які надалі будуть формувати основи її навчальної діяльності.

Ми пропонуємо вашій увазі навчальні посібники курсу «Основи інформатики. 7 клас» та «Основи інформатики. 8 клас». Програма та посібники побудовані відповідно до розробленої нами концепції вивчення курсу інформатики.

У посібнику «Основи інформатики. 7 клас» включені такі теми:

1. Основи роботи за комп'ютером (Обчислювальна система. Історія розвитку обчислювальної техніки. Операційні системи. Основи роботи з дисками. Архіватори. Віруси та антивірусні програми).

2. Текстовий процесор.
3. Основи алгоритмізації та програмування. Лінійні алгоритми.
4. Графічний редактор.
5. Розгалуження.
6. Електронні презентації.
7. Електронні таблиці.
8. Циклічні структури.
9. Всесвітня мережа Інтернет.

Зі змісту посібника добре видно, що курс 7 класу побудовано таким чином, що вивчення сучасних комп'ютерних технологій чергується із знайомством з основами алгоритмізації та програмування. Крім того, виконуючи завдання з розділу «Комп'ютерні науки», учень паралельно опановує сучасні комп'ютерні технології. Та навпаки, під час виконання практичних завдань «Курсу користувача», учень розв'язує задачі з основ алгоритмізації та програмування. Таким чином, на протязі всього курсу в учнів формується уявлення про інформатику як єдиний та нерозривний курс, в якому немає головних та другорядних розділів або тем.

Пропоновану ідеологію вивчення інформатики продовжено у посібнику «Основи інформатики та обчислювальної техніки. 8 клас», до якого включені такі теми:

1. Продовження знайомства з *ОС Windows*
2. Основи програмування на мові Паскаль. Перше знайомство з мовою програмування Паскаль
3. Графічний редактор *CorelDRAW*
4. Продовжуємо знайомство з текстовим процесором *MS Word*
5. Модуль *CRT*. Розгалуження та вибір
6. Готуємо та проводимо тестування
7. Інтерактивні електронні презентації
8. Циклічні алгоритми

4. Особливості навчальних посібників

4.1. Головні герої

Вивчення тем курсу інформатики побудовано у наших навчальних посібниках відмінно від більшості традиційних підручників. Учні почнуть знайомство з інформатикою за допомогою нових друзів – семикласників Петрика, Стасика, Тарасика та таксі Дусі, до яких потім приєднаються андроїд Асімо та електронний песик Айбоша. З друзями на протязі першого року відбувається 27 цікавих історій, учасником яких стає семикласник, працюючий з посібником. Усі історії відбувається саме у той час, коли триває робота з навчальним матеріалом відповідного параграфа. Під час цих подорожей і відбувається засвоєння нових теоретичних знань та формування практичних навичок. Описані у посібнику історії могли статися і з самим учнем, тому більшість ситуацій, в які потрапляють нові знайомі, добре відомі дітям. Отже, засвоєння частини матеріалу відбувається підсвідомо. Крім навчання інформатиці, нові друзі ще й виховують учнів, формуючи в них певні моральні принципи та норми поведінки. Навчальний матеріал у посібника викладено таким чином, що навіть якщо інформатика як предмет ще не вивчається дитиною у школі, то за допомогою Петрика, Стасика, Тарасика та Дусі вона зможе розпочати самостійне знайомство з інформатикою.

У посібнику інформатики для 8 класу учні будуть зустрічати пригоди та розмірковування Петрика, Стасика, Тарасика та Дусі виключно при постановці завдань, виконанні практичних робіт та навчально-тренувальних завдань. На відміну від попереднього року навчання, весь теоретичний матеріал не викладений у вигляді діалогів друзів. У посібнику для 8 класу велика увага приділяється мотивації навчальної діяльності. Лише добре усвідомивши, чому ця тема або питання є актуальним, для чого це потрібно саме йому, учень докладе максимум зусиль та наполегливої праці при роботі з навчальним матеріалом.

4.2. Структура параграфів

Успех – вот что создает великих людей.
Наполеон I

В обох навчальних посібниках параграфи поділені на взаємозв'язані між собою *блоки*. Такий поділ дає можливість регламентувати час роботи з посібником як учителю, якщо посібник використовується під час уроку, так і учням та батькам, які допомагають дитині організувати навчальну діяльність вдома.

У кінці кожного теоретичного блоку наведені *запитання для самоперевірки*, відповівши на які, учень у разі потреби може повернутися до опрацювання теоретичного матеріалу чи формування практичних навичок.

Робота з параграфом має закінчуватися виконанням *навчально-тренувальних завдань*. Усі вони пов'язані із викладеним у параграфі навчальним матеріалом та за змістом відповідають пригоді, участь у якій приймали друзі. Враховуючи, що кожен учень має власні смаки та уподобання, різні захоплення та початковий рівень знань, створюючи ці завдання, ми намагалися охопити різні сфери діяльності людини. Кожне завдання посібника 7 класу представлено у викладенні одного з героїв посібника, а посібник 8 класу містить біля 70% сюжетних завдань.

Навчально-тренувальні завдання поділені на рівні, отже, кожен учень зможе обрати прийнятне для себе на даний момент завдання. Оскільки ми радимо кілька разів повертатися до роботи з навчальним матеріалом (відповідні інструкції є у посібниках), то навіть ті завдання, з якими не змогла впоратися дитина на початку вивчення теми, через декілька днів роботи виконуються швидко та правильно.

Цей методичний прийом особливо важливий на початку знайомства з курсом «Основи інформатики та обчислювальної техніки», оскільки формує ситуацію успіху, дозволяє учням повірити у власні сили, позитивно оцінити власну працю та навчальні досягнення, переконатися у стійкості отриманих знань та навичок.

Більшість навчально-тренувальних завдань демонструють єдність різних навчальних предметів. Розв'язуючи ці завдання, учні покращать свої знання з математики, фізики, біології, географії, літератури та інших навчальних предметів, ще раз переконуючись, що інформатика є наукою, що поєднує сучасні досягнення більшості фундаментальних наук.

4.3. Перевірка отриманих розв'язків

Есть люди, которые не совершают ошибок. Это те, за кого думают другие.
Хенрик Ягодзиньский, польський сатирик

При роботі з навчально-тренувальними завданнями з інформатики виникає велика кількість проблем як дітей, так і у батьків. Адже у підручниках фізики чи математики до більшості задач наведено відповіді, що дають можливість перевірити правильність отриманих розв'язків. Із завданнями з інформатики все значно складніше, до більшості завдань однозначної відповіді просто не існує. Тому створюючи посібники, ми намагалися навчити дитину, як оцінити правильність виконання завдань з різних розділів інформатики.

У посібниках детально описано технологію перевірки створених алгоритмів та програм за допомогою заповнення таблиці виконавця. Крім того, ми намагалися показати учню, як не лише користуватися наведеними у параграфі тестами для перевірки правильності розроблених програм, а й самостійно розробляти систему тестів, а потім з її допомогою перевіряти роботу програми. Тому до частини завдань у посібниках наведено систему тестів, за допомогою яких учні та їх батьки зможуть перевірити правильність складених алгоритмів та програм.

Формуванню різних методів перевірки правильності роботи самостійно створених програм велику увагу ми продовжили приділяти у посібнику 8 класу, застосовуючи для цього нові для учня методи та підходи.

4.4. Практичні роботи

При изучении наук примеры полезнее правил.
Исаак Ньютон

Велику частину навчального часу ми пропонуємо приділити формуванню практичних навичок, тому значний об’єм матеріалу у посібниках представлений у вигляді практичних та експериментально-дослідницьких робіт. Ці роботи учні можуть виконувати як у класі, так і вдома. Крім того, у більшості робіт наведено завдання різного рівня складності, що дає можливість використовувати їх при роботі з учнями, що мають різний рівень підготовки. Учні, що виказують зацікавленість інформатикою, зможуть виконувати завдання творчого рівня, організовуючи власну навчальну та пошукову діяльність.

4.5. Тематичний облік знань

Кожен модуль, який може складатися з 2-3 параграфів курсу, ми пропонуємо завершувати виконанням творчого завдання, виконуваного як окремими учнями, так і в мікрогрупах, на уроках інформатики чи у процесі вивчення інших навчальних предметів. Така робота з мікро-проектами дозволить продовжити формування основних практичних навичок учнів, доведе цілісність навчального курсу, продемонструє зв’язок інформатики з іншими науками та буде формувати творче відношення учнів до навчальної діяльності. Теми таких робіт, як і методика їх проведення, на базовому рівні описані в посібнику.

Ми пропонуємо вашій увазі вступ та матеріал двох параграфів посібника «Основи інформатики. 8 клас». Один з них є частиною розділу «Комп’ютерні технології», другий – розділу «Комп’ютерні науки». Такий вибір невипадковий. Ми хотіли б, щоб ви наочно пересвідчилися у тому, що всі описані вище принципи ми спробували реалізувати у нових посібниках, у результаті чого ми отримали «незвичні» посібники, з якими захочуть працювати як діти, так і дорослі. Сподіваємося, що ви є нашими однодумцями, отже, спільною працею ми зможемо досягти того, щоб навчати та навчатися інформатиці було цікаво та нескладно.

Дорогий друже!

Користуючись цим посібником, ти можеш продовжити вивчати курс «Основи інформатики», розпочатий у 7 класі. Ти знову зустрінеш своїх «старих» знайомих – уже восьмикласників Петрика, Стасика, Тарасика та таксу Дусю. Але цього разу вони лише пропонуватимуть тобі навчально-тренувальні завдання, окреслюватимуть розв'язувані навчальні проблеми та намічатимуть шляхи їх розв'язування. А пройти весь шлях навчання інформатики цього року тобі доведеться самому, але, звичайно, під керівництвом твого вчителя та з допомогою однокласників та дорослих.

Для того, щоб успішно засвоїти пропонований навчальний матеріал теми, радимо тобі, перед тим, як його вивчати, відповісти на контрольні запитання та виконати практичні завдання. Якщо це не викликало труднощів, можеш сміливо розпочинати роботу над новим матеріалом. Якщо ж ти забув минулорічний навчальний матеріал і не зміг безпомилково та у повному обсязі виконати пропоновані завдання, радимо тобі ще раз повернутися до відповідної теми (у цьому тобі допоможе посібник «Основи інформатики. 7 клас») і лише після цього знову виконати завдання.

Матеріали до посібника, потрібні для виконання завдань, розміщено на сайті http://svitinfo.com/book.

Зверни увагу, що посібник написано саме для тебе. Адже ти хочеш зручно та комфортно працювати на своєму комп'ютері? Бажаєш мати «неповторний» *Робочий Стіл*? Правильно встановлювати та вилучати програми? Створити власну інтерактивну презентацію з унікальними малюнками та продемонструвати її на мультимедійній дошці? Розробити власний макет щоденника, який використовуватимуть у школі майбутнього? Тоді настав час взяти посібник у руки та починати працювати!!!

Успіхів тобі, нових досягнень, бажання успішно навчатися та перемагати!

Необхідно проілюструвати підготовлений до уроку реферат? Створити власний малюнок для додавання його в електронну презентацію? Необхідно розробити емблему спортивного гуртка, яка добре б виглядала як на невеликому за розміром дипломі, так і на величезному плакаті? Бажаєте подарувати своїм друзям веселу та оригінальну власноруч виготовлену листівку?

Ці та багато інших графічних завдань ви зможете виконати, опрацювавши матеріал наступного розділу.

Розділ ІІІ. Графічний редактор CorelDRAW

Пригадай та повтори

1. Для чого призначено графічні редактори та які їх типи ви знаєте?
2. Як подають зображення векторний та растровий графічні редактори?
3. Як змінити розмір аркуша, на якому малюють у графічному редакторі *Paint*?
4. Поясніть призначення та основні способи роботи з інструментами (рис.7.1) графічного редактора *Paint*.
5. Які є стилі заливки у графічному редакторі *Paint*? У чому полягає відмінність між ними?
6. Що розуміють під редагуванням зображень?

Рис.7.1. Інструменти редактора *Paint*

Навчально-тренувальні завдання

1. Виконайте один з наведених нижче малюнків (рис.7.2).

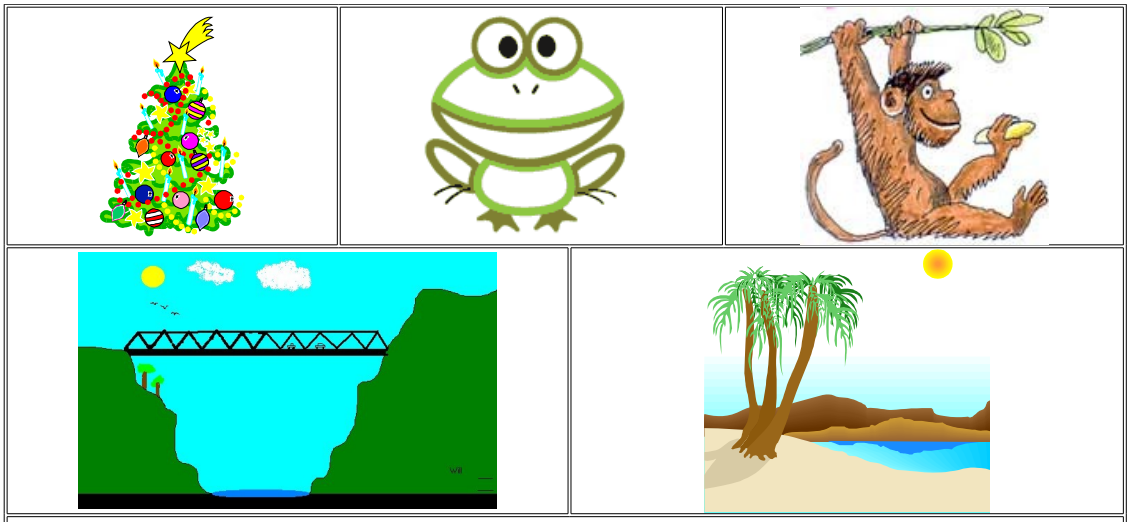

Рис.7.2. Варіанти завдань для малювання

2. З моменту відкриття у Парижі ресторану «Рататуй» родичі пацюка-кухаря Ремі погодилися допомагати йому, заготовляючи продукти для готування кулінарних шедеврів. Сьогодні Ремі попросив *x* родичів-пацюків назбирати помідори. Пацюки побігли до великого городу, де зазвичай вони заготовляли овочі, зібрали помідорів нарівно і понесли Ремі. Але дорогою пацюк Санчо пригадав томатні бої, які він бачив, мандруючи Іспанією. Тому він жбурнув помідором у пацюка Жака, що тягнув пакет з овочами поруч. Так розпочалася велика «томатна бійка», у результаті якої кожний пацюк жбурнув у кожного помідором. Як наслідок, вони принесли у ресторан «Рататуй» помідорів удвічі менше, ніж зібрали на городі.

*Розробіть програму для визначення кількості помідорів, що отримав Ремі для приготування кулінарного шедевру. Наберіть текст програми у середовищі програмування Турбо Паскаль та перевірте правильність її роботи за наведеними та самостійно розробленими тестами.

Використовуючи можливості графічного редактора *Paint*, проілюструйте «томатні бійки» родичів Ремі.

№	Аргументи	Результат
1	Кількість пацюків, що побігли збирати помідори: *x*=5	Кількість помідорів, що отримав Ремі: 20
2	Кількість пацюків, що побігли збирати помідори: *x*=10	Кількість помідорів, що отримав Ремі: 90
3	Кількість пацюків, що побігли збирати помідори: *x*=24	Кількість помідорів, що отримав Ремі: 552

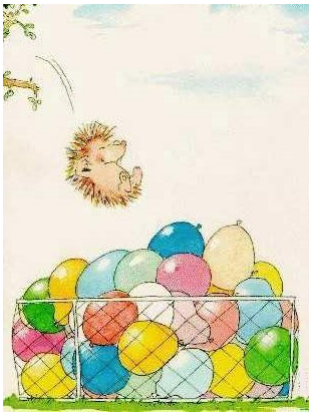

Рис.7.3. Веселий їжачок

3. На малюнку (рис.7.3) ви бачите веселого їжачка, який лопає повітряні кульки, подаровані друзями на День народження. Але звірята не знайшли кульки тих кольорів, що їм подобалися. Будь-ласка, допоможіть звірятам, перефарбувавши кульки лише у червоний, помаранчевий та фіолетовий кольори. Файл *hedgehog.jpg*, який ви будете редагувати, розміщено у робочих матеріалах посібника.

§ 7. Перше знайомство з CorelDRAW. Об'єкти та дії з ними

7.1. Переваги та недоліки растрових та векторних зображень

До цього часу ми працювали з растровим графічним редактором *Paint*. І це зрозуміло, адже більшість фотографій та малюнків, які ми створюємо, редагуємо, зберігаємо та використовуємо на своєму комп'ютері, зберігаються саме у формі растрових зображень. Багато малюнків, які ви знайдете в Інтернеті, теж растрові. Програмісти розробили багато програм, призначених для роботи саме з растровими зображеннями.

Але всі ці зображення мають один ***головний недолік*** – їх важко масштабувати. Адже, коли зменшують растрове зображення, декілька сусідніх точок перетворяться на одну, тому ми не побачимо дрібних деталей нашого малюнка. Якщо ми збільшуємо растровий малюнок, то збільшується розмір кожної точки, отже, ми побачимо малюнок, складений з великих та нечітких прямокутників. Крім цих недоліків, великі растрові зображення забирають багато пам'яті та місця на диску.

Для вирішення всіх цих проблем і створили векторний спосіб кодування зображень.

За векторного способу кодування будь-яке зображення складається з кількох складників - ***об'єктів***. ***Властивості кожного об'єкта можна змінювати незалежно від інших***. Це пояснюється тим, що характеристики об'єкта зберігаються векторним графічним редактором у пам'яті комп'ютера як математичні формули та геометричні абстракції. Наприклад, для збереження зображення прямокутника достатньо задати розмір його сторін, місце розташування та колір.

Отже, ***розміри, кривину та місце розташування*** кожного об'єкта векторний графічний редактор зберігає у формі числових коефіцієнтів. Завдяки цьому можна масштабувати зображення за допомогою множення параметрів графічних елементів на коефіцієнт масштабування, ***не змінюючи при цьому якості зображення***. Тому, працюючи у векторному графічному редакторі, не доведеться заздалегідь думати про розмір створюваного зображення, адже працюють над маленькою поштовою маркою чи величезним рекламним щитом однаково.

Ще одна суттєва перевага векторних зображень - ***малий розмір файлів***, потрібних для їх зберігання.

Але вони теж мають певні недоліки. Працюючи з векторним зображенням, ви припускаєте певну умовність. Адже всі малюнки складаються з об'єктів, заданих певними математичними формулами. Тому ви не отримаєте реалістичне зображення, оскільки для цього знадобилась би величезна кількість об'єктів, і створений файл мав би надвеликий розмір. Отже, векторні графічні редактори просто не пристосовані для роботи з фотографіями або репродукціями картин.

Лише поєднуючи можливості растрової та векторної графіки (рис.7.4), ви можете бути впевнені у тому, що отримали всі засоби для створювання, редагування, зберігання та подальшого використання будь-яких зображень.

Рис.7.4. Основні переваги та недоліки растрового та векторного редактора

Перевір себе

1. Назвіть основні недоліки растрових зображень. Чим їх можна пояснити?
2. Як зберігаються у пам'яті комп'ютера характеристики об'єктів векторного зображення?
3. Чому не змінюється якість векторного зображення внаслідок масштабування?
4. Які недоліки мають векторні зображення?

7.2. *Переваги CorelDRAW. Елементи вікна редактора*

Графічний редактор *CorelDRAW* призначено для роботи з векторною графікою. Він є складником пакету програм, розроблених компанією *Corel*. До складу цього пакету входять:

- векторний графічний редактор *CorelDRAW*, що є лідером серед аналогічних програмних продуктів;
- растровий графічний редактор *Corel Photo-Paint*;
- програма *Corel CAPTURE* для створення зображень з екрана комп'ютера.

Інтерфейс різних програмних засобів цього пакету схожий, і вони доповнюють можливості один одного. Тому засвоївши основні методи роботи з одним складником пакету, ви зможете самостійно опанувати роботу з іншим.

Ми починаємо знайомство з векторним графічним редактором *CorelDraw*, ***головними перевагами*** якого є:

- великий набір засобів створювання та редагування зображень;
- зручний інтерфейс;
- висока якість отриманих зображень.

Запустивши програму *CorelDraw*, ви побачите, що її вікно (рис.7.5) подібне до вікон застосунків, з якими ми працювали раніше. Але у ньому є кілька нових елементів.

У центрі вікна зображено аркуш паперу, який називають *робочою областю*. Ви можете малювати як усередині *робочої області*, так і за її межами. Але під час друку зображення буде виведено лише ту його частину, що міститься всередині *робочої області*.

Для зручності роботи з кольорами у правій частині вікна розташована *палітра кольорів*.

Основні засоби для роботи містяться на *панелях інструментів*, а *панель властивостей* змінює свій вигляд залежно від дій, що їх зараз виконують у редакторі. Розташування цих панелей у вікні ви можете змінити самостійно, перетягнувши їх мишею у потрібне місце.

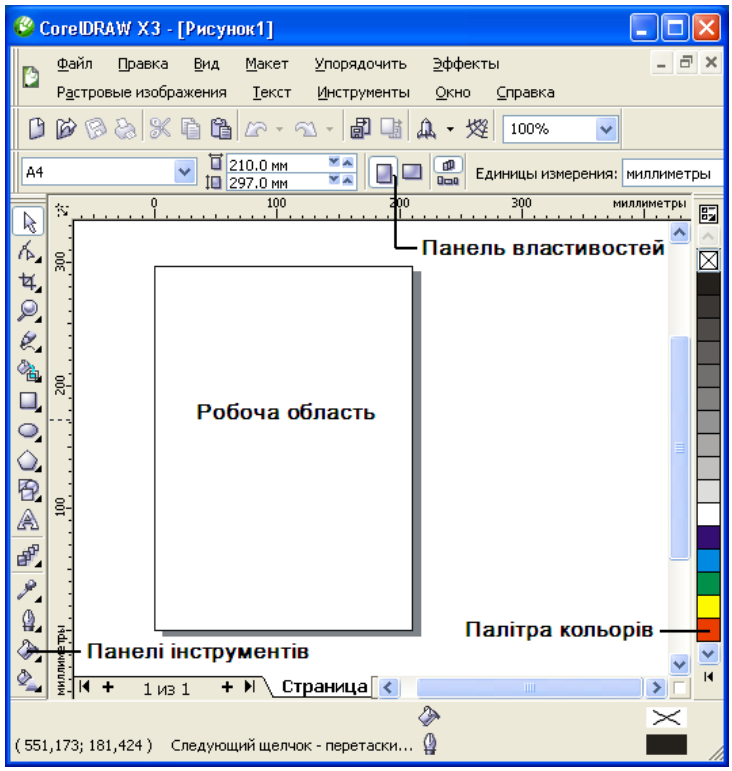

Рис.7.5. Вікно графічного редактора *CorelDRAW*

Перевір себе

5. Які програмні продукти входять до графічного пакету, розробленого компанією *Corel*? Назвіть призначення кожного складника пакету.
6. Назвіть переваги векторного графічного редактора *CorelDraw*.
7. Що називають *робочою областю* вікна редактора? Для чого її призначено?
8. Де у вікні редактора розміщені:
   - *палітра кольорів*;
   - *панелі інструментів*;
   - *панелі властивостей*?

Чи може користувач змінити їх розташування?

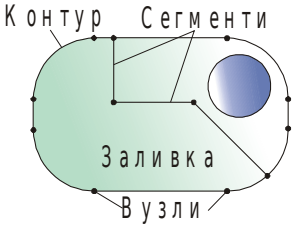

Рис.7.6. Характеристики об'єкта

7.3. *Об'єкт та його характеристики. Дії з об'єктами*

Створюючи будь-яке векторне зображення, ми створюватимемо та редагуватимемо ***об'єкти*** (елементи зображення),

з яких воно складається. Об'єктом може бути пряма чи крива, еліпс, прямокутник, багатокутник тощо. Оскільки за допомогою комбінацій об'єктів можна створити новий об'єкт, то об'єкти можуть мати досить складну форму (рис.7.6). Але незважаючи на різний зовнішній вигляд, ***усі об'єкти векторного зображення мають цілу низку однакових характеристик***:

- ***Деяку кількість точок або вузлів, з'єднаних між собою прямими або кривими (сегментами)***. Саме координати вузлів та параметри сегментів визначають зовнішній вигляд об'єкта.
- ***Заливка*** – це область всередині об'єкта, залита певним кольором (сумішшю кольорів або візерунком).
- ***Контур*** – це лінія, яку утворюють сегменти. Він має певний *колір* та *товщину*, може бути замкненим чи розімкненим.

Найпростіші ***дії, які можна виконувати з будь-яким об'єктом*** – це його ***виділяти, переміщати, масштабувати, обертати, нахиляти, копіювання та видаляти***.

Щоб ***переміщати об'єкт,*** оберіть на *панелі інструментів* інструмент *Указатель*. З його допомогою ***виділіть*** потрібний ***об'єкт***, натиснувши на ньому ліву клавішу миші. Якщо об'єкт обрано, то навколо нього з'являться чорні квадрати, які називають ***маркерами виділяння*** (рис.7.7). Утримуючи натиснутою ліву клавішу миші, ви можете легко переміщати обраний об'єкт у будь-яке місце *робочої області* або за її межі. Переміщати об'єкт можна також, використовуючи клавіші переміщення курсору. Щоб ***зняти виділення*** об'єкта, натисніть ліву клавішу миші у будь-якому вільному місці екрана.

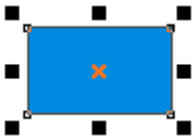

Рис.7.7. Об'єкт з маркерами виділяння

Для ***масштабування об'єктів*** (змінювання їх розмірів) треба виконати такі самі дії, як і в графічному редакторі *Paint*. Аналогічні дії ви також виконували, працюючи з автофігурами у текстовому процесорі *Word*. *Зверніть увагу на те, що будь-яку дію можна виконати лише з виділеним об'єктом.*

Для ***обертання об'єкта*** двічі натисніть на ньому ліву клавішу миші. При цьому активуються *маркери обертання* (кутові) та *нахиляння* об'єкта (горизонтальні та вертикальні) (рис.7.8). Ці маркери можна також активізувати, натиснувши лівою клавішею миші *центральний маркер* об'єкта. Для ***обертання об'єкта*** натисніть лівою клавішею миші *кутовий маркер обертання*. Не відпускаючи її, переміщайте цей маркер. Коли об'єкт повернеться на потрібний кут, відпустіть клавішу миші.

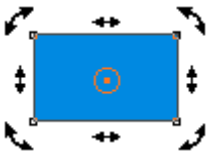

Рис.7.8. Об'єкт з маркерами обертання та нахилу

Щоб ***нахилити об'єкт,*** лівою клавішею миші натисніть *маркер нахиляння* (вертикальний або горизонтальний) (рис.7.9) та переміщайте його. Отримавши потрібний результат, відпустіть клавішу миші.

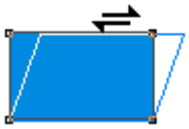

Рис.7.9. Нахил об'єкту

Копіюють та ***видаляють*** об'єкти аналогічно до цих дій в інших програмних засобах, розглянутих раніше (за допомогою команд підменю *Правка*, контекстного меню або комбінації відповідних клавіш).

Коли працюють з об'єктами, іноді виникає потреба ***скасувати останню виконану дію***. Для цього використайте підменю *Правка* головного меню редактора. У ньому оберіть команду *Отменить <назва останньої дії>* (або натисніть комбінацію клавіш *Ctrl+Z*).

Коли редагують зображення, часто виникає потреба виконати певні дії з кількома об'єктами як з єдиним цілим. Для цього об'єкти треба ***згрупувати***. Цю операцію ми вже виконували, працюючи з автофігурами у текстовому процесорі.

Спочатку потрібно виділити об'єкти, які ми хочемо згрупувати. Для цього оберіть інструмент *Указатель* і оберіть кожний об'єкт, утримуючи натиснутою клавішу *Shift*. За допомогою цього інструмента можна виділити кілька об'єктів іншим способом. Для цього

натисніть ліву клавішу миші на вільній частині *робочої області*. Не відпускаючи її, розтягуйте пунктирну *рамку виділяння*, що з'явилася на екрані, поки всі потрібні вам об'єкти не опиняться в її середині. Відпустіть ліву клавішу миші. Але цим способом важко скористатися, якщо в *робочій області* розташовано велику кількість об'єктів, а виділити вам потрібно не всі. Якщо треба виділити всі об'єкти, розташовані у *робочій області*, двічі натисніть лівою клавішею миші інструмент *Указатель*.

Коли всі потрібні об'єкти виділені, натисніть кнопку *Сгруппировать* (або комбінацію клавіш *Ctrl+G*), яка міститься на *панелі властивостей*. Аналогічні дії можна виконати, скориставшись підменю *Упорядочить* головного меню редактора. У ньому треба обрати команду *Сгруппировать*.

Щоб виконати обернену дію – ***розгруповувати об'єкти*** – оберіть об'єкт за допомогою інструмента *Указатель*, а після цього натисніть кнопку *Отменить группировку* (комбінація клавіш *Ctrl+U*) або *Отменить группировку полностью* на *панелі властивостей*. Аналогічні дії виконують за допомогою меню *Упорядочить*, обравши у ньому команду *Отменить группировку* або *Отменить группировку полностью*.

Перевір себе

9. Що таке об'єкт зображення? Які характеристики мають усі види об'єктів?
10. Що називають заливкою та контуром об'єкта?
11. Які дії можна виконати з будь-яким об'єктом?
12. Назвіть порядок дій, щоб:
    - виділити об'єкт (зняти виділення);
    - перемістити;
    - змасштабувати;
    - обернути;
    - нахилити;
    - скопіювати;
    - видалити об'єкт.
13. Для чого групують об'єкти? Опишіть послідовність дій, щоб згрупувати кілько об'єктів та потім їх розгрупувати.
14. Як у редакторі *CorelDraw* скасувати останню виконану дію?

7.4. *Змінення масштабу переглядання зображення*

Під час роботи з зображенням часто доводиться змінювати масштаб переглядання. Адже ми можемо працювати як з дрібними елементами зображення, так і переглядати його повністю. Для ***змінення масштабу переглядання зображення*** призначено інструмент *Масштаб* (гаряча клавіша *Z*). Змінити масштаб переглядання зображення можна кількома способами:

- Щоб ***збільшити масштаб*** переглядання зображення, оберіть інструмент *Масштаб* та помістіть указівник миші у центр малюнка. Указівник набуде вигляду збільшувального скла зі знаком «+». Натисніть ***ліву*** клавішу миші, і масштаб буде ***збільшено***. Якщо клавішу натиснути ще раз, масштаб збільшиться ще. Щоб ***зменшити масштаб*** переглядання зображення натисніть ***праву*** клавішу миші (або службову клавішу *F3*).
- Установіть указівник миші у лівий верхній кут області, яку треба переглянути у більшому масштабі. Натисніть ліву клавішу миші і, утримуючи її, переміщайте мишу. При цьому виділена область буде позначатися пунктирним прямокутником. Підведіть указівник до правого нижнього кута області переглядання і відпустіть клавішу миші. Зображення буде перемальовано так, що всі об'єкти, що потрапили всередину виділеної області, будуть збільшені.
- Щоб збільшити масштаб переглядання всіх об'єктів, що містяться в *робочій області* редактора, треба двічі натиснути кнопку інструмента *Масштаб*.

Панель властивостей інструмента *Масштаб* містить ще кілька корисних кнопок:

- кнопка *Только выбранные объекты (Shift+F2)* дозволяє змінити масштаб переглядання лише виділених об'єктів;
- кнопка *Все объекты (F4)* змінює режим переглядання так, щоб на екран умістилися всі об'єкти, що є у *робочій області*;
- кнопка *Страница целиком (Shift+F4)* дозволяє переглянути повністю всю сторінку (*робочу область*).

Перевір себе

15. Опишіть усі відомі вам способи змінювати масштаб переглядання зображення.
16. Яку дію треба виконати, щоб переглянути на екрані всю *робочу область* повністю?

7.5. . Змінення кольору заливки та контуру об'єкта

Щоб змінити колір заливки та контуру об'єкта (надалі називатимемо його ***абрисом***), використовують *палітру кольорів*, розташовану у правій частині вікна редактора. У кожній комірці цієї палітри розміщено колір, який можна використати для змінення. Якщо ви не бачите у палітрі потрібний колір, скористайтеся кнопками прокручування (угору чи вниз). У верхній комірці палітри намальовано *хрестик*, що відповідає варіанту, коли колір *відсутній*.

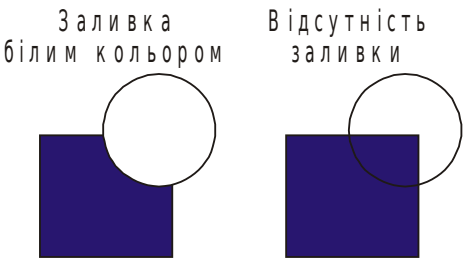

Рис.7.10. Заливка об'єкта білим кольором та відсутність кольору заливки

Використовуючи кольори для заливання, треба розрізняти *відсутність кольору* (незафарбований об'єкт буде прозорим, і ми побачимо за ним інші об'єкти) та *заливання білим кольором* (об'єкт втрачає прозорість, об'єкти за ним ми вже не побачимо) (рис.7.10).

Є багато способів ***призначити (або змінити) колір заливки та абрису***. Почнімо з найпростішого:

- Виділіть об'єкт за допомогою інструмента *Указатель*.
- Підведіть указівник миші до *палітри* та натисніть ***лівою*** клавішею миші комірку, заповнену обраним для ***заливки*** кольором. Відпустіть клавішу миші, об'єкт заповнився обраним кольором.
- Оберіть колір для ***абрису*** об'єкта. Натисніть на цій комірці ***праву*** клавішу миші. Відпустивши клавішу, ви побачите, що контур об'єкта змінив свій колір на обраний вами.

Для використання ***додаткових відтінків кольорів*** (ви їх не бачите в *основній палітрі*), підведіть указівник миші до потрібного кольору, ***натисніть і не відпускайте*** клавішу миші. Поруч з'явиться *допоміжна палітра* з відтінками обраного кольору. Оберіть з неї потрібний вам колір та відпустіть клавішу миші. Якщо ви натискали при цьому ліву клавішу, то обраним кольором буде виконано заливку об'єкта. Натискаючи та втримуючи праву клавішу миші, ви змінюєте колір абрису на бажаний.

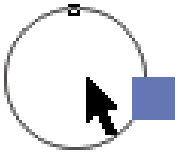

Рис.7.11. Зміна кольору заливки об'єкту

Є ще один простий спосіб ***змінити кольори заливки та абрису***. Він відрізняється від попереднього тим, що його можна застосувати для перефарбування будь-якого об'єкта, а не лише виділеного.

Для того, щоб змінити ***колір заливки,*** треба:

- Установити вказівник миші на потрібному кольорі у палітрі та натиснути ліву клавішу миші.
- Не відпускаючи її, переміщати мишу, поки її вказівник не розташується над об'єктом. При цьому вказівник миші зміниться на *Вказівник миші та прямокутник*, а у прямокутнику ви побачите обраний у палітрі колір (рис.7.11).
- Відпустіть клавішу миші. Колір заливки при цьому зміниться на обраний.

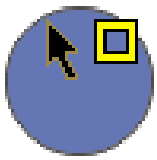

Рис.7.12. Зміна кольору абрису об'єкту

Щоб змінити ***колір абрису,*** треба:

- Установити вказівник миші на потрібному кольорі та натиснути ліву клавішу миші.
- Не відпускаючи її, переміщати мишу, поки її вказівник не розташується на контурі об'єкта. При цьому вказівник миші змінить вигляд на *Вказівник миші та рамку* (рис.7.12).
- Відпустіть у цьому місці клавішу миші. При цьому колір абрису об'єкта зміниться на обраний у палітрі.

Перевір себе

17. У чому полягає відмінність між незалитим об'єктом та об'єктом, залитим білим кольором?
18. Де на палітрі кольорів міститься комірка, що вказує на відсутність кольору заливки або абрису?
19. Як змінити колір заливки та абрису виділеного об'єкта? Будь-якого об'єкта в робочій області?
20. Як використати допоміжну палітру кольорів?

Практична робота № 6

Змінюємо колір заливки та абрису. Виділяємо, переміщаємо, масштабуємо, обертаємо та нахиляємо об'єкти

1. Запустіть програму *CorelDRAW* та відкрийте файл *зміна_кольору.cdr* (він розміщений у *папці* матеріалів до посібника). Для цього оберіть підменю *Файл* головного меню програми, а у ньому виконайте команду *Открыть* (комбінація клавіш *Ctrl+O*).
2. У робочій області розміщено два об'єкти – прямокутник та еліпс, у яких відсутня заливка (крізь еліпс ви бачите прямокутник), а контур об'єкта має чорний колір. Змініть масштаб переглядання зображення так, щоб ви добре бачили ці об'єкти.
3. Виділіть еліпс та залийте його білим кольором. Що при цьому змінилося? ***Запишіть у зошит***, чим розрізняється заливка білим кольором та відсутність заливки.
4. Залийте еліпс блакитним кольором, а колір його абрису змініть на жовтий.
5. Не виділяючи прямокутник, залийте його червоним кольором, а його контур – зеленим.
6. Змініть розташування еліпса та прямокутника так, щоб ці об'єкти не перекривалися.
7. Виділіть разом еліпс та прямокутник та однією дією залийте їх білим кольором. Колір абрису для цих об'єктів установіть помаранчевим. Ви переконалися, що змінити колір заливки та абрису можна одразу для кількох виділених об'єктів.
8. Виділіть прямокутник. На палітрі правою клавішею миші натисніть комірку з ☒ *хрестиком*. Тим самим ви встановили, що абрис об'єкта прямокутник буде відсутній. Зображення прямокутника зникло при цьому з екрана. Чи вилучили ви при цьому прямокутник? Чому ви так думаєте? *Відповідь на ці запитання запишіть у зошит.*
9. Залийте прямокутник синім кольором, після чого поверніть цей об'єкт на довільний кут проти годинникової стрілки.
10. За допомогою інструмента *Указатель* оберіть еліпс. Нахиліть його у горизонтальному напрямку.
11. Збільште розмір прямокутника та зменште розмір еліпса. Розташуйте об'єкти так, щоб еліпс частково перекривав прямокутник.
12. Обравши підменю *Файл* головного меню редактора, а у ньому команду *Сохранить как*, збережіть отримане зображення у файлі *еліпс_прямокутник.cdr* у власній папці та закрийте файли у редакторі.

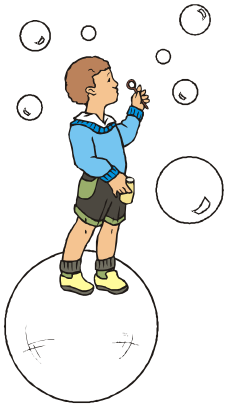

Рис.7.13

Практична робота № 7

Копіюємо та переміщаємо об'єкти. Розфарбовуємо малюнок

1. Відкрийте файл *бульбашки.cdr*. Ви бачите чорно-біле зображення хлопчика, що пускає мильні бульбашки.
2. Розфарбуйте хлопчика та його одяг різними кольорами, щоб отримати результат, показаний на рис. 7.13.

3. Збільште кількість мильних бульбашок, що видув хлопчик. Для цього виділіть та скопіюйте бульбашку будь-яким відомим вам способом.

Зауваження. *Найпростіше це зробити з виділеним об'єктом, використовуючи комбінації клавіш Ctrl+C(копіювати) та Ctrl+V(уставити). Створена копія об'єкта буде точно збігатися з початковим об'єктом (тому вам може здатися, що копіювання пройшло невдало) або розміщуватися трохи осторонь, залежно від налаштувань вашого редактора.*

4. Перемістіть копію бульбашки на нове місце та змініть її розмір. Виконайте ці дії кілька раз (кількість бульбашок на малюнку має становити 22).

5. Розфарбуйте бульбашки різними кольорами, а за бажанням змініть кольори їх абрисів.

6. Створіть ще одну копію бульбашки та зменште її розмір. Цю копію ми використовуватимемо для того, щоб змінити вигляд светра хлопчика. Залийте копії синім кольором та розташуйте цей кружечок на светрі хлопчика.

7. Скопіюйте кружечок потрібну кількість разів, переміщаючи отримані копії, щоб вийшло зображення, яке ви бачите на рис.7.14.

8. Збережіть малюнок у файлі з назвою *хлопчик.cdr* у вашій папці та закрийте змінений файл.

Рис.7.14

Навчально-тренувальні завдання

1. У редакторі *CorelDRAW* відкрийте файл *море.cdr*, у якому розміщено малюнок, наведений на рис.7.15. Ліву частину малюнка розфарбуйте за власним смаком. Використовуючи операції копіювання, переміщання та масштабування об'єктів, додайте на праву частину малюнка зірочки, що є на лівій частині. Використовуючи операцію змінювання кольору заливки, відредагуйте праву частину малюнка так, щоб усі чайки були не білі, а різноколірні. Збережіть відредагований вами малюнок у файлі *море1.cdr*.

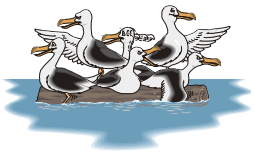

Рис.7.15

2. *Застосовуючи до об'єктів файлу *квіти.cdr* операції копіювання, переміщання, масштабування, обертання та нахиляння, створіть на малюнку букет квітів. Будуючи букет, не варто використовувати всі зібрані у файлі об'єкти. Оберіть лише кілька з них, а всі інші можете вилучити. Не забувайте, що порядок об'єктів на екрані можна змінювати, а об'єкти групувати. Збережіть створений малюнок у файлі *букет.cdr* у власній папці.

3. ****Завдання від Дуськи***

Ми з Петриком полюбляємо грати у різні ігри, особливо коли на вулиці негода, і нам доводиться сидіти вдома. Сьогодні ми грали у математичні ігри з доміно. Петрик сказав, що він знає завдання, яке я ніколи не зможу виконати.

На рис.7.16 зображено кілька кісточок доміно. Якщо на вільні місця покласти потрібні кісточки, то у кожному з рядів (двох горизонтальних та одному вертикальному) буде однакова сума чисел.

Допоможіть мені, будь ласка, заробити смачну сосиску, закінчивши малюнок з розв'язком. Для цього відкрийте файл *доміно.cdr* і, використовуючи операції копіювання та

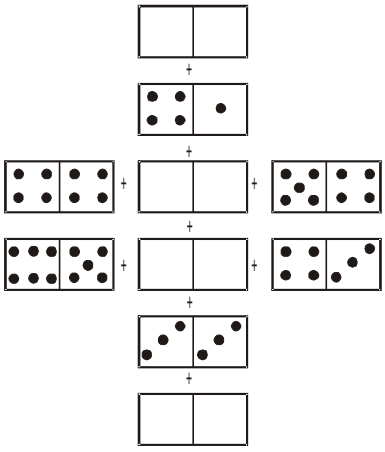

Рис.7.16 . Початкове розташування кісточок доміно

переміщання об’єктів (чорного круга), намалюйте правильне розташування кісточок доміно (розставте у порожніх кісточках чорні кружечки).

§ 18. Оператор вибору Case

Часто при розробці програм ми використовуємо кілька операторів розгалуження, розміщуючи їх у програмі як послідовно один за одним, так і вкладаючи один оператор в інший. Наші програми стають від цього доволі громіздкими та можуть працювати неефективно. Як при написанні програми спростити цей множинний вибір та підвищити ефективність створюваних програм, ви з’ясуєте, опрацювавши матеріал цього параграфа.

18.1. Розв’язуймо задачі з використанням операторів розгалуження. Оцінюймо ефективність програм

Уже кілька тижнів Асімо працює над тестувальною програмою, призначеною перевіряти рівень знань та умінь учнів з інформатики. Він запропонував друзям допомогти йому, написавши фрагмент програми мовою Паскаль. Умову поставленої задачі андроїд сформулював так:

Пройшовши тестування з інформатики, учень набрав *x* % балів з *y* можливих балів. Визначте, якому рівню знань відповідають знання учня, користуючись такою системою оцінювання:

- Якщо кількість набраних балів становить 100 – 76% від максимально можливої кількості, учень має *високий рівень знань* з інформатики.
- Якщо під час тестування учень набирає 75 – 51 % балів від загальної кількості, рівень його навчальних досягнень *достатній*.
- Якщо учень набрав 50 – 26 % балів від максимально можливої кількості, він отримає оцінку *середнього рівня*.
- Якщо ж у процесі тестування набрано 25 – 0 % балів, цей учень має знання з інформатики *низького рівня*.

Розгляньмо варіант фрагмента програми, запропонованого Петриком та Дуською:

```
{---------------------------------передаємо значення x та y з програми Асімо-----------------------------------}
If x>=y*0.76 {кількість балів x не менша за 76% від максимальної  кількості балів y}
        Then writeln (‘Рівень знань високий’)
        Else If x>=y*0.51
                Then writeln (‘Рівень знань достатній’)
                Else If x>=y*0.26
                        Then writeln (‘Рівень знань середній’)
                        Else writeln (‘Рівень знань низький’);
{---------------------------------кінець фрагмента програми-----------------------------------------------------}
```

Цей фрагмент програми містить три *вкладені* умовні оператори. Розгляньмо, як працює цей фрагмент.

Після отримання значень *x* та *y* з програми Асімо, запропонований Петриком та Дуською фрагмент програми обраховує значення умови (*x*>=*y**0.76, тобто, кількість набраних балів *x* не менша за 76% від максимально можливої кількості балів *y*), записаної у зовнішньому

операторі розгалуження. Якщо ця умова набуває істинного значення (true), то ми маємо виконати команди, записані після оператора Then, тобто повідомити, що рівень знань учня високий та закінчити виконання умовного оператора. Якщо ж ця умова набуває хибного значення (false), програма переходить до першої вітки else.

У цій вітці стоїть другий оператор розгалуження, отже, ми маємо перевірити умову, записану після службового слова if. Цю умову записано так: *x*>=*y**0.51 (кількість набраних балів *x* не менша за 51% від максимально можливої кількості балів *y*). Але насправді, якщо ми вже потрапили в цей вкладений оператор, ми переконалися в тому, що кількість набраних балів менша за 76% від *y* (це ми зробили, перевіряючи умову зовнішнього розгалуження). Тому записану в другому вкладеному операторі розгалуження умову потрібно розуміти так: *кількість набраних балів x повинна бути меншою за 76% від y та не меншою за 51% від y*. Якщо ця умова набуде істинного значення, на екран буде виведено повідомлення про те, що учень має достатній рівень знань з інформатики (програма виконала команди, записані у другій вітці then і вийшла з вкладеного оператора розгалуження). У випадку, якщо умова набуває значення false (хибність), ми знову переходимо до другої вітки else, у якій міститься третій вкладений умовний оператор.

Виконання третього вкладеного оператора розгалуження знову починається з обрахування значення умови: *x>=y*0.26*. Якщо ми потрапили у цю вітку вкладеного оператора, це свідчить, що дві попередньо записані умови набули хибного значення (тобто, кількість набраних під час тестування балів *x* була меншою за 76% від *y* (перша умова) та меншою за 51% від *y* (друга умова)). Залишилось обрахувати, чи значення набраних балів *x* не менше за 26% від максимальної кількості балів *y*. Якщо записана умова набуває значення *істина*, ми виконуємо команди, записані в останній вітці then третього вкладеного оператора розгалуження (виводимо на екран повідомлення про те, що учень має середній рівень навчальних досягнень) та закінчуємо виконувати фрагмент програми. У випадку хибного значення записаної умови, можна стверджувати, що значення хибність набули всі попередньо перевірені умови. Тому ми переходимо до останньої вітки else третього вкладеного оператора розгалуження, виводимо повідомлення про те, що рівень знань учня низький та закінчуємо виконувати цей фрагмент програми.

Рис.18.1. Родина мотрійок

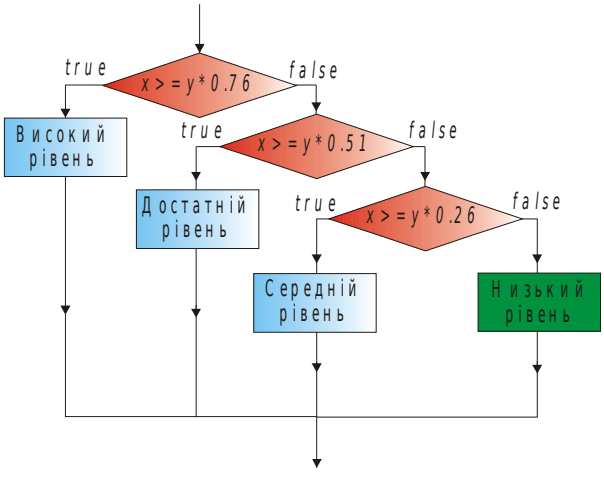

Рис.18.2. Блок-схема фрагмента програми, записаного Петриком та Дусею

Отже, у наведеному фрагменті програми ***внутрішній оператор розгалуження виконується лише тоді, коли ми виконали зовнішній для нього оператор і отримали при цьому хибне значення його умови***. Тобто, виконання вкладених операторів розгалуження можна порівняти з відкриванням мотрійки (рис.18.1): ніколи не зможеш відкрити найменшу іграшку, поки послідовно не відкриєш усі фігурки, більші за неї.

Блок-схему фрагмента програми, розробленого Петриком та Дусею, наведено на рис.18.2.

Розгляньмо варіант програми, запропонований Стасиком та Тарасиком:

```
{---------------------------- передаємо значення x та y з програми Асімо----------------------}
If x>=y*0.76 {кількість балів x не менша за 76% від максимальної  кількості балів y}
        Then writeln ('Рівень знань високий');
```

```
If (x<y*0.76) and (x>=y*0.51) {кількість балів x менша за 76% та не менша за 51% від максимальної кількості
                                балів y}
        Then writeln ('Рівень знань достатній');

If (x<y*0.51) and (x>=y*0.26) {кількість балів x менша за 51% та не менша за 26% від максимальної кількості
                                балів y}
        Then writeln ('Рівень знань середній');

If (x<y*0.26) {кількість балів x менша за 26% від максимальної кількості балів y}
        Then writeln ('Рівень знань низький');
{--------------------------------кінець фрагмента програми-------------------------------------}
```

Цей фрагмент програми містить чотири оператори розгалуження, які виконуються *послідовно*.

Спочатку ми обрахуємо значення простої умови, записаної у першому операторі. Якщо вона набула істинного значення, виконуємо вітку then цього оператора. Не залежно від того, виконали ми цей оператор чи ні, переходимо до виконання другого оператора розгалуження.

Тут ми маємо перевірити, якого значення набула складена умова. Якщо її значення true, виведемо на екран повідомлення про те, що учень має достатній рівень знань з інформатики та перейдемо до виконання наступного умовного оператора.

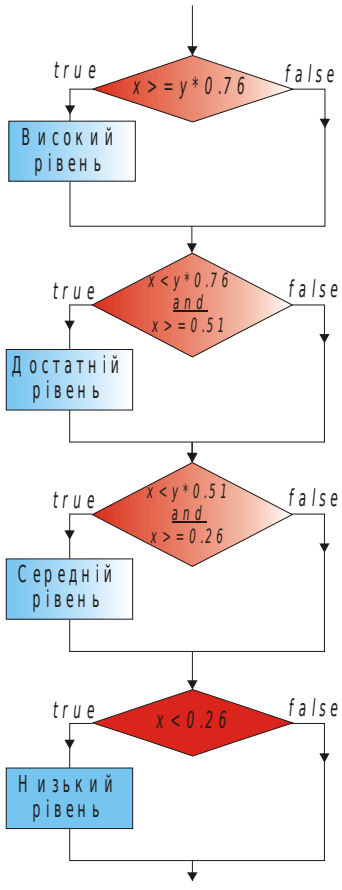

Рис.18.3. Блок-схема фрагмента програми Стасика та Тарасика

Далі аналогічно послідовно виконаємо ще два оператори розгалуження та закінчимо виконання цього фрагмента програми.

Отже, у фрагменті програми Стасика та Тарасика, ми обов'язково виконуватимемо всі оператори розгалуження (рис.18.3). А які дії у них виконувати, буде залежати від значення записаної у кожному операторі умови.

Із порівняння наведених фрагментів програм, стає зрозумілим, що ефективніший варіант реалізації запропонували Петрик та Дуська, оскільки можна отримати результат, перевіривши меншу кількість умов. Але є такі значення аргументів програми, за яких кількість дій, виконаних в обох фрагментах, виявиться приблизно однаковою. Тому, ***пишучи програми, іноді доводиться поступатися ефективністю для того, щоб отримати зрозуміліший для вас запис програми***.

Але обидва пропоновані фрагменти програми доволі громіздкі. Для спрощення записування та виконання програм, що містять як укладені, так і послідовні умовні оператори, створили оператор вибору.

Перевір себе

1. Опишіть послідовність виконання вкладеного умовного оператора. За якої умови виконується внутрішній оператор розгалуження?
2. Опишіть послідовність виконання послідовних операторів розгалуження. Чи обов'язково мають виконуватися всі умовні оператори, послідовно записані у програмі?
3. Якому варіанту записування умовних операторів (вкладеному чи послідовному) віддають перевагу? Чому?

18.2. Оператор вибору Case

Оператор вибору є як в *Навчальній Алгоритмічній Мові*, так і в мові програмування Паскаль. Він має такий формат запису:

```
Case <керівна змінна або вираз> of
        <набір значень 1 або діапазон 1>: <серія команд 1>;
        <набір значень 2 або діапазон 2>: <серія команд 2>;
        . . .
        <набір значень N або діапазон N>: <серія команд N>
        Else <альтернативна до всіх наборів або діапазонів значень серія команд>
end;
```

Виконання цього оператора починається з обчислення значення виразу або керівної змінної, записаних після службового слова case. Сама назва змінної вказує, що саме залежно від її значення будуть виконуватися певні дії. Керівна змінна може мати один з перераховних типів (це такий тип, значення якого можна перерахувати) - лише цілочисловий, символьний або логічний тип. Змінні дійсного та рядкового типів не можна використовувати як керівні змінні оператора case.

Залежно від обрахованого значення виразу або керівної змінної буде реалізовано лише одну відповідну цьому значенню серію команд.

Якщо жодний набір або діапазон значень не відповідає значенню виразу або керівної змінної, виконується альтернативна до всіх наборів або діапазонів значень серія команд (її записують після службового слова else). Ця серія команд є необов'язковою (її відсутність можна порівняти із скороченою формою оператора розгалуження).

Записуючи набір можливих значень керівної змінної, значення, які мають той самий тип, що й керівна змінна, відділяють одне від одного комою.

Набір можливих значень керівної змінної можна подати не лише перелічивши ці значення, а й за допомогою ***діапазону***, який записують так:

<початкове значення діапазону>..<кінцеве значення діапазону>

Наприклад, діапазон 2..6 містить такі цілі числа: 2, 3, 4, 5, 6, а діапазон 'a'..'c' складається з символів 'a', 'b', 'c'.

Запишемо фрагмент програми визначення рівня знань учня, використовуючи оператор вибору. Оскільки значенням керівної змінної *не може бути дійсна величина*, уведемо допоміжну змінну *c*, яка зберігатиме значення відсотка набраних учнем знань, заокруглене до цілого числа. Щоб обрахувати її значення треба кількість набраних балів (значення змінної *x*) помножити на 100%, а знайдений результат поділити на максимально можливий бал (значення змінної *y*): *c*:= *trunk (x*100/y);*

```
{----- передаємо значення x та y з програми Асімо----------}
c:= trunk (x*100/y);
Case x of
        76 .. 100 : writeln ('Рівень знань високий');
        51 .. 75 : writeln ('Рівень знань достатній');
        26 .. 50 : writeln ('Рівень знань середній')
       else writeln ('Рівень знань низький')
end;
{----------кінець фрагмента програми----------}
```

Записаний фрагмент програми працюватиме так:

- Після отримання значень *x* та *y* з програми Асімо, визначатиметься, скільки відсотків від максимальної кількості можливих балів набрав учень. Потім це значення буде заокруглено до цілого числа та збережено у змінній *c*.
- Залежно від того, яке значення отримано та збережено у змінній *c*, буде виконуватися лише одна з чотирьох можливих команд виведення повідомлення на екран.
    - Якщо значення змінної *c* належить діапазону від 76 до 100, то буде виконана перша команда оператора case (тобто, на екран буде виведено повідомлення, що учень має високий рівень знань) і після цього виконання оператора вибору буде закінчено.

- У разі, якщо значення *c* належить діапазону від 51 до 76, то програма виведе на екран повідомлення про те, що рівень знань учня достатній та перейде до виконання наступного за case оператора (readln).
- Якщо ж значення змінної *c* належить третьому можливому діапазону (від 26 до 50), то програма виконає третю команду оператора вибору, тобто виведе повідомлення про середній рівень знань учня та закінчить на цьому виконання оператора вибору.
- У випадку, коли значення змінної *c* не належить жодному з трьох розглянутих діапазонів, то програма перейде до виконання команди, записаної після оператора else (виведе на екран повідомлення про низький рівень знань учня).

Отже, наша програма залежно від того, якого значення набуває змінна *c*, ***виведе на екран лише одне з чотирьох можливих повідомлень і перейде до виконання наступної за оператором вибору дії***.

Перевір себе

4. Який формат записування оператора case у мові програмування Паскаль? Яка послідовність виконання оператора вибору?
5. Що називають керівною змінною? Яким може бути тип цієї змінної?
6. Як задають можливі значення керівної змінної?

18.3 Розв'язуймо задачі з використанням оператора вибору

Розгляньмо, як використовувати оператор вибору, розв'язуючи задачі на такому прикладі:

Розробіть програму, яка визначатиме вартість міжміської телефонної розмови за її тривалістю у хвилинах та кодом міста абонента (вартість хвилини розмови залежить від відстані між містами, її визначають за введеним кодом). Інтерфейс програми повинен мати вигляд, наведений на рис.18.4.

Рис.18.4. Вигляд інтерфейсу програми про телефонний зв'язок

Для збереження значень аргументів та результату роботи програми введімо такі змінні:

kod – змінна типу *word*, призначена для збереження коду міста;

trival – цілочислова змінна типу *byte*, у якій зберігатимемо тривалість розмови у хвилинах (ураховуємо, що вона не перевищить 255 хвилин);

vart_rozm – змінна типу real, значенням якої буде загальна вартість розмови (для її визначення треба вартість однієї хвилини помножити на тривалість розмови).

Вартість однієї хвилини розмови визначають за кодом міста. Тому для обчислення загальної вартості розмови можна використати оператор вибору, керівною змінною якого буде код міста.

Блок <визначення вартості розмови> деталізуємо так:

```
Case kod of
    4494: vart_rozm:=0.4*trival;
    432: vart_rozm:=0.8*trival;
    57, 322, 352: vart_rozm:=trival;
    56, 372: vart_rozm:=1.2*trival
    Else writeln ('Інформація про вартість розмови з цим містом невідома')
end;
```

Залежно від уведеного користувачем коду міста, вартість розмови буде визначено за однією з формул, записаних в операторі вибору. Якщо користувач уведе код, значення якого не входить у набір значень жодної вітки оператора case, на екран виведеться повідомлення про відсутність інформації про вартість розмови з цим містом.

Залишилось вивести на екран вартість розмови. Але цю дію потрібно виконувати лише тоді, коли ми змогли обчислити вартість розмови. Тому блок виведення результату роботи програми запишемо, скориставшись неповною формою оператора розгалуження:

```
If vart_rozm<>0
  Then ('Вартість розмови становить ', vart_rozmov:6:2, ' грн.');
```

Пропонуємо вам закінчити розробляти програму про вартість телефонної розмови, виконавши практичну роботу № 18.

Практична робота № 18

Розв'язуємо та перевіряємо задачі з використанням операторів вибору та розгалуження

1. Створіть у середовищі програмування Турбо Паскаль новий файл, щоб в ньому набрати та редагувати текст програми про вартість міжміської телефонної розмови.
2. Наберіть у цьому файлі загальну структуру програми, зазначивши її назву (**Telefon**) та оформивши описову частину програмного блока (під'єднайте модуль *CRT* та опишіть змінні, що використовуватимемо у програмі).
3. *Почніть розробляти інтерфейс програми відповідно до рис.18.4. У першому вікні (колір тла цього вікна синій) виведіть на екран інформацію про назви міст, вартість хвилини розмови з якими ви знаєте, коди цих міст та зазначте вартість хвилини розмови з кожним зазначеним містом.
4. *Друге вікно, колір тла якого зелений, призначено для введення значень аргументів задачі. Деталізуйте блок, що відповідає за введення значень аргументів задачі відповідно до зразка на рис.18.4.
5. Наберіть текст блока <визначення вартості розмови>, деталізований у параграфі 18.3.
6. *Створіть вікно, призначене для виведення отриманого результату (колір тла цього вікна червоний), та наберіть текст блока <виведення результату>.
7. Збережіть текст програми у файлі *telefon.pas* у власній папці. Відкомпілюйте програму та за потреби виправте допущені помилки.
8. Виконайте програму, перевіривши правильність її роботи за наведеними нижче тестами. За потреби внесіть зміни у текст програми та ще раз збережіть програму у файлі *telefon.pas*.

№	Аргументи	Результат
1	Код міста: 4494 Тривалість розмови у хвилинах: 15	Вартість розмови становить 6.00 грн.
2	Код міста: 432 Тривалість розмови у хвилинах: 21	Вартість розмови становить 16.80 грн.
3	Код міста: 352 Тривалість розмови у хвилинах: 9	Вартість розмови становить 9.00 грн.
4	Код міста: 56 Тривалість розмови у хвилинах: 7	Вартість розмови становить 8.40 грн.
5	Код міста: 895 Тривалість розмови у хвилинах: 4	Інформація про вартість розмови з цим містом невідома

9. *Перевірте правильність роботи програми за самостійно розробленими тестами.
10. *Розробіть програму, яка за номером місяця, уведеного користувачем, виводить його назву та визначає пору року, якій відповідає цей місяць.

За потреби використайте ключ для виконання завдання.

Ключ до виконання завдання

В описовій частині програмного блока введіть змінну *nomer* типу *byte*, призначену для збереження введеного користувачем значення номера місяця.

Подальші дії програми залежать від цього значення, тому для розв'язання задачі використайте оператор вибору, керівною змінною якого і буде цілочислова змінна *nomer.* Можливі значення цієї змінної у кожній вітці оператора case можна задавати діапазоном значень, а можна просто перераховувати, записуючи можливі значення через кому. Можна одночасно скористатися обома цими формами записування. *Наприклад, 12, 1..2.*

Записуючи оператор вибору треба врахувати, що користувач може ввести ціле число, яке не буде номером місяця. Тому в операторі case потрібно передбачити вітку, яка буде альтернативною до всіх попередньо розглянутих наборів значень керівної змінної *nomer.*

11. **Використовуючи можливості модуля *CRT*, розробіть зручний та зрозумілий інтерфейс програми. Збережіть змінений текст програми у файлі *rik.pas*, відкомпілюйте програму та виправте допущені помилки.
12. **Перевірте правильність роботи програми, користуючись самостійно розробленими тестами.

Навчально-тренувальні завдання

1. Запишіть набір можливих значень величини, використовуючи для цього діапазон:
   1) натуральні числа від 1 до 100;
   2) цілі числа від -10 до 10;
   3) 4, 5, 6, 7, 8;
   4) символи англійського алфавіту від *d* до *k*.

2. Обрахуйте значення змінних *x* та *y* після виконання такого фрагмента програми, якщо значення змінної *a* дорівнює:

1) 5;	2) 34;	3) 178;	4) 400.

```
x:=true;
y:=1;
case a mod 10 of
    1: ;
    2..3, 5, 7: y:=a;
    4, 6, 8: begin
                    x:=not x;
                    y:=y+a*2
             end;
    9: begin
             x:=false;
             a:=a*1.23;
             y:=(y-a)/4
       end;
end;
```

3. Напишіть програму, яка за введеним вами номером дня тижня виводить його назву. Якщо цей день є робочим, то програма виводить на екран розклад ваших уроків на цей день.

4. *Вартість квитка на «Льодову арену» залежить від часу початку сеансу (див. таблицю).

Час початку - час закінчення сеансу	Вартість квитка, грн.
8:00 – 9:30 10:00 – 11:00 12:00 – 13:00	25 грн.
14:00 – 15:00 16:00 – 17:00	30 грн.
18:00 – 19:00 19:00 – 20:00 20:00 – 21:00	35 грн.
22:00 – 23:30	25 грн.

Вартість дитячого квитка становить половину вартості квитка для дорослого. Якщо у відвідувача ковзанки є дисконтна картка, купуючи квиток, він отримує знижку, яка становить *x*%.

Напишіть програму для визначення суми коштів, які повинен сплатити відвідувач ковзанки у касу. Спілкуючись з касиром, покупець повідомляє:

- час початку сеансу *(наприклад, 19)*;
- окремо кількість дитячих та дорослих квитків;
- наявність дисконтної картки.

Передбачте можливість неправильного введення часу початку сеансу.

Використовуючи можливості модуля *CRT*, розробіть зручний та зрозумілий інтерфейс користувача програми.

У графічному редакторі *CorelDRAW* створіть малюнок на тему «На ковзанці».

4. Завдання від Стасика

На уроці фізики ми довідалися, що в 1960 році на XI Генеральній конференції з мір та ваг вчені різних країн домовилися використовувати єдину Міжнародну систему одиниць SI. Але у певних життєвих ситуаціях не завжди зручно задавати значення величин лише в основних одиницях цієї системи. Наприклад, не зручно вказувати товщину сторінки зошита в метрах, а тривалість сеансу в кінотеатрі у секундах. Тому я вирішив написати програму, яка переводитиме значення введеної величини у ті одиниці, що обере користувач. На рис.18.5. наведено інтерфейс розробленої мною програми переведення часу.

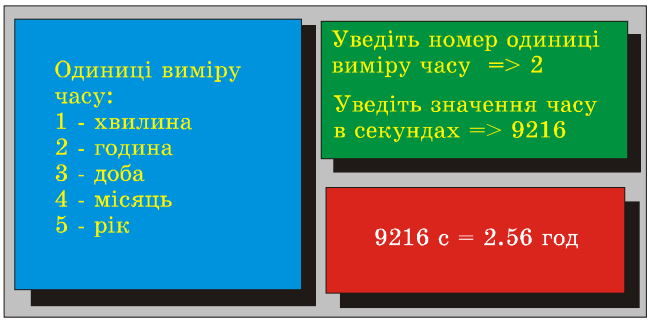

Рис.18.5. Інтерфейс програми Стасика

1. *Розробіть описану вище програму, оформивши її інтерфейс відповідно до рис.18.5.
2. **Розробіть аналогічну програму, але призначену переводити відстань, значення якої вводять у метрах, у *мм, дм, км, фути, милі, дюйми* залежно від потреби користувача. До початку розробляння цієї програми у текстовому процесорі створіть документ, у який додайте таблицю, що міститиме значення всіх пропонованих одиниць виміру відстані (вигляд цієї таблиці визначте самостійно). Якщо ви не знаєте, чому дорівнює значення якоїсь запропонованої одиниці відстані, знайдіть потрібну інформацію в Інтернеті.