Сумський державний педагогічний університет імені А. С. Макаренка
Фізико-математичний факультет
Кафедра інформатики

УДК 378.016:51:004

Притика Оксана Валентинівна

ФОРМУВАННЯ НАВИЧОК ОРГАНІЗАЦІЇ ЦИКЛІЧНИХ ОБЧИСЛЕНЬ НА УРОКАХ ІНФОРМАТИКИ СТАРШОЇ ШКОЛИ

Галузь знань: 01 Освіта
Спеціальність 014 Середня освіта (Інформатика)
Кваліфікаційна робота на здобуття освітнього рівня «Магістр»

Науковий керівник:
_____ О.В.Семеніхіна,
доктор педагогічних наук, професор,
професор кафедри інформатики

Виконавець:
_____ О.В. Притика

Суми – 2021

ЗМІСТ

ВСТУП

Одним з основних завдань закладу освіти є інтелектуальний розвиток учнів, важливою складовою якого є алгоритмічне мислення. Найбільший потенціал для формування алгоритмічного мислення школярів, крім математики, має інформатика. Багато в чому роль інформатики у розвитку алгоритмічного мислення обумовлена навичками циклічних обчислень, які формуються при вивченні програмування.

За аналізом науково-методичних розвідок, присвячених навчанню програмувати та дотичних до цієї теми, виявлено: концептуальні засади вивчення програмування (М. Жалдак, В. Конюшко, Б. Маккарти, Г. Шилдт та ін.); проблеми освітніх розривів у ланці «школа-університет» під час вивчення мов програмування і способи їх подолання (В. Клочко, Н. Морзе, О. Овчарук та ін.) Так, М. Жалдаком обґрунтовано проблему невідповідності необхідних знань з програмування та часу, який відводиться на їх засвоєння у ЗЗСО. І. Мінтій у навчанні програмувати пропонує використовувати теорію винахідницьких рішень. О.Співаковським пропонується діяльнісний підхід та активні методи навчання. На проектному методичному підході, як дієвому шляху розв’язання проблеми формування алгоритмічного мислення молоді загострюють увагу Т. Вдовичин, І. Дединський, Л. Лазурчак та ін. Натомість, С. Іщеряковим наголошується на дієвості та ефективності задачного підходу вивчення програмування, який ґрунтується на принципі «одна задача – один розв’язок». Отже, проблема змісту, якості і рівня підготовки учнів при формуванні в них навичок програмувати перебуває в центрі уваги педагогів-дослідників. Втім, не зважаючи на значний напрацьований досвід у цій галузі, через постійний розвиток ІТ, мов програмування та середовищ програмування і відповідно часте оновлення навчальних програм з інформатики маємо констатувати відсутність ефективних напрацьованих методик формування навичок організації циклічних обчислень учнями старшої школи та достатнього дидактичного матеріалу.

Об’єкт дослідження: навчання інформатики учнів старшої школи.

Предмет дослідження: формування навичок організації циклічних обчислень на уроках інформатики старшої школи.

Мета дослідження: виявити особливості формування навичок організації циклічних обчислень на уроках інформатики старшої школи

Поставлена мета дослідження обумовила вирішення низки завдань:

1) охарактеризувати типові мови програмування, які вивчаються в ЗЗСО, описати циклічні обчислення різними мовами програмування;

2) виявити стан розробленості проблеми формування навичок організації циклічних обчислень на уроках інформатики старшої школи;

3) проаналізувати навчальні програми з інформатики та чинні підручники на предмет формування навичок організації циклічних обчислень на уроках інформатики старшої школи;

4) розробити авторські матеріали з формування навичок організації циклічних обчислень на уроках інформатики старшої школи.

Для досягнення мети використано низку **методів** дослідження:

теоретичні – систематизація і узагальнення науково-методичних джерел для обґрунтування актуальності проблеми дослідження; контент-аналіз з метою характеристики мов програмування та їх затребуваності на ринку праці; аналіз навчальних програм та чинних підручників з інформатики для визначення вимог до рівня підготовленості учнів з програмування загалом і умінь організації циклічних обчислень, зокрема;

емпіричні – опитування учнів і бесіди з учителями для виявлення практичного стану розробленості проблеми формування навичок організації циклічних обчислень на уроках інформатики старшої школи.

Практична значущість дослідження полягає в розробленні авторських матеріалів з формування навичок організації циклічних обчислень на уроках інформатики старшої школи.

Апробація матеріалів дослідження здійснювалася на наукових заходах різних рівнів, серед яких: XIV Всеукраїнська науково-практична конференція «Інформаційні технології у професійній діяльності» (1 листопада 2021 року,

м. Рівне) [78] та на онлайн-семінарі Лабораторії використання ІТ в освіті (22 квітня 2021 року).

Структура та обсяг роботи. Кваліфікаційна робота складається зі вступу, двох розділів, загальних висновків та списку використаних джерел.

У першому розділі «Проблеми навчання циклічних обчислень на уроках інформатики старшої школи» за контент-аналізом матеріалів мережі Інтернет виявлено найбільш популярні мови програмування, які вивчаються в ЗЗСО, а також виявлено стан розробленості проблеми формування навичок організації циклічних обчислень на уроках інформатики старшої школи.

У другому розділі «Особливості формування навичок організації циклічних обчислень на уроках інформатики старшої школи» на основі аналізу навчальних програм з інформатики та чинних підручників з інформатики старшої школи на предмет формування навичок організації циклічних обчислень на уроках інформатики старшої школи виявлено методичні проблеми, які пропонується вирішувати з використанням авторських дидактичних матеріалів.

Загальний обсяг роботи 63 сторінок основного тексту. Список використаних джерел включає 37 одиниць. Робота містить 26 рисунків та 4 таблиці.

Робота буде цікавою працюючим і майбутнім учителям інформатики, які досліджують проблеми формування навичок організації циклічних обчислень на уроках інформатики.

РОЗДІЛ 1.
ПРОБЛЕМИ НАВЧАННЯ ЦИКЛІЧНИХ ОБЧИСЛЕНЬ НА УРОКАХ ІНФОРМАТИКИ СТАРШОЇ ШКОЛИ

1.1. Мови програмування, які вивчаються у ЗЗСО

Цифровізація суспільства призвела до потреби активного опанування цифрових технологій вже у закладах загальної середньої освіти. І якщо певний час школи орієнтувалися на підготовку в межах предмету «Інформатика» на користувача програмного засобу чи інформаційної системи, то сьогодні прийшло усвідомлення важливості формування на уроках інформатики теоретико-математичних основ функціонування інформаційних систем, що серед іншого передбачає формування елементарних навичок алгоритмізації і програмування, яке реалізується, як правило, на основі навчання циклічних обчислень (якщо обчислювальний процес містить багаторазові обчислення за однаковими (однотипними) математичними залежностями, але для різних значень змінних, то такі обчислення називають циклічними), і при цьому не залежить від конкретної мови програмування.

Опишемо коротко мови програмування, котрі наразі обираються для вивчення у закладах загальної середньої освіти для формування навичок циклічних обчислень. Серед таких: Scratch, Visual Basic, Pascal, Object Pascal, C++, Python, Java та Ruby.

Scratch

Scratch – мова програмування з однойменним візуальним об’єктно-орієнтованим середовищем для роботи. Вони (мова і середовище) мають свій логотип (рис. 1.1). У Scratch використовуються об’єкти-спрайти, якими за допомогою алгоритмів керують самі учні.

Рис. 1.1. Талісман середовища програмування Scratch

Для роботи у Scratch виділяють блоки декількох видів: рух, зовнішність, звук, перо (використання черепашачої графіки), контроль, сенсори, операції та змінні. Кожен блок має свої налаштування та параметри програмування на певні дії. Одна з основних відмінностей Scratch від іншим мов програмування – відсутність потреби у знаннях: в Scratch все програмується за допомогою візуальних блоків.

Найпоширеніше застосування Scratch – це навчання у формі створення мультфільмів або ігор. Крім цього, Scratch можна використовувати для освітніх цілей і створювати в програмі ілюстративні матеріали для уроків не тільки з програмування, а й з історії, біології, фізики та інших предметів. У Scratch передбачено функція звукового редактора, яка розширює можливості роботи з різними видами даних.

Проєкт зі створення Scratch ініційований у 2003 році за фінансової підтримки підприємств Science Foundation, Intel Foundation, Microsoft, MacArthur Foundation, LEGO Foundation, Code-to-Learn Foundation, Google, Dell, Fastly, Inversoft і MIT Media Lab research consortia [17].

Scratch створений у 2007 році в лабораторії Lifelong Kindergarten Массачусетського технологічного інституту під керівництвом професора Мітчела Резника (Mitchel Resnick). На сайті scratch.mit.edu зареєстровано понад 47 млн користувачів з усього світу. З них 18,8 млн з США, 2,7 млн з Великобританії, 3 млн з Китаю і 119 204 користувачів із України [7].

Основний вік учасників спільноти з вивчення Scratch – 10-19 років (рис. 1.2).

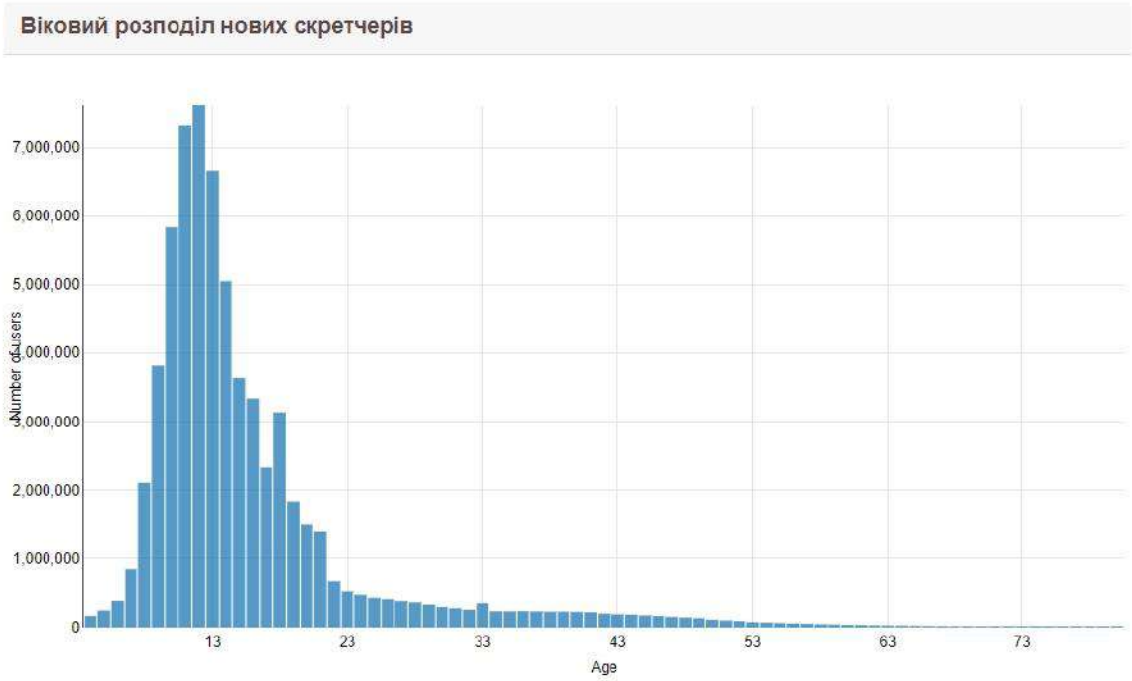


Рис. 1.2. Розподіл користувачів за віком на сайті scratch.mit.edu

Scratch став настільки актуальним для людей, що за жовтень 2021 року на сайті було опубліковано більше 2 млн. нових проєктів і майже 813 тис. оновлених (рис. 1.3).

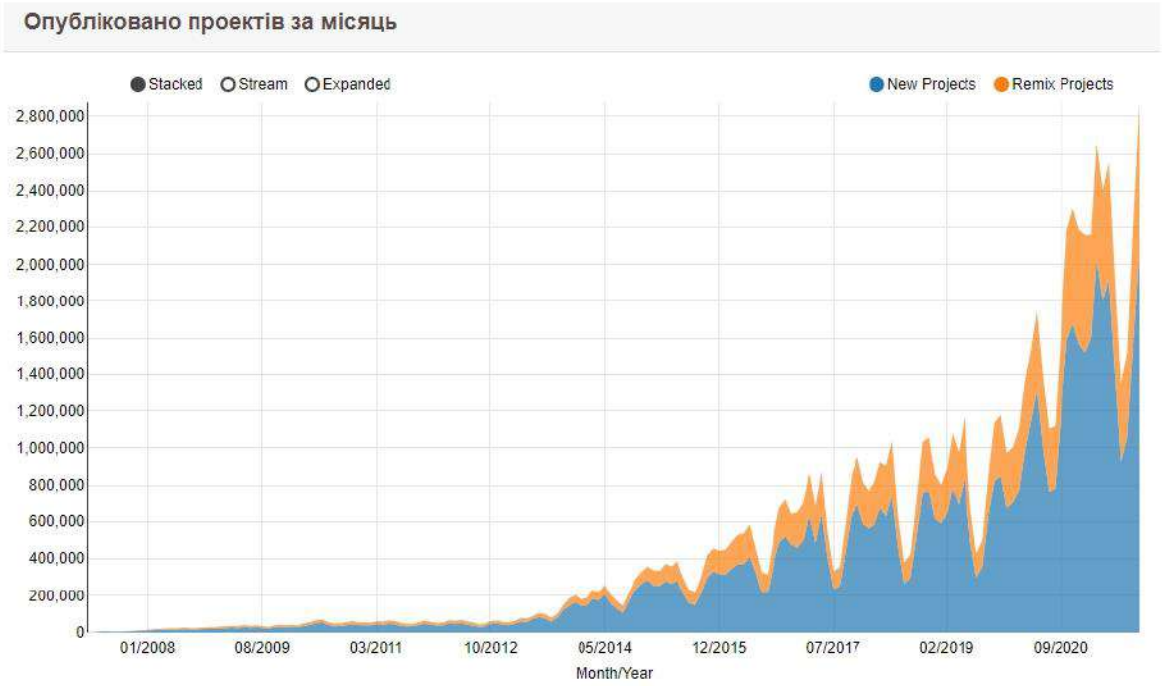


Рис. 1.3. Оновлення ресурсу Scratch новими проєктами

У 2014 році вийшла версія Scratch для дітей молодшого віку під назвою ScratchJr. Це мобільний додаток для Android та iOS, в якому діти мають можливість керувати спрайтами, але у спрощеній формі: у блоках не використовується текст, дітям доступний обмежений набір дій (прості рухи спрайтів і робота зі звуками та зображеннями).

Scratch є базовою для декількох інших візуальних мов програмування. Найвідомішою з них є Snap!, основними особливостями якої є можливість створення власних блоків і використання об'єктів першого класу, тому цю мову можна використовувати для навчання людей старшого віку. На основі Scratch також здійснюють програмування Arduino, що робить цей процес більш простим.

Існує два способи роботи в середовищі Scratch. Найбільш простий спосіб – робота в редакторі Scratch онлайн (https://scratch.mit.edu/projects/editor/) (рис. 1.4). Для того щоб мати можливість зберігати створені проєкти, необхідно зареєструватися.

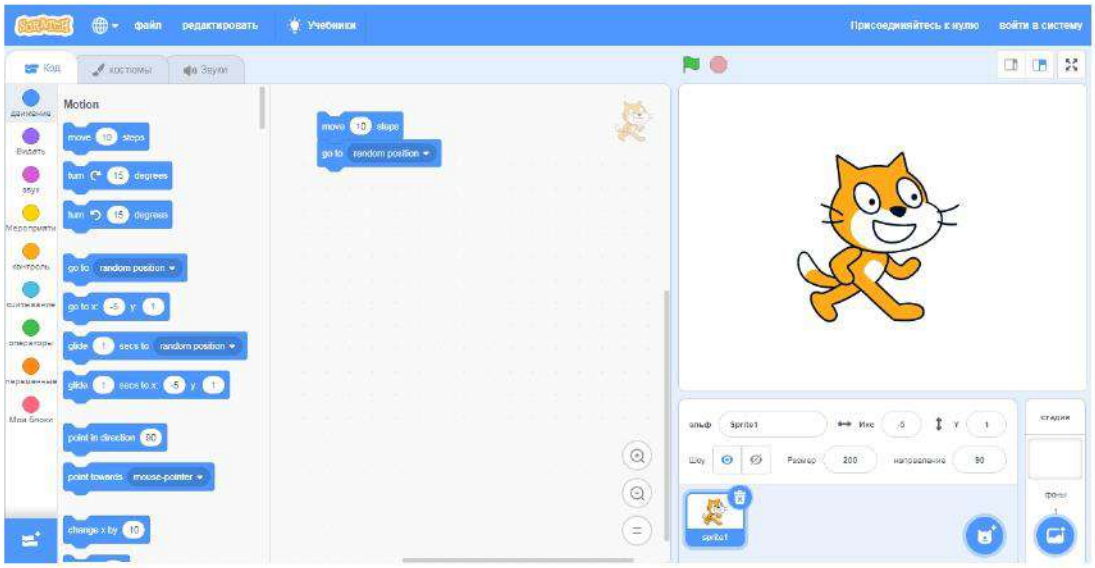


Рис. 1.4. Інтерфейс онлайн-редактора Scratch

Другий спосіб роботи – робота в редакторі офлайн, який можна завантажити зі сторінки https://scratch.mit.edu/scratch2download/. Scratch підтримується на операційних системах Windows, Linux і Mac OS X.

Обидва редактори Scratch повністю ідентичні і мають однакові функції. Проєкти, створені в редакторі офлайн, можна завантажити на сайт, і навпаки, проєкти, створені онлайн, можна завантажити на свій комп'ютер.

У середовищі Scratch можна створювати ігри, мультфільми або інтерактивні історії.

Серед переваг цього середовища можна виділити наочність створення та роботи програм, низький «поріг входження» – створення програм дається відносно легко як учням з гуманітарним нахилом, так і учням з добре розвиненими алгоритмічними навичками.

До недоліків можна віднести відсутність можливостей створення прикладних програм, що працюють незалежно від середовища, та відсутність можливостей роботи на відносно низькому рівні (пряма взаємодія з файлами та пам'яттю, графічною підсистемою ОС та ін.).

Хоча з даним середовищем програмування дітей можуть знайомити в початкових класах ЗЗСО, але його можна більш детально вивчати і в старшій школі для поглиблення знань з ІТ та створення більш складних проєктів.

Visual Basic

Мова програмування Visual Basic досить довгий час вивчається у школах. Вона рекомендована до вивчення в деяких підручниках [12].

Microsoft Visual Basic – це засіб розробки програмного забезпечення, який створений у 1991 році і який підтримується корпорацією Microsoft.

Мова програмування Visual Basic є діалектом мови програмування Basic і поєднує в собі процедури, елементи об'єктно-орієнтованих та компонентно-орієнтованих мов програмування. Середовище розробки містить інструменти для візуального конструювання користувацького інтерфейсу (рис. 1.5).

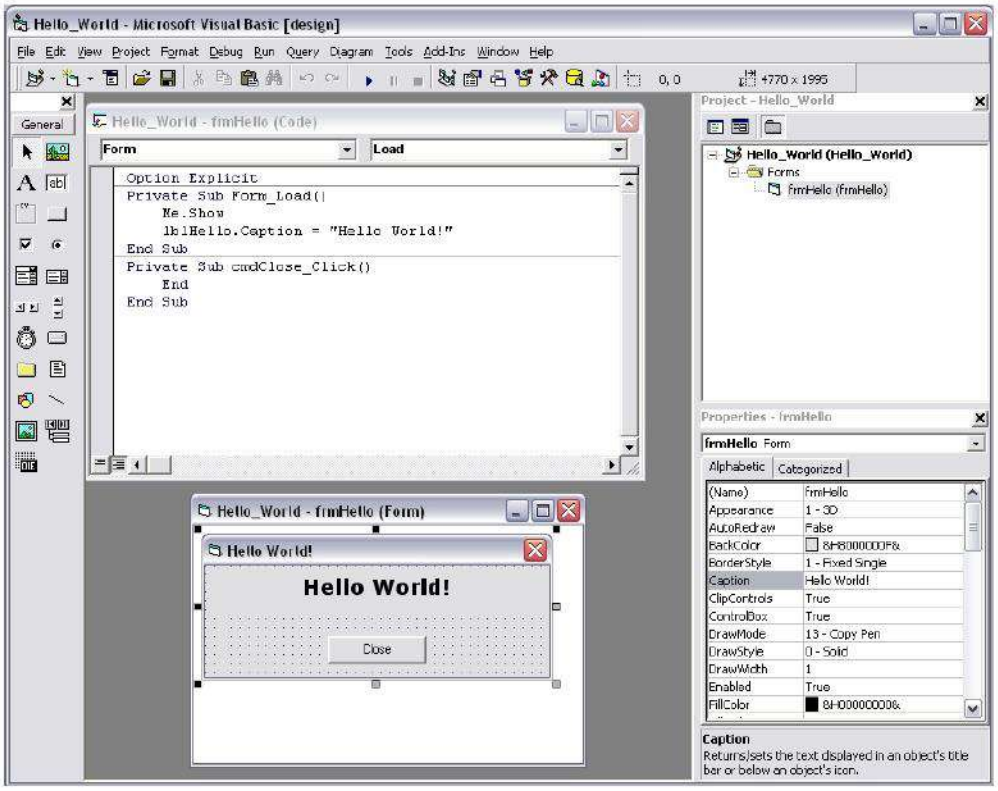


Рис. 1.5. Програма мовою програмування Visual Basic

Мова програмування Visual Basic вважається потужним засобом швидкої розробки прототипів програм, програм, що працюють з базами даних і програм, що працюють під управлінням майже усіх версій операційних систем сімейства Microsoft Windows [3].

Ця мова позиціонується як об'єктно-орієнтована, але в ній немає механізму успадкування класів. Тому необхідно явно записувати всі методи. Крім цього, в мові програмування Visual Basic відсутні вказівники, низькорівневий доступ до пам'яті та ASM-включення. Також розроблена програма вимагає встановлених бібліотек з динамічним зв'язком, що досить часто стає предметом критики. Мова програмування Visual Basic не є кросплатформною мовою програмування [12].

Як і у всіх сучасних системах візуального проєктування, у Visual Basic застосовується об'єктно-орієнтований підхід до програмування. Будь-який додаток, написаний на Visual Basic, являє собою сукупність об'єктів.

Все це дозволяє використовувати дану мову при вивченні програмування в ЗЗСО, не зважаючи на те, що вона вже є застарілою.

Pascal

Pascal – це одна з найбільш відомих мов програмування [11], які використовуються для навчання програмувати. Pascal є базою для низки інших мов, серед яких Ада, Модула-2, Delphi.

Мова Pascal була створена Ніклаусом Віртом в 1968-1969 роках після його участі в роботі комітету розробки стандарту мови Алгол-68. Мова названа на честь французького математика, фізика, літератора і філософа Блеза Паскаля, який створив першу в світі механічну машину, що вміла додавати два числа. Перша публікація Вірта про мову датована 1970 роком. Представляючи Pascal, автор в якості мети її створення вказував побудову невеликої та водночас ефективної мови, що впливає на стиль програмування та використовує структурне програмування і структуровані дані [37].

Особливостями мови є строга типізація та наявність засобів структурного програмування. Pascal була однією з перших таких мов [32].

Деякі недоліки Pascal не проявляються або навіть є перевагами в навчанні програмувати. До 1980-х років Pascal був основою для створення навчальних програм, в окремих випадках на його основі було створено спеціалізовані навчальні мови програмування. Так, на початку 1980-х років в СРСР для навчання школярів основам інформатики та обчислювальної техніки Андрій Єршов розробив паскалеподібну «навчальну алгоритмічну мову» [28].

Найбільш відомою реалізацією Pascal, що забезпечила широке поширення і розвиток мови, є Turbo Pascal фірми Borland, що виросла потім в об'єктний Pascal для DOS (починаючи з версії 5.5), потім Windows і далі в Delphi, де були впроваджені значні розширення мови [15].

На сьогднішній день більшість програмістів, що пишуть програми мовою Pascal, використовують програму Pascal ABC (рис. 1.6). Середовище програмування Pascal ABC використовується як початкове для навчання програмування школярів мовою програмування Паскаль. Середовище містить

потужну довідкову систему та вбудований задачник з автоперевіреними завданнями.

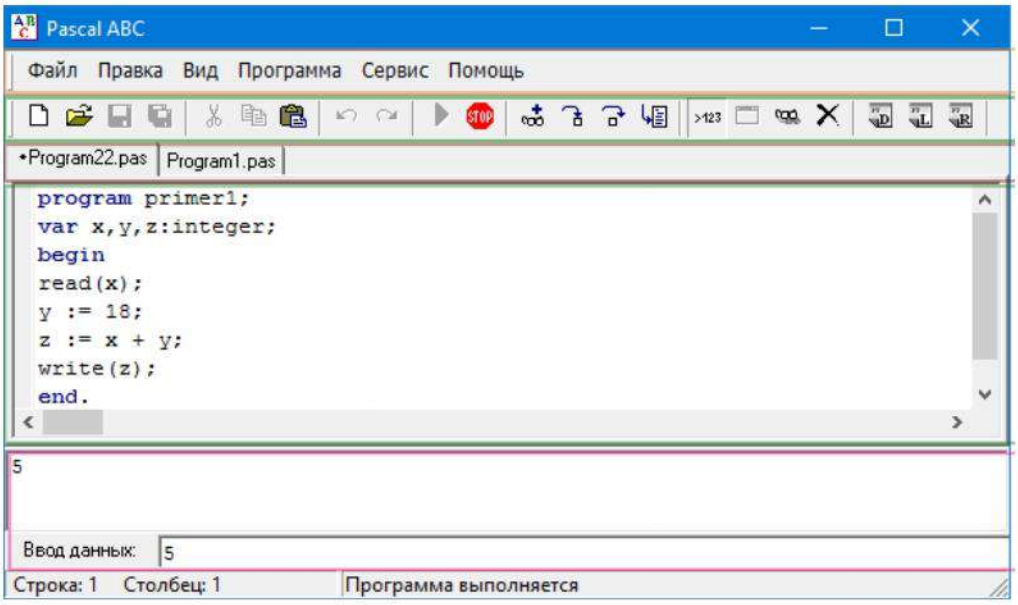


Рис. 1.6. Програма, написана мовою Pascal у середовищі програмування Pascal ABC

Не зважаючи на відносну простоту, мова виявилась придатною для розв'язування широкого кола задач, у тому числі для розробки досить великих та складних проєктів, зокрема, операційних систем.

Мова програмування Pascal є одним з найпопулярніших засобів для розвитку логічного мислення старшокласників: за аналізом науково-методичних джерел [15; 18] можна стверджувати, що базовою платформою для навчання змістової лінії основ алгоритмізації, є процедурні мови програмування, зокрема й Pascal, і саме ця мова програмування була створена Н. Віртом для опанування основ алгоритмізації та програмування, є оптимальною та зручною для навчання програмувати.

Object Pascal (Delphi)

Мова програмування Object Pascal бере свій початок від класичної мови Pascal. Вона була об'єктно-орієнтованим розширенням Pascal, розробленим компанією Apple. Існує кілька діалектів Object Pascal, одним із яких є Delphi.

Кожен діалект Object Pascal додає нові елементи та реалізує ті що є дещо по-іншому.

Оскільки Delphi є зареєстрованим товарним знаком у багатьох країнах, більшість сумісних країн посилаються на мову як Object Pascal, навіть після того, як компанія Borland (власник мови) перейменувала її. За словами Borland, перейменування було зроблено головним чином тому, що Object Pascal прирівнювався до застарілої версії. Деякі програмісти наголошували, що це тому, що Object Pascal не може бути товарним знаком, і тому перейменування було антиконкурентним заходом.

За допомогою мови Object Pascal можна виробляти прості розрахунки, розробляти програми для проведення складних математичних (рис. 1.7), інженерних та економічних обчислень з використанням візуального середовища розробника [14].

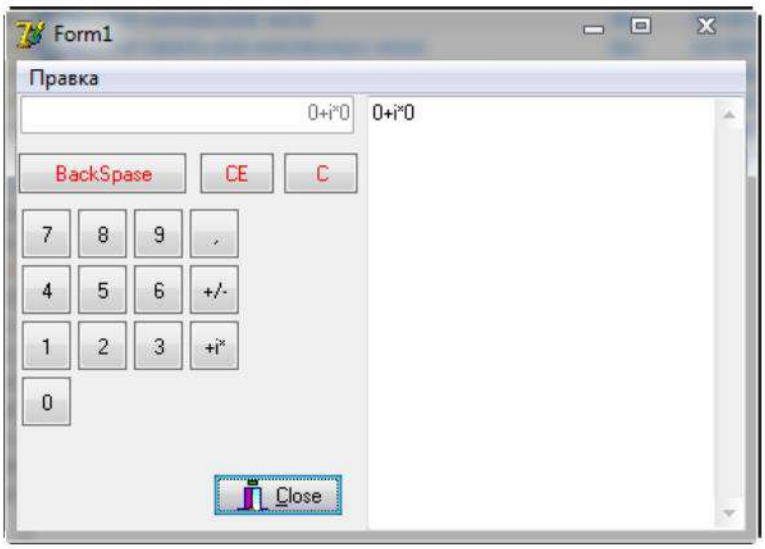


Рис. 1.7. Розроблений мовою Object Pascal калькулятор комплексних чисел

Object Pascal – це строго типізована мова високо рівня, яка підтримує структурне і об'єктно-орієнтоване проєктування [28]. Перевагою мови є легке читання коду, швидка компіляція, і використання декількох модульних файлів для модульного програмування.

C++

Мова C++ статично типізована мова програмування загального призначення. Програми, написані цією мовою, потребують попередньої компіляції перед виконанням. C++ підтримує і процедурне програмування, і об'єктно-орієнтоване програмування, і узагальнене програмування. Ця мова має досить багату стандартну бібліотеку, яка включає в себе поширені контейнери і алгоритми введення-виведення, регулярні вирази, підтримку багатозадачності та інші можливості.

Мова програмування C++ поєднує властивості як високорівневих, так і низькорівневих мов [6]. У порівнянні з мовою програмування C більшу увагу приділено підтримці об'єктно-орієнтованого і узагальненого програмування [8].

Мова програмування C++ (рис. 1.8) є однією з найпопулярніших, що широко використовується для розробки програмного забезпечення. Галузі її застосування включають розробку операційних систем, різноманітних прикладних програм, драйверів пристроїв, додатків для вбудованих систем, високопродуктивних серверів та розважальних програм.

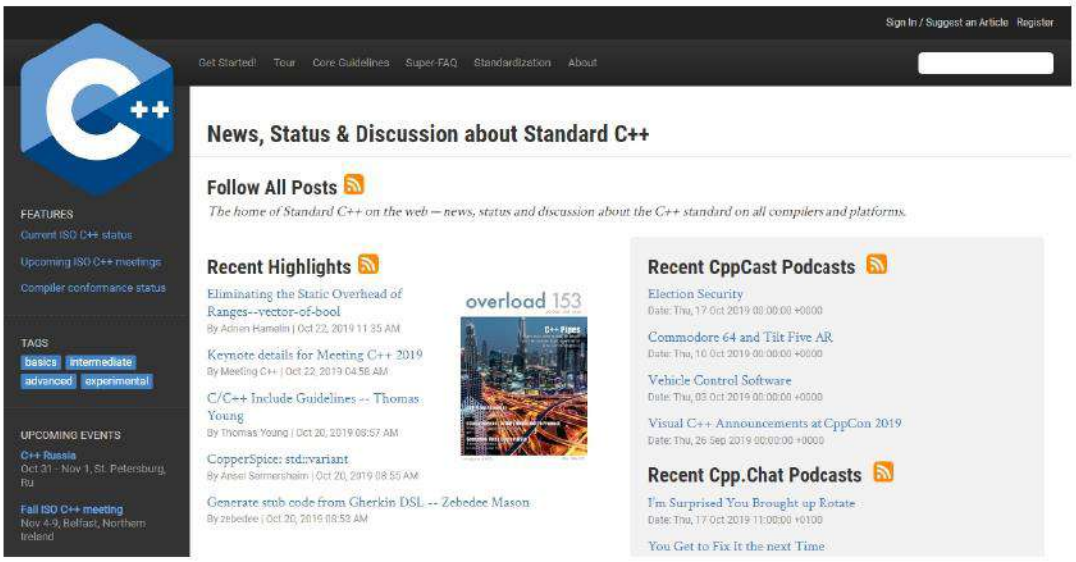


Рис. 1.8. Веб-сторінка розробника мови програмування C++

Натепер існує досить велика кількість реалізацій мови програмування C++: і безкоштовних, і комерційних, і для різних платформ. Наприклад, GCC, Visual C++, Intel C++ Compiler, Embarcadero (Borland) C++ Builder і інші [6].

Мова програмування C++ значним чином вплинула на інші мови програмування, в першу чергу на Java та C#.

Синтаксис мови програмування C++ успадкований від мови програмування C. Одним з принципів розробки було збереження сумісності з попередником. Проте, C++ не є розширеною версією C – кількість програм, які можуть однаково успішно оброблятися як компіляторами C, так і компіляторами C ++, досить велика, але не включає всі можливі програми на C.

Доступ до можливостей стандартної бібліотеки мови C++ забезпечується за допомогою включення в програму (за допомогою директиви #include) відповідних стандартних заголовкових файлів. Всього в стандарті C++ визначено 79 таких файлів. Засоби стандартної бібліотеки оголошуються в просторі імен std [35].

Мова програмування C++ – дуже потужний засіб розробки компільованого програмного забезпечення різного рівня складності [35]. Для цієї мови створено досить багато середовищ, що спеціалізуються на розробці програмного забезпечення для певної платформи – від мікроконтроллерів до потужних серверів.

Python

Мова програмування Python (рис. 1.9) – засіб розробки програмного забезпечення різного рівня складності. Програми, що написані за допомогою цієї мови, вимагають встановленої на комп'ютері користувача відповідної версії інтерпретатора. Для Python створено досить багато середовищ розробки програм.

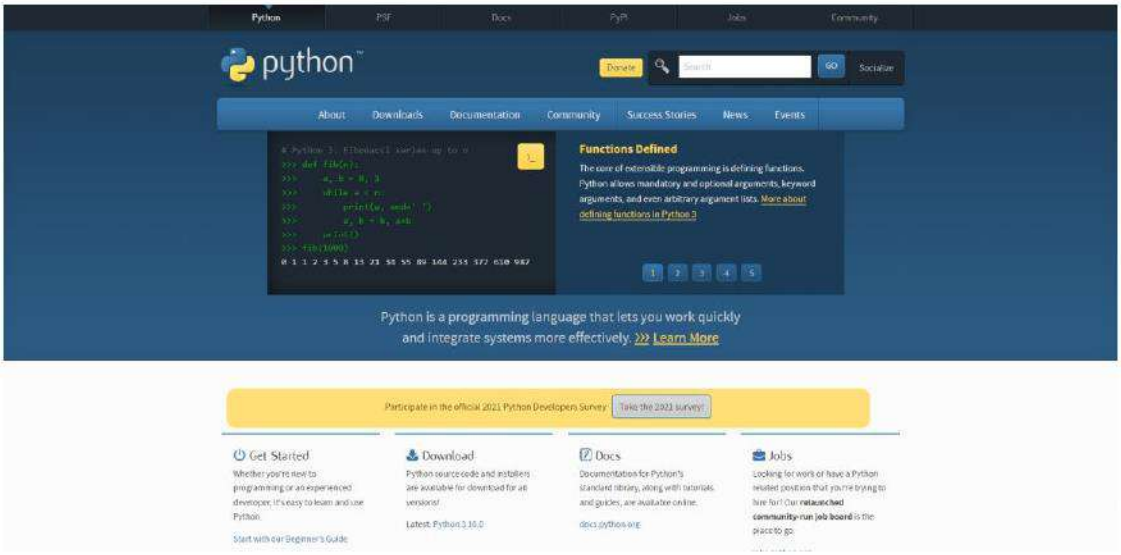


Рис. 1.9. Веб-сторінка розробника мови програмування Python

Мова Python постійно розвивається, що підтверджується статистикою оновлення випуску Python – 20 релізів за 2021 рік (дані станом на 20 листопада 2021) [1].

Розробка мови Python започаткована в кінці 1980-х років [10] співробітником голландського інституту CWI Гвідо ван Россумом.

Мова програмування Python працює майже на всіх відомих платформах UNIX (включаючи FreeBSD і Linux), Plan 9, Mac OS, Windows Mobile, Symbian, Android та ін. [4].

Завдяки своїй лаконічності Python дозволяє опанувати синтаксис мови, а відсутність «успадкування» у вигляді аксіом, що формувалися протягом багатьох років, сприяє освоєнню об'єктно-орієнтованого програмування.

Особливістю мови Python є легкість інтегрування з наявними компонентами, що дозволяє впроваджувати Python у написані програми.

Серед переваг Python [10]:

– висока швидкість виконання програм;

– різні стилі програмування Python: об'єктно-орієнтований, процедурний, функціональний;

– наявність у стандартних бібліотеках Python засобів для роботи з електронною поштою, протоколами Інтернету, ftp, http, базами даних тощо;

– виконання скриптів, написаних за допомогою Python, на більшості сучасних ОС. Така кросплатформеність забезпечує Python застосування в різноманітних галузях;

– Python підходить для будь-яких завдань у галузі програмування (офісні програми, веб-додатки тощо);

– мова Python відкрита для всіх бажаючих.

Зазначимо, що у деяких підручниках з інформатики рекомендується до вивчення мова програмування Python. Так, наприклад, у підручнику за 10 клас старшої школи за редакцією груп авторів під керівництвом В.Д. Руденко, Н.В. Речич, В.О. Потієнко [31] йде опис саме мови Python.

Java

Мова програмування Java – це об'єктно-орієнтована мова, що розробляється компанією Sun Microsystems з 1991 року (пізніше придбана компанією Oracle). Дата офіційного випуску – 23 травня 1995 року.

Мова Java (рис. 1.10) зародилася як частина проєкту створення передового програмного забезпечення для різних побутових приладів. Реалізація проєкту була започаткована мовою C++ [34].

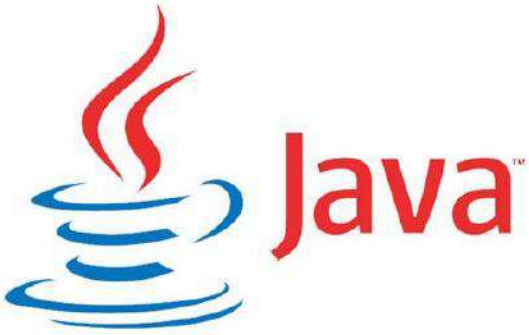


Рис. 1.10. Логотип Java

Програми на Java можуть бути трансльовані в байт-код, який виконується на віртуальній java-машині (JVM) програмою, що обробляє байт-код і направляє інструкції обладнанню, як інтерпретатор, але з тією

відмінністю, що байт-код, на відміну від тексту, обробляється значно швидше [22].

Серед переваг програмування мовою Java можна виділити наступні [9]:

– об'єктно-орієнтоване програмування Java включає в себе концепцію, в якій можна визначити тип даних, його структуру, а також набір функцій, що застосовуються до нього;

– java – мова високого рівня з простим синтаксисом. Java більш схожа на людську мову на відміну від мов низького рівня, які нагадують машинний код. Мови високого рівня перетворюються за допомогою компіляторів або інтерпретаторів. Це спрощує розробку, роблячи мову більш легкою для написання, читання і обслуговування;

– незалежність від платформи – описує кросплатформні можливості Java. Можна створити Java-додаток на Windows, скомпілювати його у байт-код і запустити на будь-якій іншій платформі, яка підтримує віртуальну машину Java;

– java – мова для розподіленого програмування і комфортної віддаленої спільної роботи. Вона має вбудований механізм спільного використання даних і програм декількома комп'ютерами, що підвищує продуктивність і ефективність процесу розробки.

Серед недоліків мови Java виділимо [9]:

– платне комерційне використання – компанія Oracle оголосила, що з 2019 року почне стягувати плату за використання Java Standard Edition 8 у «комерційних цілях». За усі оновлення та виправлення помилок доведеться заплатити. Плата залежить від кількості користувачів або комп'ютерів;

– низька продуктивність – у будь-якої мови високого рівня досить низька продуктивність через компіляції та абстракції за допомогою віртуальної машини;

– відсутність нативного дизайну – для створення графічного інтерфейсу користувача розробники використовують різні інструменти, орієнтовані для конкретної мови;

– багатослівний і складний код – це може здатися перевагою, яка допоможе при вивченні мови. Однак, довгі, надмірно складні речення ускладнюють читання і перегляд коду.

Ruby

Ruby – інтерпретована, повністю об'єктно-орієнтована мова програмування з чіткою динамічною типізацією. Вона поєднує в собі Perl-подібний синтаксис з об'єктно-орієнтованим підходом. Окремі риси запозичені з мов програмування Python, Lisp, Dylan і CLU. Кросплатформена реалізація інтерпретатора мови Ruby поширюється на умовах відкритого програмного забезпечення. Код, написаний на Ruby, може бути зрозумілий навіть людині, яка не розбирається в програмуванні.

Ruby (рис. 1.11) – це ретельно збалансована мова. Її творець Юкіхіро Мацумото (також відомий як «Matz») об'єднав частини своїх улюблених мов (Perl, Smalltalk, Eiffel, Ada і Lisp), щоб сформувати нову мову, в якій парадигма функціонального програмування збалансована принципами імперативного програмування.

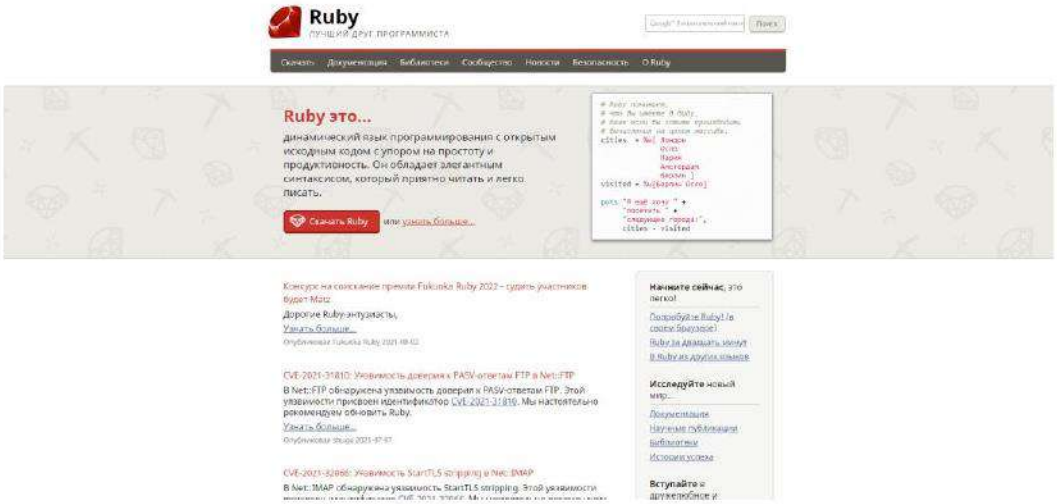


Рис. 1.11. Веб-сторінка розробника мови програмування Ruby

Розробник Ruby часто повторював, що «намагається зробити Ruby природньою, але не простою мовою, яка відображає життя» [26].

Мова програмування Ruby вважається однією з кращих для опанування першої мови програмування. Швидкий цикл розробки (редагування – запуск – редагування), використання інтерпретатора, початкова об'єктна орієнтованість, нетипізовані змінні, які не вимагають оголошення, – все це дозволяє учням сконцентрувати свою увагу на загальних принципах програмування.

Не менш важливі мультиплатформеність Ruby і її належність до категорії вільно поширюваного програмного забезпечення. Ще один вагомий аргумент на її користь – можливість практичного використання мови у різноманітних галузях [36].

Отже, у ЗЗСО для вивчення пропонуються мови Scratch, Visual Basic, Pascal, Object Pascal, C++, Python, Java, Ruby.

Аналіз мови програмування, які можуть бути рекомендовані для вивчення у шкільному курсі інформатики (Scratch, Visual Basic, Pascal, Object Pascal, C++, Python, Java, Ruby) дає підстави охарактеризувати їх за певними параметрами (табл. 1.3).

На рис. 1.12 видно, що не зважаючи на рік появи першої версії тієї чи іншої мови програмування, більшість з мов залишаються затребуваними у сучасному світі.

Додатково наведемо рейтинг мов програмування за статистичними даними ресурсу TIOBE [11], який розраховується за кількістю запитів до пошукових систем, що містять назву мови.

Розглянемо статистичні дані рейтингу мов програмування TIOBE станом на листопад 2021р. (рис. 1.13)

	Scratch	Visual Basic	Pascal	Object Pascal	C++	Python	Java	Ruby
Тип мови	Графічна, навчальна	Процедурна, об'єктно-орієнтована, подієво-орієнтована	Імперативна, структурована	Об'єктно-орієнтована, загальна, процедурна	Об'єктно-орієнтована, мультипарадигмальносна, процедурна, компільована	Об'єктно-орієнтована, рефлективна, імперативна, функціональна	Мультипарадигмальна, мова JVM	Об'єктно-орієнтована
Тип виконання	Інтерпретуючий	Компілюючий, інтерпретуючий	Компілюючий	Компілюючий	Компілюючий	Інтерпретуючий	Інтерпретуючий	Інтерпретуючий
Рік появи	2007	1991	1970	1986	1983	1991	1995	1995
Останні оновлення	3.0 (02.01.2019)	6.0 (1998)	7.1 (03.1994)	10.3.2 Rio (18.07.2019)	C++17 (12.2017)	3.7.17 (19.10.2019)	Java SE 12 (03.2019)	2.6.5 (01.10.2019)
Автор/ розробник	Мітчелл Резник / MIT Media Lab	Майкрософт	Ніклаус Вірт	Ларрі Теслер	Страуструп Бйорн	Python Software Foundation і Гвідо ван Россум	Джеймс Гослінг і Sun Microsystems	Мацумото, Юкіхіро
Розширення файлів	.sb , .sb2 , .sb3	.bas, .cls, .frm, .vbp, .vbg	.pas, .inc	.p, .pp, .pas	.cc, .cpp, .cxx, .c, .c++, .h, .hpp, .hh, .hxx, h++	.py, .pyw, .pyc, .pyo, .pyd	.java, .class, .jar	.rb чи .rbw
Система типів	Динамічна	Статична, сувора, динамічна	Статична, сильна, безпечна	Сильна, динамічна	Статична	Сильна, динамічна	Сильна, динамічна	Строга, динамічна
Основні реалізації	Scratch	Microsoft Visual Studio	Free Pascal, Turbo Pascal, PascalABC.NET	Lazarus, Delphi, Virtual Pascal	Microsoft Visual C++, Intel C++, Embarcadero C++ Builder, Turbo C++, DevC++	CPython, Jython, IronPython, PyPy, Stackless	C++, C, Ада, Simula, Smalltalk, Object Pascal, Оберон, Eiffel, Модула-3, Mesa	Ruby MRI, JRuby,Rubinius
Підготовчі дії до розвитку мови	Snap!, AppInventor, Pocket Code	Visual Basic .NET, REALbasic, Gambas, Xojo, Basic4ppc	Ада, Object Pascal, Java, Oxygene	C#, Java, Nim	C, Сфмула, Алгол 68, Клу, ML, Ада	Ruby, Boo, Groovy, ECMAScript, CoffeeScript, Swift, Nim	-	-

Рис. 1.12. Порівняльна характеристика мов програмування

Nov 2021	Nov 2020	Change	Programming Language	Ratings	Change
1	2	↑	Python	11.77%	-0.35%
2	1	↓	C	10.72%	-5.49%
3	3		Java	10.72%	-0.96%
4	4		C++	8.28%	+0.69%
5	5		C#	6.06%	+1.39%
6	6		Visual Basic	5.72%	+1.72%
7	7		JavaScript	2.66%	+0.63%
8	16	⇈	Assembly language	2.52%	+1.35%
9	10	↑	SQL	2.11%	+0.58%
10	8	↓	PHP	1.81%	+0.02%
11	21	⇈	Classic Visual Basic	1.56%	+0.83%
12	11	↓	Groovy	1.51%	-0.00%
13	15	↑	Ruby	1.43%	+0.22%
14	14		Swift	1.43%	+0.08%
15	9	⇊	R	1.28%	-0.36%
16	12	⇊	Perl	1.22%	-0.29%
17	18	↑	Delphi/Object Pascal	1.22%	+0.36%

Рис. 1.13. Рейтинг популярності мов програмування за даними індексу TIOBE станом на листопад 2021

Згідно з даними на рис. 1.13 можна бачити, що описані нами мови програмування мають високі рейтинги. Так, лідером рейтингу є мова Python (11,77%), третю позицію займає мова програмування Java (10,72%), четверте – C++ (8,28%), а далі Visual Basic (1,56%) і Ruby (1,43%) займають 6 та 13 місця відповідно. Мова Object Pascal (1,22%) утримує 17 місце. Мова Scratch посідає 24 місце.

Більш повний перелік рейтингу мов програмування подано у додатку А.

Станом на листопад 2021 р. за останній рік мови C++, Ruby та Visual Basic та Object Pascal піднялися у рейтингу, а мови Python та Java знизили свої позиції.

На рис. 1.14 відображено графік популярності деяких мов програмування за рейтинговою статистикою ресурсу TIOBE [11].

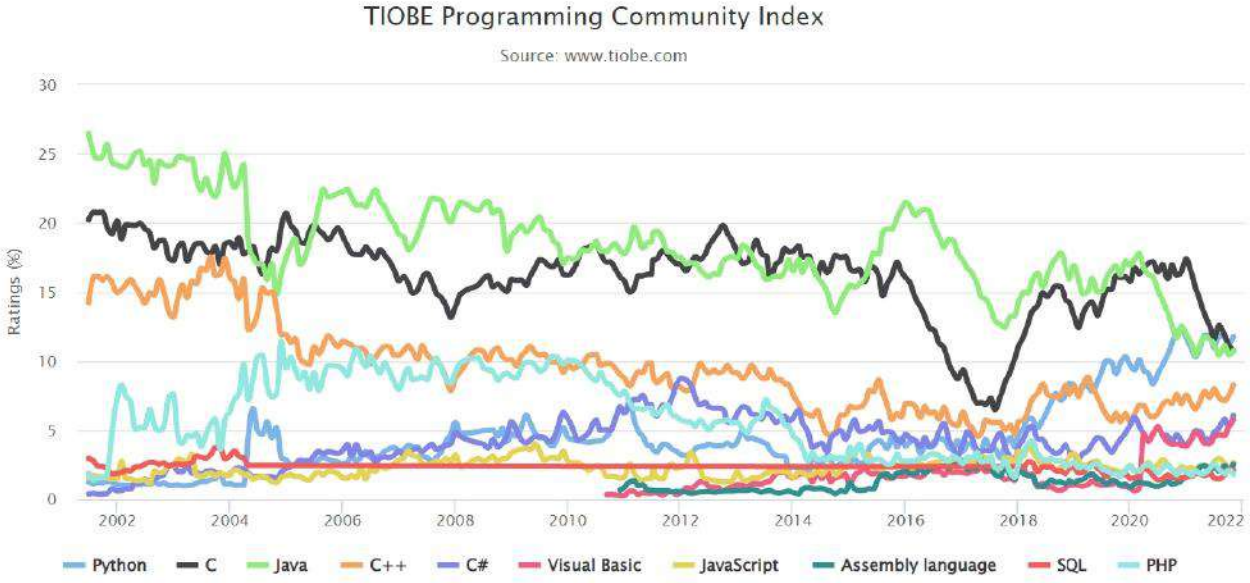


Рис. 1.14. Популярність найрентинговіших мов програмування (2002-2021 роки)

Бачимо, що майже до 2020 року постійним лідером серед мов програмування була мова Java. А до 2021 року ще й мова C. Але варто виділити швидке зростання рейтингу мови Python, яка, починаючи з 2018 року, вийшла на провідні позиції. Станом на січень 2018 року Python мав рейтинг 4,68%, а вже станом на сьогодні (листопад 2021) – 11,77%.

За даними ресурсу TIOBE найбільш популярними є Python, Java та C.

Отже, констатуємо популярність на початок 2021 року мов Python, Java та C, а для вивчення в школі – Scratch, Object Pascal, C++, Python, Java та Ruby.

1.2. Типи циклів та їх унаочнення для формування навичок організації циклічних обчислень

Програма, написана будь-якою мовою, складається з певного коду. Зазвичай він виконується послідовно: рядок за рядком, зверху донизу. Але є такі конструкції коду, які змінюють лінійне виконання програми. Їх називають керуючими конструкціями.

Завдяки таким конструкціям, код можна виконувати вибірково. Наприклад, запустити один блок коду замість іншого.

Одним із видів керуючих конструкцій в програмуванні є циклічні конструкції або цикли.

Цикли – це різновид керуючих конструкцій для організації багаторазового виконання однієї й тієї ж ділянки коду.

Код усередині такої конструкції виконується циклічно (повторюється). Кожне виконання коду – це ітерація циклу. Кількість ітерацій регулюється умовою циклу. Код, який виконується усередині циклу, називають тілом циклу.

Відомі такі види циклів (рис. 1.15).

Рис. 1.15. Види циклічних конструкцій

Цикли з передумовою (цикли «Поки») – це цикли в яких умова виконання визначається перед першою ітерацією.

Робота циклів з передумовою заключається у наступному.

Обчислюється значення виразу (тобто умова), що має бути логічним виразом. Якщо результат обчислення виразу істинний, то виконується тіло циклу (простий або складовий оператор). Потім знову перевіряється умова і

т.д. Якщо результат хибний, то відбувається вихід із циклу і керування передається першому оператору, який є наступним за циклом.

Блок-схема циклу з передумовою зображена на рис. 1.16.

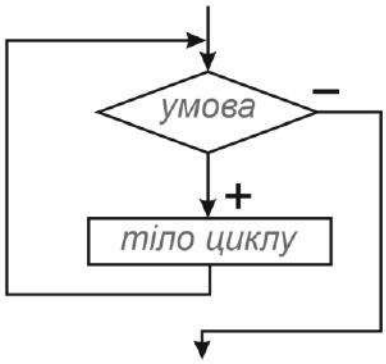


Рис. 1.16. Цикл з передумовою

Цикли з післяумовою (цикли «До») виконання визначається після першої ітерації. Саме такі види завжди виконуються мінімум один раз. Цикли з умовою актуальні тоді, коли потрібно виконати певну дію, доки реалізується певна умова: наприклад, зчитувати введені користувачем дані, доки він не введе слово “stop”.

Робота циклу з післяумовою відбувається так.

Виконується послідовність операторів, що зазначені на початку циклу (тому тіло циклу виконається хоча б один раз). Далі проводиться перевірка продовження циклу: якщо значення записаного виразу дорівнює false (хибне значення), тіло циклу виконується знову. Якщо значення виразу дорівнює true (істинне), відбувається вихід із циклу.

Схематично цикл з післяумовою можна бачити на рис. 1.17.

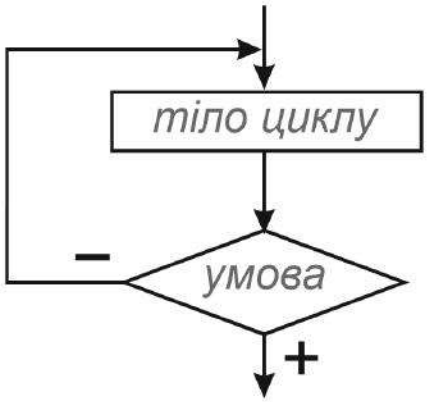


Рис. 1.17. Цикл з післяумовою

Цикли з лічильником (цикли «Для») – кількість ітерацій визначається змодельованим лічильником. В умові циклу задається його початкове та кінцеве значення. Кожну ітерацію лічильник збільшує. За допомогою циклів з лічильниками можна наперед визначити кількість ітерацій.

Робота таких циклів базується на виконанні певних дій ту кількість разів, яка записана в умові циклу. Умовою циклу слугую лічильник, якому наперед задано кількість повторів і він тільки контролює цю кількість.

Ці цикли бувають корисними, коли потрібно перебрати всі елементи в колекції. Цикли з лічильником називають «циклами для...» – «для кожного елемента деякої колекції здійснити такі дії».

Блок-схема циклу з лічильником показана на рис. 1.18.

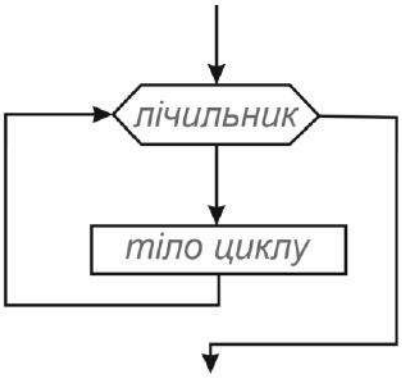


Рис. 1.18. Блок схема циклу з лічильником

Допустимі випадки, коли виконання циклу можна перервати до досягнення його умови. Наприклад, якщо є деяка колекція зі 100 чисел і

необхідно зрозуміти, чи вона містить негативні числа. Можна розпочати перебір всіх чисел, використовуючи цикл «для». Але коли буде знайдено перше негативне число, не обов'язково перебирати інші числа, що залишилися. В такому випадку можна перервати виконання циклу, якщо його подальше виконання не має сенсу. Подібні ситуації називають переривання циклу.

Безумовні цикли – цикли, що виконуються нескінченно. Наприклад: «Поки 1 = 1, друкувати "1 = 1"». Така програма виконуватиметься, доки її вручну не перервуть.

Дані цикли теж бувають корисні, коли використовуються разом із перериванням циклу «зсередини».

Розглянемо синтаксис усіх видів циклів на мовах програмування Паскаль, C++, Java та Python. Ці мови найбільше зустрічаються при вивченні інформатики в ЗЗСО.

Таблиця 1.1

Синтаксис циклічних конструкцій

	Паскаль	**C++**	**Java**	**Python**

Цикл з передумовою	**while** умова
	do
	тіло циклу;
		while
		(умова) {
		тіло циклу;
		}
			while
			(умова) {
			тіло циклу;
			}
				while умова:
				тіло циклу

Цикл з післяумовою	**repeat** тіло циклу **until** умова	**do** { тіло циклу } **while** (умова);	**do** { тіло циклу } **while** (умова);	

Цикл з лічильником	**for** параметр
	:= поч_знач
	to кін_знач
	do
	тіло циклу;
		for
		(лічильник
		= значення;
		лічильник <
		значення;
		крок циклу)
		{
		тіло циклу;
		}
			for (
			ініціалізація
			лічильника;
			[умова**];**
			[зміна
			лічильника**])**
			{
			тіло циклу
			}
				for int_var **in**
				функція_range **:**
				тіло циклу

Незважаючи на те, що цикли for і while (з лічильником та з умовою) необхідні для повторення певної кількості разів однієї і тієї ж операції, ці цикли відрізняються один від одного і мають свою специфіку. Навіть з урахуванням їхньої формальної взаємозамінності.

While зручний тоді, коли операція, що повторюється, проводиться до того часу, поки умова правильна, тобто повертає значення True. Звідси можлива ситуація, коли цикл не спрацює жодного разу або повторюватиметься нескінченно.

Цикл for застосовується для послідовного маніпулювання елементами ітератора. Іншими словами, він проходить по черзі елементи об'єкта (наприклад, списку) і закінчується після їхнього повного перебору.

Цикл for досить часто використовується для роботи з масивами. Наприклад, програмний код на мові C++ за допомогою циклу з лічильником дозволяє заповнити масив із n елементів нулями:

```
int a[n];
for (int i = 0; i < n; i++) {
   a[i] = 0;
}
```

Таким чином, for зручний для перебору, а while – для перевірки істинності умови перед кожною ітерацією.

З табл.1.1 бачимо, що синтаксис циклічних конструкцій в більшій мірі однаковий, але кожна мова програмування має свої відмінності. Незважаючи на це принцип використання циклів є однаковим у різних задачах на використання циклів.

1.3. Вивчення практичного стану проблеми навчання циклічних обчислень учнів старших класів

Магістерське дослідження на першому етапі своєї реалізації виявило зростання протиріч у бажанні вивчати програмування учнями різних класів.

Зробити такий висновок дозволив аналіз висловлювань учнів 5-11 класів щодо вивчення програмування за результатами опитувань.

Опитування проводилось серед учнів шкіл м. Суми, а саме КУ Сумська спеціалізована школа І-ІІІ ступенів № 2 ім. Д. Косаренка, КУ Сумська загальноосвітня школа І-ІІІ ступенів №6, КУ Сумська спеціалізована школа І-ІІІ ступенів №7 імені М. Савченка, КУ Сумська спеціалізована школа І-ІІІ ступенів №17, КУ Сумська спеціалізована школа І-ІІІ ступенів №25.

Запитання анкети

1. В якому класі ти навчаєшся?
2. Чи хочеш вивчати програмування в школі?
3. Чи вивчаєш програмування поза школою?
4. З якими мовами програмування ти знайомий?
5. Постав позначку там, де можна використати циклічні обчислення для визначення результату:
   a. 1+2+3+4+…+10
   b. 1-2*3-4+5
   c. 1, 1, 2, 3, 5, 8, 13, ***x*=?**
   d. 2x+3=x-5

За результатами анкетування (в опитуванні брали участь 73 учні різних класів) (рис.1.19), виявлено, що відношення до програмування дуже різне, причому воно змінюється в бік погіршення із переходом до старшої школи.

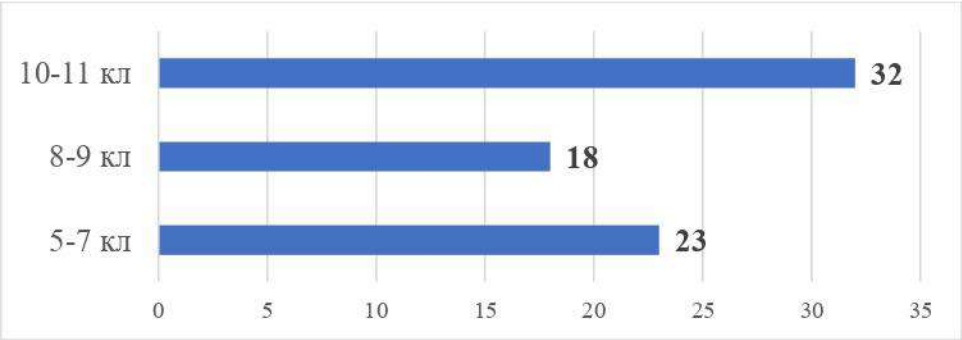


Рис. 1.19. Розподіл учасників опитування

Нижче наведені думки учнів щодо програмування.

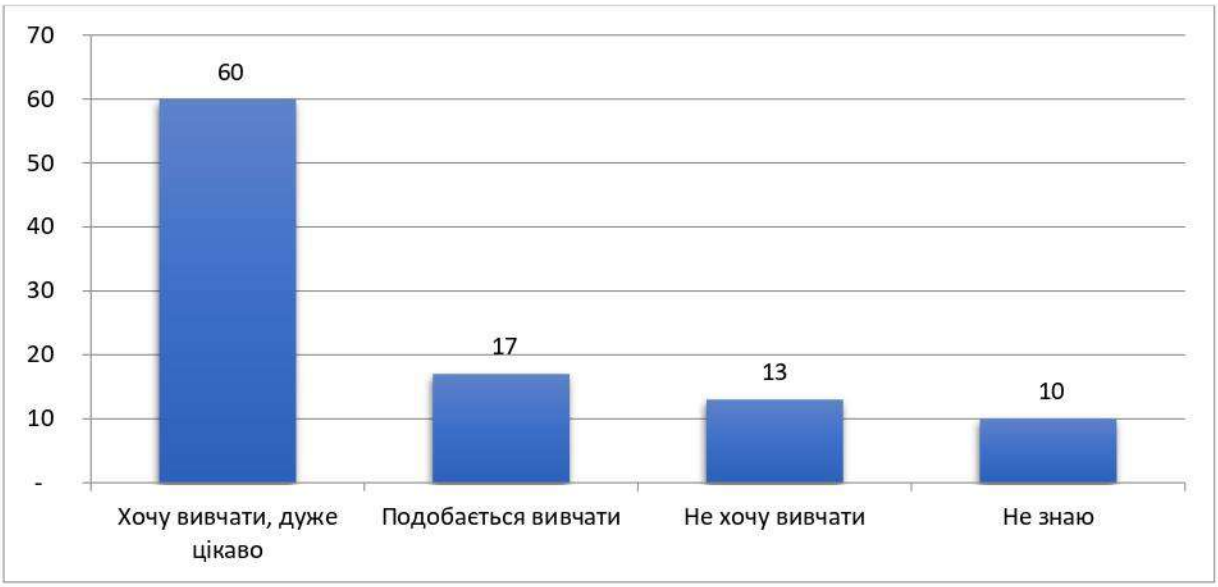


Рис. 1.20. Вислови учнів 5-6-х класів щодо ставлення до програмування (%)

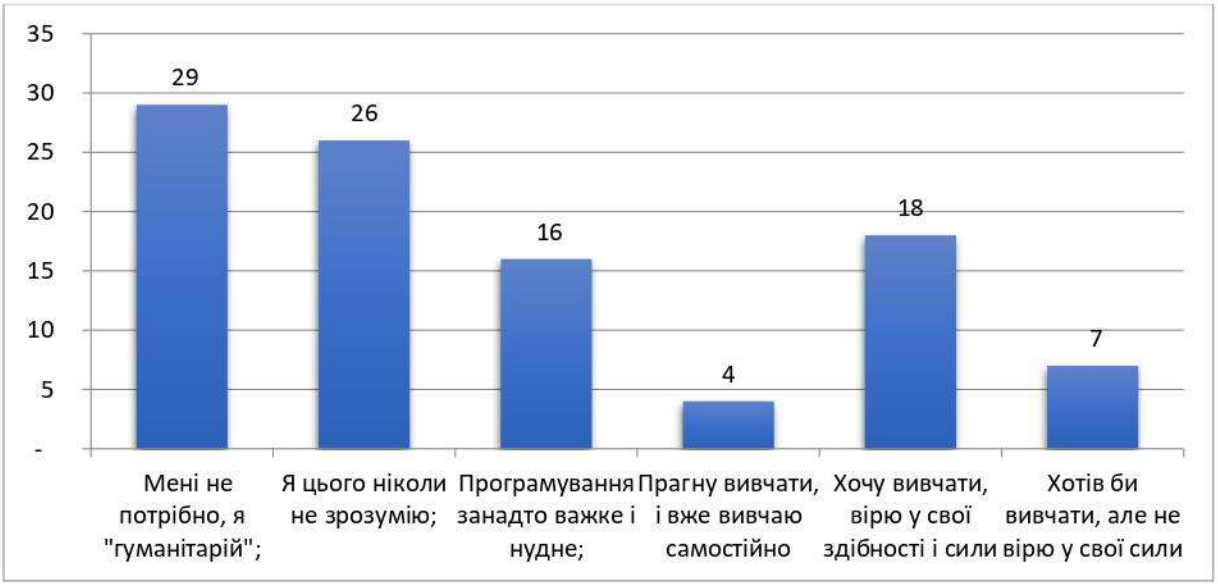


Рис. 1.21. Вислови учнів 8-9-х класів щодо ставлення до програмування (%)

На запитання «Чи вивчаєш програмування поза школою?» більшість опитаних відповіли «Ні» (табл.1.2).

Таблиця 1.2

Розподіл відповідей на запитання 3

Клас	5-7 кл	8-9 кл	10-11 кл
Кількість відповідей загалом	5	2	6
По відношенню до вікової групи	22%	11%	19%
По відношенню до загальної кількості	7%	3%	8%

На четверте запитання про обізнаність у мовах програмування маємо ситуацію, що найчастіше учнями в школі вивчається Scratch (86,7%), Python (61,3%), Object Pascal (20,0%), Pascal (12,0%), C++ (6,7%), інша (5,3%) мова програмування. Високий відсоток учнів, які знають мову Scratch, пояснюємо тим, що саме ця мова є обов'язковою для вивчення ще в початкових класах.

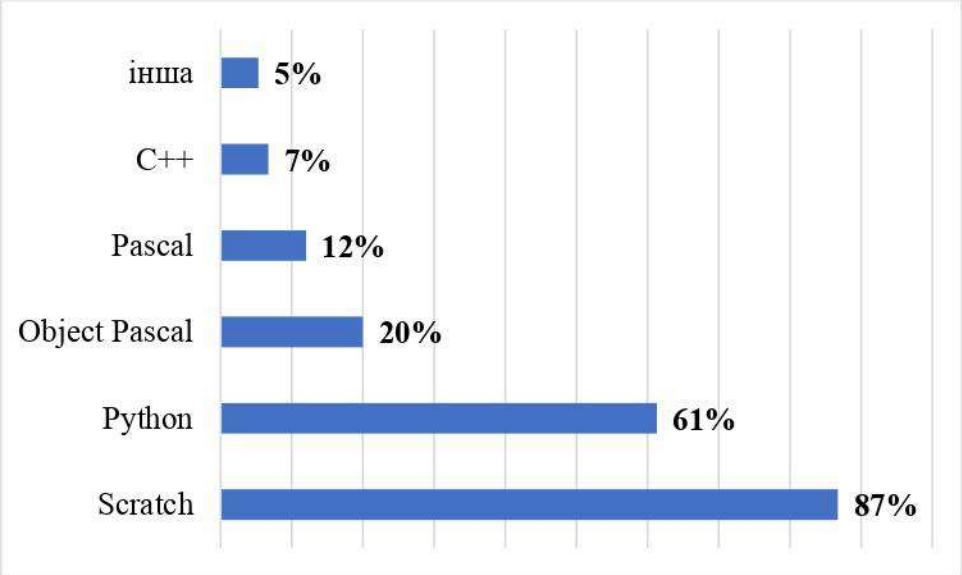


Рис. 1.22. Обізнаність учнів щодо мов програмування

На п'яте запитання про доцільність використання циклічних обчислень (лише відповіді *a* і *c* були правильними) коректно відповіли лише 9% від загальної кількості респондентів (табл.1.3).

Таблиця 1.3

Розподіл відповідей на запитання 5

Клас	5-7 кл	8-9 кл	10-11 кл
Кількість відповідей загалом	1	2	4
По відношенню до вікової групи	4%	11%	13%
По відношенню до загальної кількості	1%	3%	5%

Загалом аналіз відповідей показав, що прослідковується зниження інтересу до програмування з 60% (5-6 класи) до 22% учнів (4% прагнуть і вивчають самостійно, 18% виявили зацікавленість і хочуть вивчати). Зіставлення результатів опитування свідчить про зниження надії навчитися програмувати («я цього ніколи не зрозумію» – 26%, «програмування занадто важке і нудне» – 16%, «хотів би вивчати, але не вірю у свої сили» – 7%).

Також виявлено, що учням старшої школи притаманний низький рівень зацікавленості вивченням програмування та відсутність мотивації.

Вивчення думки вчителів щодо ситуації з небажанням вивчати програмування, зокрема, циклічних обчислень, виявила, що серед основних причин – мала кількість годин на вивчення циклів, відсутність достатньої матеріально-технічної бази, традиційні (лекційно-практичні) методи навчання, відсутність достатньої кількості завдань для опанування циклічних обчислень.

Висновки до розділу 1

У розділі розглянуто популярні мови програмування та описано особливості організації циклічних обчислень в них. Виявлено стан розробленості методичної проблеми навчання учнів організації циклічних обчислень в ЗЗСО.

Серед найбільш популярних мов програмування на початок 2021 року відзначені мови Python, Java та C, а для вивчення в школі – Scratch, Object

Pascal, C++, Python, Java та Ruby. Кожна з мов передбачає використання циклів різних типів (цикли з лічильником, з передумовою, з післяумовою).

Аналіз практичного стану розробленості проблеми виявив тенденцію зниження бажання вивчати програмування із переходом до старших класів. При цьому серед опитаних різняться думки щодо причин зниження такого бажання. Учням старшої школи притаманний низький рівень зацікавленості вивченням програмування та відсутність мотивації.

Вивчення думки вчителів щодо ситуації з небажанням вивчати програмування, зокрема, циклічних обчислень, виявила, що серед основних причин – мала кількість годин на вивчення циклів, відсутність достатньої матеріально-технічної бази, традиційні (лекційно-практичні) методи навчання, відсутність достатньої кількості завдань для опанування циклічних обчислень.

РОЗДІЛ 2.
ОСОБЛИВОСТІ ФОРМУВАННЯ НАВИЧОК ОРГАНІЗАЦІЇ ЦИКЛІЧНИХ ОБЧИСЛЕНЬ НА УРОКАХ ІНФОРМАТИКИ СТАРШОЇ ШКОЛИ

2.1. Аналіз навчальних програм і підручників з інформатики щодо навчання циклічних обчислень

На сьогоднішній день освіта знаходиться на етапі впровадження концепції Нової української школи [25] в заклади освіти. Ця концепція має на меті створити такий заклад для дітей, у якому буде приємно навчатись і який буде давати школярам не тільки знання, а й вміння застосовувати їх у житті.

Обов'язковими результатами навчання за концепцією є знання кількох іноземних мов, здатність до самоосвіти, підприємливості й ініціативності, уявлення про світобудову, дотримання здорового способу життя тощо. Для реалізації вказаної мети відповідні зміни торкнуться усіх навчальних предметів, в тому числі, інформатики, яку розглядають як один із засобів формування не тільки освітнього, а й розвивального та інтелектуального потенціалу індивідууму.

Наразі для старших класів закладів загальної середньої освіти (ЗЗСО) є чинними дві навчальні програми вибірково-обов'язкового предмету «Інформатика» [24] – рівень стандарту та профільний рівень.

Стандартний рівень вивчення інформатики у старших класах базується на двох частинах – базовий та вибірковий (варіативний) модулі. Основним для вивчення є базовий модуль, зміст якого може бути розширеним за рахунок запропонованих вибіркових модулів.

Під час вивчення розділів базового модуля відбувається завершення формування у дітей предметних і ключових компетентностей щодо використання сучасних ІКТ на визначеному чинним Державним стандартом базової і повної загальної середньої освіти [30] рівні.

Базовий модуль є основою інформатики у старших класах.

Вибіркові (варіативні) модулі розширюють курс інформатики. Учитель самостійно має можливість обирати їх, спираючись на профіль навчання закладу освіти, запитів, індивідуальних інтересів і здібностей дітей, особливостей регіону проживання, наявності у кабінеті інформатики матеріально-технічної бази та необхідного ПЗ.

Реалізація профільного навчання під час викладання курсу інформатики може здійснюватися як шляхом розширення змісту окремих тем, так і добором інших профільно-орієнтованих навчальних завдань.

За допомогою поєднання двох модулів навчання інформатики у 10-11 класах забезпечується необхідна гнучкість та свобода у відборі і комплектації необхідного матеріалу для навчання школярів і реалізації дидактичних цілей.

Особливості навчальних програм з інформатики для учнів старшої школи подано у табл. 2.1.

Таблиця 2.1.

Характеристика навчальних програм рівня стандарт та профільного рівня

Характеристика	*Рівень стандарту* [24]	*Профільний рівень* [23]
Загальний обсяг навчального плану	105 годин	350 годин
Обсяг базового модуля	35 годин	350 годин
Обсяг вибіркових модулів	70 годин (2 модулі по 35 годин)	–
Тематики базового модуля	▪ Інформаційні технології в суспільстві.	▪ Мова програмування та структури даних ▪ Сучасні інформаційні технології

Характеристика	*Рівень стандарту* [24]	*Профільний рівень* [23]
	▪ Моделі і моделювання. Аналіз та візуалізація даних. ▪ Системи керування базами даних. ▪ Мультимедійні та гіпертекстові документи.	▪ Аналіз і візуалізація даних ▪ Графіка\мультимедіа ▪ Електронні публікації ▪ Бази даних ▪ Алгоритми ▪ Веб-технології ▪ Парадигми та технології програмування
Розділи вибіркових модулів	▪ Графічний дизайн ▪ Комп'ютерна анімація ▪ Тривимірне моделювання ▪ Математичні основи інформатики ▪ Інформаційна безпека ▪ Веб-технології ▪ Основи електронного документообігу ▪ Бази даних ▪ Формальна логіка ▪ Комп'ютерні технології опрацювання звукової інформації ▪ Креативне програмування	–

За даними таблиці 2.1 бачимо, що обидва рівні мають істотну різницю в навчальному навантаженні. Це обґрунтовується тим, що для профільного

рівня інформатика вивчається по 5 годин на тиждень, 175 годин на рік, у той час, коли на рівні стандарту вивчення інформатики займає тільки 1 годину на тиждень у 10-х класах та 2 години на тиждень у 11-х класах.

Для профільного рівня відсутній вибірковий модуль, і всі основні розділи інформатики вивчаються на базовому рівні.

Стосовно переліку тем двох навчальних програм зазначимо, що окремі теми співпадають, а деякі взагалі не розглядаються. Наприклад, розділ «Інформаційні технології в суспільстві» базового модуля рівня стандарт має певну схожість з розділом «Сучасні інформаційні технології» профільного рівня; розділ «Аналіз та візуалізація даних» присутній взагалі в обох навчальних програмах; «Системи керування базами даних» базового модуля рівня стандарт є певним відбитком вибіркового модуля «Бази даних» цього ж рівня і розділу «Бази даних» профільного рівня.

На рівні стандарту елементи програмування вивчаються у вибірковому модулі «Креативне програмування» [24]. На профільному рівні [23] вивчення програмування переплітається відразу з трьома розділами: «Мова програмування та структури даних», «Алгоритми», «Парадигми та технології програмування».

Варто зазначити, що у змісті навчального матеріалу розділу, що стосується вивчення програмування, не зазначено конкретної мови програмування. Це означає, що вибір тієї чи іншої мови залишається за вчителем, і вже він розробляє конспекти уроків та інструкції до лабораторних робіт відповідно до обраної мови і змісту матеріалу, який зазначено у навчальних програмах.

Розглядаючи актуальні підручники з інформатики для старших класів ЗЗСО на предмет вивчення програмування в них та спираючись на навчальну програму можна стверджувати, що програмування можна знайти тільки у підручнику для профільних класів.

Проаналізуємо саме такий підручник авторів В. Д. Руденко, Н. В. Речич, В. О. Потієнко [31].

У даному підручнику для вивчення програмування виділяється розділ «Мова програмування та структури даних». Зміст даного розділу зосереджено у вивченні тем (рис. 2.1).

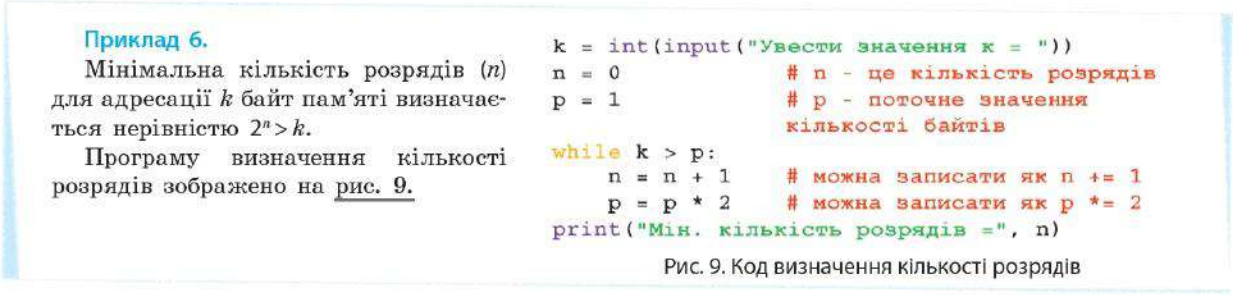
Зміст

Рис. 2.1. Зміст розділу «Мова програмування та структури даних» у [31]

Із змісту можна бачити, що автори підручника пропонують для вивчення мову програмування Python. Майже весь розділ і присвячено її вивченню.

Циклічні обчислення зосереджені у п.3.3 «Реалізація циклічних алгоритмів».

У підручнику розглядаються цикли з параметром та з умовою (перед і після). Варто зазначити, що весь теоретичний матеріал супроводжується розв'язаними прикладами (рис. 2.2). Таких прикладів з циклічними обчисленнями – 8.



Приклад 6.

Мінімальна кількість розрядів (n) для адресації k байт пам'яті визначається нерівністю $2^n > k$.

Програму визначення кількості розрядів зображено на рис. 9.

```
k = int(input("Увести значення к = "))
n = 0              # n - це кількість розрядів
p = 1              # p - поточне значення
                   кількості байтів
while k > p:
    n = n + 1      # можна записати як n += 1
    p = p * 2      # можна записати як p *= 2
print("Мін. кількість розрядів =", n)
```

Рис. 9. Код визначення кількості розрядів

Рис. 2.2. Приклад із підручника

На закріплення знань учням пропонується для розв'язання 8 задач (рис. 2.3).

Завдання для самостійного виконання

1. Розробіть код обчислення суми для чисел: 2, 7, 21, 9, 33, 13.
2. Розробіть код, обчислення суми непарних чисел, що більші 7, але менші 25.
3. Розробіть код обчислення суми чисел натурального ряду, максимальне значення якого не перевищує 7.
4. Перший член геометричної прогресії 6, а її знаменник — 0.5. Розробіть код обчислення значень членів прогресії, більших 0.6, і визначення номера останнього члена прогресії, що підсумовується.
5. Дано куб, сторони якого набувають п'ять значень: 3, 4.5, 6, 7.5, 9. Розробіть код визначення об'єму для кожного з них.
6. Радіус першої кулі дорівнює 2 см, а радіус кожної наступної збільшується на 0.5 см. Розробіть код для визначення бокових поверхонь перших шести куль.
7. Відомо результати плавання вільним стилем на дистанції 50 м п'яти учнів кожного з трьох 10 класів школи. Розробіть код, за допомогою якого визначається середній час запливу учнями кожного класу.
8. У банк покладено S грн під N відсотків річних. Розробіть код, за допомогою якого визначається кількість років, через які сума вкладу буде не менше N грн.

Рис. 2.3. Завдання на циклічні обчислення

Незважаючи на достатню кількість прикладів завдань на використання циклів, не вистачає типових задач з програмування та прикладів саме їх розв'язання.

Нажаль, підручників для вибіркових модулів рівню стандарт не має, щоб ми могли проаналізувати їх на вивчення програмування.

Отже, вивчення мов програмування відбувається у 10-му класі як за рівнем стандарту, так і профільним рівнем, про що свідчать відповідні навчальні програми з інформатики для учнів 10-11 класів. Але вивчення програмування на рівні стандарту передбачається у розділі «Креативне програмування» і відноситься до вибіркового модуля, тобто не всі учні (школи, класи) обирають даний модуль для вивчення, а тому в окремих школах вивчення мови програмування може взагалі бути відсутнім.

2.2. Методичні особливості навчання програмувати в старшій школі

Процес навчання програмування в ЗЗСО прийнято умовно розбивати на кілька етапів. Перед початком навчання вчитель стикається з проблемою вибору мови програмування для вивчення. Одна група вчителів розглядає тему «Алгоритмізація та програмування» на основі формальних алгоритмів, побудувавши навчання учнів на мові блок-схем. Інша група вчителів інформатики навчає учнів тієї мови програмування, яку сама знає і за допомогою якої сама вміє розв'язувати типові завдання. Таким чином, на початку вивчення програмування вчителі частіше обирають мову програмування з урахуванням власної компетентності і рідше з огляду на інтереси учнів.

Більшість вчителів на початку XXI століття викладали мову Basic. Потім у багатьох школах вивчалася мова Pascal, яка вважалася більш прийнятною з методичної точки зору для вивчення основних принципів програмування. Мова Pascal є навчальною структурною мовою програмування, яка передбачає не тільки вивчення алгоритмічних конструкцій, формування логічного і алгоритмічного мислення в учнів, а й вирішення складних технологічних і виробничих завдань. Наразі уже більшого поширення набувають мови програмування C, C ++, Visual C ++, Delphi, Java, Python та інші.

В якості основних видів організації процесу навчання програмуванню можна виділити традиційні види занять: лекційні, лабораторні та практичні заняття. Для проведення занять необхідно вибрати відповідне навчально-методичне, технічне і програмне забезпечення.

Вивчення програмування в школі проводиться для різних категорій учнів. Головним чином це пов'язано зі спеціалізацією (профільністю) навчання. Одна категорія учнів вивчає програмування у відповідності до обов'язкового змісту, де важливим є вивчення основних алгоритмічних конструкцій з використанням простих типів даних і масивів. Інша категорія учнів вивчає програмування на поглибленому рівні, що відповідає навчанню у профільних класах. Крім базових знань в галузі програмування учні повинні оволодіти однією із сучасних мов програмування і навчитися оперувати основними типами даних та алгоритмізувати обчислювальні процеси.

Вивчення програмування бажано починати з розгляду алгоритмів на формальних мовах і використовувати їх на формальних виконавцях. У старших класах учням необхідно опанувати поняття мови програмування, розглянути класифікацію мов програмування, їх необхідність і спрямованість. Потім розглядаються основи структурного програмування на одній алгоритмічній мові спільно з основними алгоритмічними конструкціями.

Для цього можна обрати мову C++.

Починається вивчення мови програмування зі знайомства з основними компонентами даної мови (константи, ідентифікатори, змінні, типи даних, принципи запису математичних виразів, складові оператори, коментарі), основними стандартними процедурами і функціями, структурою програми. Далі послідовно вивчаються основні алгоритмічні конструкції: лінійна конструкція, розгалуження (умовний алгоритм), оператор множинного вибору, циклічна конструкція (цикли з лічильником, з передумовою та з післяумовою). Спочатку учням дається конструкція на мові блок-схем, а потім реалізація у вигляді програм на обраній мові програмування. Кожна

конструкція закріплюється самостійним розв'язуванням учнями як загальних, так і індивідуальних завдань на практичних заняттях.

Далі у старших класах необхідно перейти до вивчення теми «Процедури і функції», а саме рекомендується розглянути поняття локальних і глобальних змінних, їх відмінності, способи реалізації та використання на основі процедур і функцій. Потім здійснюється перехід на вивчення теми «Масиви», під час практичних занять учні повинні оволодіти вмінням формування масивів та їх опрацювання за допомогою циклічних обчислень або ж функцій користувача. Під час вивчення цієї теми учні закріплюють вміння створювати алгоритмічні конструкції і використовувати їх для розв'язування задач різного рівня складності.

Останнім етапом вивчення програмування у старшій школі є ознайомлення учнів з основами візуального програмування. Саме тут на базовому рівні учні знайомляться з етами створення графічного додатку, з основними компонентами та подіями, налаштуванням різних компонентів програми, побудовою графіків тощо.

Вивчають принципи об'єктно-орієнтованого програмування та застосування їх при створенні власного додатку. Додатково формується поняття файлової змінної, реалізуючи основні методи роботи з даними.

Завдяки такій побудові вивчення мови програмування на базовому рівні учням надається можливість освоїти основи мови програмування, що сприяє розвитку технологічного, алгоритмічного і творчого мислення учнів, сформувати в них елементарні навички програмування.

В результаті вивчення учні повинні знати:

- що таке алгоритм, яка роль алгоритму в системах управління;
- у чому полягають основні властивості алгоритму;
- способи запису алгоритмів через блок-схеми, навчальну алгоритмічну мову;
- основні алгоритмічні конструкції: слідування, розгалуження, множинний вибір, цикл, структури алгоритмів;

- призначення допоміжних алгоритмів, технології побудови складних алгоритмів: метод послідовної деталізації і складальний (бібліотечний) метод;

- основні властивості величин в алгоритмах обробки інформації: що таке ім'я, тип, значення величини; сенс присвоювання;

- призначення мов програмування;

- в чому відмінність між мовами програмування високого рівня і машинно-орієнтованими мовами;

- правила подання даних однією з мов програмування високого рівня (наприклад, на мові С++);

- правила запису основних операторів: введення, виведення, присвоєння, циклу, розгалуження;

- правила запису програми;

- що таке трансляція;

- зміст етапів розробки програми: алгоритмізація, кодування, налагодження і тестування;

- що таке об'єктно-орієнтоване програмування;

- які принципи об'єктно-орієнтованого програмування існують;

- етапи візуального програмування.

При цьому учні повинні вміти:

- користуватися мовою блок-схем, розуміти опис алгоритмів на навчальній алгоритмічній мові;

- виконувати трасування алгоритму для відомого виконавця;

- складати лінійні, розгалужені і циклічні алгоритми;

- виділяти підзадачі, визначати і використовувати допоміжні алгоритми;

- складати програми вирішення обчислювальних завдань з числами;

- програмувати простий діалог;

- працювати в середовищі однієї з систем програмування;

- здійснювати налагодження і тестування програми;

- створювати візуальні додатки та інше.

Після вивчення програмування за описаною методикою учні повною мірою оволодівають навичками програмування.

Вивчення думки вчителів щодо ситуації з небажанням вивчати програмування, зокрема, циклічних обчислень, виявила, що серед основних причин – мала кількість годин на вивчення циклів, відсутність достатньої матеріально-технічної бази традиційні (лекційно-практичні) методи навчання, відсутність достатньої кількості завдань для опанування циклічних обчислень.

З наведених причин лише третя і четверта можуть піддаватися корекції на рівні «вчитель-учень», а тому нами на основі спілкування з учителями та за аналізом науково-методичних джерел визначено аспекти, які потенційно можуть змінити психологічні установки учнів при вивченні програмування в школі.

Мотивація – використання соціальних (розуміння соціальної значущості, потреби, орієнтація на майбутнє) і пізнавальних (інтереси, емоції, ідеали, прагнення) мотивів.

Оскільки мотивація сприяє прагненню до навчання, спонукає діяти з максимальною енергією, то доцільними є сугестивні прийоми (це тип прямого повідомлення, через який людина впливає на рішення, думки та поведінку іншої людини чи групи людей, не звертаючись до раціональної аргументації та не використовуючи фізичного примусу), комунікативні атаки, доведення та переконування. Важливість вивчення програмування доцільно підтримувати «життєвими» аргументами, наприклад, такими: «Візьмемо простий приклад: столові прибори (ложки, виделки, ножі) в закладах громадського харчування, як правило, лежать на початку стійки, але було б набагато зручніше, якби вони знаходилися на її кінці, оскільки у цьому випадку не потрібно буде балансувати з тарілкою на підносі, щоб взяти потрібні прибори. Навички алгоритмічного мислення знаходять своє застосування в самих різних сферах – від планування подорожей до громадської охорони здоров'я. Щоб приймати правильні рішення, націлені на певний результат, завжди корисно виділити головні елементи проблеми і зрозуміти, як вони пов'язані з її частинами. Якщо

ви опанували відповідні методи (алгоритми, підходи), все стає набагато простіше. Тому пропоную вам спробувати мислити комп'ютерними категоріями і впевнитись, що таке мислення логічне, структуроване, послідовне, і, як наслідок, результативне».

Стимулювання і психологічне налаштування. Щоб зняти блок, що гальмує розуміння суті програмування, доцільним є наведення прикладів і статистичних даних про те, скільки людей з нематематичним складом мислення досягають успіхів у програмуванні, який відсоток людей став програмістами вже у зрілому віці. Так, упереджене ставлення дівчат до програмування можна змінювати, зазначаючи, що першим програмістом у світі була жінка – Ада Лавлейс.

Ігрові методи. Підтримка вмотивованості учнів і їх налаштованість на вивчення програмування можливі із застосуванням ігрових методик. Вибір такої стратегії ґрунтується на дослідженнях С. Іщерякова [20]. На думку науковця, новачки починають програмувати з двох причин – навчання і розвага, причому у першому випадку інтерес згасає через нудність процесу і часту плутанину і нерозуміння логіки алгоритмів, а от у другому випадку, де програмування розцінюється як гра, прослідковується зацікавленість і захоплення учнів програмуванням, що і зумовлює подальші успіхи.

Природа гри зрозуміла і приваблива учням своєю яскравістю, азартністю, емоційним піднесенням, можливістю змагатись і перемагати. Вивчення програмування і опанування навичками і стратегіями будь-якої гри мають багато спільного: послідовність кроків для усвідомлення правил і розуміння концепції; невдачі при перших спробах; азарт у прагненні перемоги, який змушує діяти до останнього; ретельний аналіз тактики, яку потрібно опанувати; емоційний підйом від отримання результату (перемоги). Додатковим позитивним ефектом є перехід учня на вищі рівні ієрархії у грі, а отже і під час навчання програмувати. Це додатково спонукає до більш інтенсивної практики програмування та опанування основних його методів.

Ставлення до програмування як до гри зумовлює активне використання рольових ігор при моделюванні ситуацій, змагань та інших інтерактивних методів, що можуть наочно і зрозуміло пояснювати складні елементи програмування, які, як правило викликають найбільші труднощі і втрату інтересу.

Командна робота. Сучасна ІТ-індустрія переважно спрямована на командну діяльність над проектами, тому учні здебільшого навчаються роботі у команді над спільними проектами. Для залучення до командної роботи можна застосовувати практику перехресного обміну кодом із завданням розібратися у чужому коді та доповнити його, написання рецензії на чужий код. Заохочується обмін думками, знаннями, взаємна допомога, спільна генерація ідей. Заняття у команді сприяють соціалізації учнів та розвивають їх комунікативні навички.

Самостійна робота і рефлексія. Самостійна робота бачиться доцільною через можливість мережної взаємодії, зокрема, через використання різних інтерактивних веб-сервісів типу learningapps.org, www.pythontutor.com, www.e-olymp.com, http://cppstudio.com. Одним із варіантів взаємодії з учнями може стати пропозиція вчителя виконувати певні завдання, імпортувати їх єдину мережеву презентацію за допомогою сервісу Google Docs. Перевагою даного сервісу є наявність вбудованого чату, що дає можливість онлайн обговорення результатів.

На практичних заняттях варто приділяти увагу розвитку рефлексивного мислення, яке формується шляхом аналізу виконаних завдань і відображається у відповідях на запитання: «Навіщо це завдання потрібне? Як його застосовувати на практиці? Які труднощі виникали? Чи є альтернативні способи розв'язання?». Також учням пропонується навести приклади, які б продемонстрували практичне втілення ідей розв'язування задачі у практику життя.

Педагогічна практика свідчить, що вивчення програмування часто впливає на вибір майбутньої професійної діяльності, а також відіграє важливу

роль у навчальному процесі, зокрема, сприяє підвищенню інтелектуального розвитку учнів, їхніх інтелектуальних нахилів та аналітичних і творчих здібностей. Забезпечення ефективності навчання учнів програмуванню, активізація відповідної навчально-пізнавальної діяльності та подолання пов'язаних з цим проблем вбачається нами у впровадженні таких підходів, які орієнтуються на психолого-педагогічні чинники – мотивацію, стимулювання та зняття психологічних блоків щодо вивчення програмування, командну роботу, а також технологічні, серед яких виділяємо ігрові технології і виважену й заздалегідь продуману самостійну роботу та рефлексію навчальної діяльності.

2.3. Типові задачі для формування навичок організації циклічних обчислень

Розглянемо типові задачі з програмування, де використовуються циклічні обчислення разом та написано для них програмний код мовою C++.

Задача 1. Складіть програму, яка виводить на екран квадрати чисел від 10 до 20 включно.

Розв'язування даної задачі представлено наступним кодом:

```
int main()                    // оголошення головної функції
{
for (int i=10; i<=20; i++) // початок циклу, де вказано початкове
                              значення, кінцеве значення та крок
        cout<<i*i<<" ";     // тіло циклу (обчислюється квадрат числа)
cout<<endl;                   // перехід на новий рядок
return 0;                     // оголошення про кінець програми
}
```

У цьому випадку задача була написана з використанням циклу з лічильником for. Звісно, її можна переписати використовуючи й інші типи циклів.

З використанням циклу while:

```
int main()
{
int i=10;
while (i<=20) {
        cout<<i*i<<" ";
        i++;
}
cout<<endl;
return 0;
}
```

З використанням циклу do while:

```
int main()
{
int i=10;
do {
        cout<<i*i<<" ";
        i++;
}
while (i<20);
cout<<endl;
return 0;
}
```

Таким чином усі наведені задачі можна переписати з використанням будь-якого виду циклу.

Усі три варіанти циклів даної задачі показано за допомогою блок-схеми на рис. 2.4.

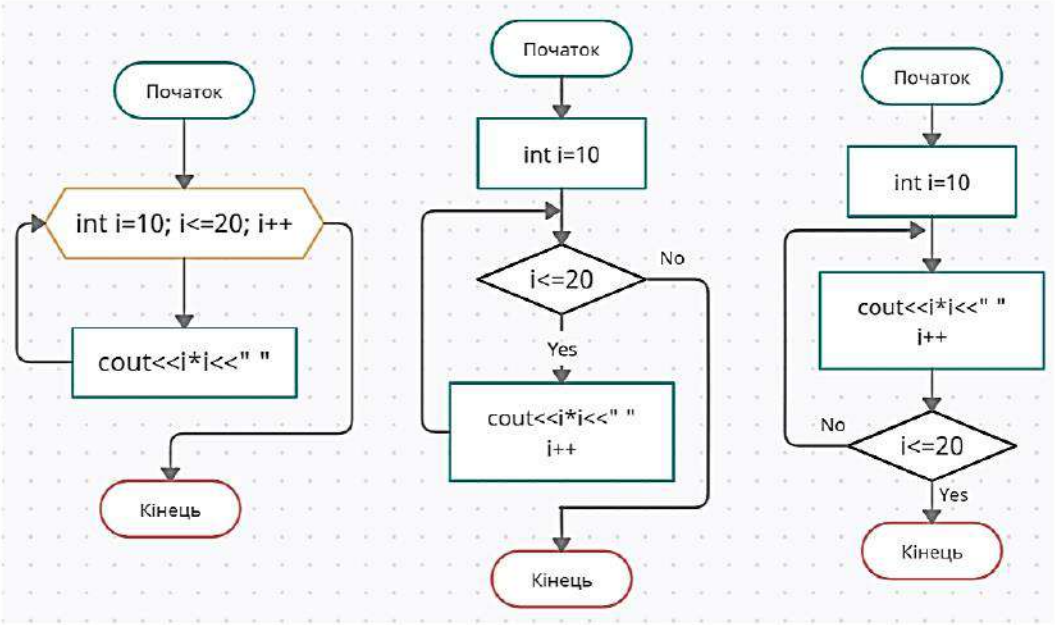


Рис. 2.4. Блок-схеми до задачі 1

Пропонуємо подібні до цієї задачі.

А. Складіть програму, яка виводить на екран куби всіх однозначних чисел

Б. Складіть програму, яка виводить на екран квадрати і куби двозначних чисел.

В. Складіть програму, яка виводить на екран у зворотному порядку квадрати чисел від 20 до 10 включно.

Ґ. Складіть програму, яка виводить на у зворотному порядку числа від 50 до 35 разом із своїми залишками при діленні на 13

Д. Складіть програму, яка виводить на екран квадрати всіх двозначних чисел, які кратні 6.

Задача 2. Дано натуральні числа від 35 до 87. Вивести на консоль ті з них, які при діленні на 7 дають залишок 1, 2 або 5.

Розв'язання даної задачі з використанням циклу for:

```
int main()
{
for (int i=35; i<=87; i++)
```

```
{
        if (i % 7 == 1 || i % 7 == 2 || i % 7 == 5)
                cout<<i<<" ";
}
cout<<endl;
return 0;
}
```

Пропонуємо подібні до цієї задачі

А. Дано натуральні двозначні числа. Вивести на консоль ті з них, які закінчуються на 3 і діляться націло на 6.

Б. Дано натуральні числа від 20 до 40. Вивести на консоль ті з них, які при множенні на 3 закінчуються на 3 або 7.

В. Дано натуральні тризначні числа. Вивести на консоль ті з них, які є парними і діляться націло на 7.

Ґ. Дано натуральні числа від 35 до 87. Вивести на консоль ті з них, які при діленні на 6 і на 3 дають залишок 1 або 2.

Д. Дано двозначні натуральні числа. Вивести на консоль ті з них, які при діленні на 7 дають залишок 1 і закінчуються на цифру 5.

Задача 3. Знайдіть суму 1+2+3+…+*n*, де число *n* вводиться користувачем із клавіатури.

Розв’язання:

```
int main()
{
int n;

cout<<"input n: ";  cin>>n;
if (n<1)
        cout<<"error"<<endl;
else
{
```

```
        int sum=0;
        for (int i=1; i<=n; i++)
                sum+=i;
        cout<<"sum = "<<sum<<endl;
}

return 0;
}
```

Пропонуємо подібні до цієї задачі

*А. Знайдіть суму 1+2+3+…+2*n, де число n вводиться користувачем із клавіатури.*

*Б. Знайдіть суму 2+4+6+…+2*n, де число n вводиться користувачем із клавіатури.*

В. Знайдіть добуток чисел від 1 до n, де число n вводиться користувачем із клавіатури.

Г. Знайдіть суму натуральних чисел від m до n, де числа m і n вводяться користувачем із клавіатури.

Д. Знайдіть добуток однозначних чисел від m до n, де числа m і n вводяться користувачем із клавіатури.

Задача 4. Знайдіть добуток цифр тризначного числа.

Розв'язання:

```
int main()
{
 int n;
 cout<<"input n: ";
 cin>>n;
 if (n<100 || n>999)
  cout<<"error n"<<endl;
 else
 {
  int a,b,c;
  a=n%10;       // перша цифра праворуч
```

```
    b=(n/10)%10;  // друга цифра праворуч
    c=n/100;      // перша цифра числа

    int res;
    res=a*b*c;
    cout<<"answer: "<<res<<endl;
  }
  return 0;
}
```

Пропонуємо подібні до цієї задачі

А. Знайдіть добуток цифр чотиризначного числа.

Б. Знайдіть добуток першої і третьої цифр тризначного числа.

В. Знайдіть суму цифр п'ятизначного числа.

Г. Визначте найбільшу цифру чотиризначного числа.

Д. Визначте найменшу цифру п'ятизначного числа.

Задача 5. Знайдіть кількість парних цифр заданого натурального числа.

Розв'язання:

```
int main()
{
   int n;

   cout<<"input n: ";     cin>>n;
   if (n<=0)
     cout<<"error n"<<endl;
   else
   {
      int count=0;
      while (n>0)
        {
              if ((n%10)%2==0)
                count++;
              n/=10;
        }
```

```
            cout<<"answer: "<<count<<endl;
        }

        return 0;
    }
```

Пропонуємо подібні до цієї задачі

А. Визначте, чи є в записі числа однакові цифри

Б. Знайдіть кількість непарних цифр заданого натурального числа.

В. Знайдіть кількість нулів у записі заданого натурального числа.

Ґ. Виведіть на екран цифри числа, які не повторюються у його записі.

Д. Визначте, чи є однаковими перша і остання цифри числа

Е. Для заданого з клавіатури числа визначити кількість цифр, які кратні 3.

Є. Для заданого натурального числа визначити середнє арифметичне його цифр.

Задача 6. Знайдіть найбільшу цифру введеного із клавіатури натурального числа.

Розв'язання:

```
int main()
{
   int n;

   cout<<"input n: ";     cin>>n;
   if (n<=0)
     cout<<"error n"<<endl;
   else
   {
      int max=0;
      while (n>0)
        {
              if ((n%10)>max)
                max=n%10;
```

```
            n/=10;
        }
        cout<<"answer: "<<max<<endl;
    }

    return 0;
}
```

Пропонуємо подібні до цієї задачі

А. *Знайдіть найменшу цифру введеного із клавіатури натурального числа.*

Б. Визначте, чи є в записі числа парні цифри

В. Знайдіть найменшу й найбільшу цифру введеного із клавіатури натурального числа.

Ґ. Визначте найбільшу цифру чотиризначного числа.

Д. Визначте найменшу цифру п'ятизначного числа.

Е. Для заданого натурального числа визначити середнє гармонічне його цифр.

Задача 7. Знайдіть усі дільники заданого натурального числа.

Розв'язання:

```
int main()
{
int n,t;
cout<<"input n: ";
cin>>n;
t=sqrt(n);
for (int i=1; i<=t; i++)
{
        if (!(n%i))
                if (i!=n/i)
                        cout<<i<<" "<<n/i<<" ";
                else
                        cout<<i<<" ";
```

```
    }
    }
```

Пропонуємо подібні до цієї задачі

А. Знайдіть усі дільники заданого натурального числа і виведіть за спаданням.

Б. Знайдіть усі парні дільники заданого натурального числа і виведіть за спаданням.

В. Знайдіть усі непарні дільники заданого натурального числа і виведіть за зростанням.

Ґ. Знайдіть суму дільників заданого натурального числа.

Д. Знайдіть добуток парних дільників заданого натурального числа.

Задача 8. Розв'язати приклад

$$\cfrac{1}{1+\cfrac{1}{3+\cfrac{1}{5+\cfrac{1}{\ldots}}}}$$
$$101+\frac{1}{103}$$

Розв'язання:

```
 int main()
 {
    int i;
    int n = 103;
    float res;
    i=n;
    res = i;
    while (i>1)
    {
       res = 1 / res;
       i-=2;
       res = i + res;
```

```
    }
    res = 1/res;
     return 0;
 }
```

Пропонуємо подібні до цієї задачі

А. Обчислити значення виразу $\cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{\dots}}}$*, який має 10 вкладень*

Б. Обчислити значення виразу $\cfrac{1}{1+\cfrac{2}{1+\cfrac{3}{\dots}}}$*, який має 5 вкладень*

В. Обчислити значення виразу $\cfrac{1}{2+\cfrac{2}{3+\cfrac{3}{\dots}}}$*, який має 100 вкладень*

Г. Обчислити значення виразу $\cfrac{1}{2+\cfrac{1}{4+\cfrac{1}{\dots}}}$*, який має 10 вкладень*

Д. Обчислити значення виразу $\cfrac{11}{12+\cfrac{2}{13+\cfrac{3}{\dots}}}$*, який має 5 вкладень*

Задача 9. Дано натуральне число n. Обчислити вираз:

$$\sum_{k=1}^{n} \sqrt{k}$$

Розв'язання:

```
 int main()
 {
    int k, n;
    double sum;
    cout<<"input n: ";     cin>>n;
    sum = 0;
    for (k=1; k<=n; k++)
         sum += Math.sqrt(k);
    return 0;
 }
```

Пропонуємо подібні до цієї задачі

А. Обчислити значення суми $\sum_{k=2}^{n=10} \sqrt{k+5}$

Б. Обчислити значення суми $\sum_{k=20}^{n=100} (2+\sqrt{k})$

В. Обчислити значення суми $\sum_{k=20}^{n=30} (k^3)$

Г. Обчислити значення суми $\sum_{k=5}^{n=10} (1/\sqrt{k})$

Д. Обчислити значення суми $\sum_{k=1}^{n=1000} sin\,(\frac{k}{n})$

Задача 10. Знайти найбільше додатне *n*, для якого виконується умова: $3n^2 - 730n < 0$

Розв'язання:

```
int main()
{
   int n;
   int t;
   n=0;
   do
   {
      n++;
      t = 3*n*n - 730*n;
   }
   while (t<0);
   n--;
   return 0;
}
```

Пропонуємо подібні до цієї задачі

А. Знайти найменше додатне n, для якого виконується умова: $3n^2 - 730n > 0$

Б. Для заданого натурального числа визначити суму кубів першої і другої цифр числа.

В. Почавши тренування, лижник пробіг у перший день 10 км, а кожного наступного дня пробігав на 10% більше, ніж у попередній. Визначити, у який

день він уперше пробіжить більше 20 км. У який день сумарний пробіг перевищить 100 км.

Ґ. За даним натуральним числом визначити порядковий номер цифри 3 у ньому. Якщо такої цифри немає, то відповіддю має бути 0. Якщо таких цифр кілька, то відповідь має співпадати з наймолодшим розрядом (номер крайньої правої цифри)

Д. Визначте кількість цифр заданого натурального числа

Розглянуті типові задачі на циклічні обчислення дають можливість зрозуміти принцип організації циклів та навчитися їх організовувати для розв'язання різних задач.

Бачимо ефективними для формування навичок організації циклічних обчислень розв'язувати такі та подібні задачі на уроках інформатики в старших класах.

Висновки до розділу 2

У розділі розглянуто методичні особливості навчання програмуванню та формування в учнів навичок циклічних обчислень на уроках інформатики старшої школи.

За аналізом навчальних програм з інформатики виявлено, що вивчення мов програмування відбувається у 10-му класі як за рівнем стандарту, так і профільним рівнем, але вивчення програмування на рівні стандарту передбачається у розділі «Креативне програмування» і відноситься до вибіркового модуля, тобто не всі учні (школи, класи) обирають даний модуль для вивчення, а тому в окремих школах вивчення мови програмування може взагалі бути відсутнім.

Розглянуто методичні особливості навчання учнів програмуванню та подано приклади розв'язування 10 типових задач на організацію циклічних обчислень мовою C++. Дібрано однотипні завдання для наведених задач для самостійного розв'язування.

ВИСНОВКИ

В роботі висвітлено проблему формування навичок організації циклічних обчислень на уроках інформатики старшої школи. У дослідженні вирішені усі поставлені завдання, що уможливило формулювання таких висновків.

1. Розглянуто популярні мови програмування та описано особливості організації циклічних обчислень в них. Виявлено стан розробленості методичної проблеми навчання учнів організації циклічних обчислень в ЗЗСО. Серед найбільш популярних мов програмування на початок 2021 року відзначені мови Python, Java та C, а для вивчення в школі - Scratch, Object Pascal, C++, Python, Java та Ruby. Кожна з мов передбачає використання циклів різних типів (цикли з лічильником, з передумовою, з післяумовою).

2. Аналіз практичного стану розробленості проблеми виявив тенденцію зниження бажання вивчати програмування із переходом до старших класів. При цьому серед опитаних різняться думки щодо причин зниження такого бажання. Учням старшої школи притаманний низький рівень зацікавленості вивченням програмування та відсутність мотивації. Вивчення думки вчителів щодо ситуації з небажанням вивчати програмування, зокрема, циклічних обчислень, виявила, що серед основних причин – мала кількість годин на вивчення циклів, відсутність достатньої матеріально-технічної бази, традиційні (лекційно-практичні) методи навчання, відсутність достатньої кількості завдань для опанування циклічних обчислень.

3. За аналізом навчальних програм з інформатики виявлено, що вивчення мов програмування відбувається у 10-му класі як за рівнем стандарту, так і профільним рівнем, але вивчення програмування на рівні стандарту передбачається у розділі «Креативне програмування» і відноситься до вибіркового модуля, тобто не всі учні (школи, класи) обирають даний модуль для вивчення, а тому в окремих школах вивчення мови програмування може взагалі бути відсутнім.

4. Розглянуто методичні особливості навчання учнів програмуванню та подано приклади розв’язування 10 типових задач на організацію циклічних

обчислень мовою С++. Дібрано однотипні завдання для наведених задач для самостійного розв’язування.

Робота не вирішує повністю аналізовану проблему. Подальшого дослідження потребують: питання методичного супроводу формування навичок організації циклічних обчислень на уроках інформатики старшої школи в умовах дистанційної освіти, а також в позаурочному форматі навчання.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Atamanyuk S., Semenikhina O., Shyshenko I. Theoretical fundamentals of innovation of higher education in Ukraine. Pedagogy and Education Management Review (PEMR). Tallinn, Estonia, 2021. Issue 2(4). P. 30-36.

2. Dehtiarova N., Petrenko S., Rudenko Yu. Pedagogical design in the context of blended learning for future computer science teachers. Modern approaches to the development of knowledge management. Ljubljana. Slovenia. pp. 313-323.

3. Download Python. URL: https://www.python.org/downloads/

4. Drushlyak M. G., Semenikhina O. V., Kondratiuk S. M., Krivosheya T. M., Vertel A. V., Pavlushchenko N. M. The Automated Control of Students Achievements by Using Paper Clicker Plickers. MIPRO 2020 : Proceedings of 43 International convention on information and communication technology, electronics and microelectronics, 28 вересня – 2 жовтня 2020, Opatija (Croatia). 2020. P. 688-692.

5. Drushlyak M. G., Shishenko I. V., Borozenets N. S., Nekyslykh K. M., Semenikhina O. V. Computer Probabilistic Models Construction and Analysis of Professional Activity of their Use by Ukrainian Mathematics Teachers. Proceedings of 44 International convention on information and communication technology, electronics and microelectronics “MIPRO 2021”, Opatija (Croatia), 28 September – 1 October, 2021. P. 712-717. DOI: 10.23919/MIPRO52101.2021.9596868

6. Drushlyak M., Semenikhina O., Proshkin V., Sapozhnykov S. Training pre-service mathematics teacher to use mnemonic techniques. Journal of Physics: Conference Series. 1840 (2021), 012006. C.1-12 DOI:10.1088/1742-6596/1840/1/012006

7. Java - Энциклопедия языков программирования. URL: http://progopedia.ru/language/java/

8. Kudrina, O., Shpileva, V., Klius, Y., Lavrova, O., Esmanov, O., & Semenikhina, O. Industrial enterprise tax transaction costs planning using digital tools. TEM Journal. 2020. Volume 9(2), P. 619-624. DOI:10.18421/TEM92-26

9. Lazorenko S. A., Semenikhina O. V. Development of Information and Digital Culture of Future Specialists in Physical Culture and Sports as a Modern Problem of Education. Science and Education a New Dimension. Pedagogy and Psychology, VIII (95), Issue 239, 2020 Nov. P. 29-32.

10. Microsoft. Visual Basic 6.0. URL: https://docs.microsoft.com/en-us/previous-versions/visualstudio/visual-basic-6/visual-basic-6.0-documentation

11. Okhrimenko O., Semenikhina O., Shyshenko I. Future teachers' readiness for the digital modernization of inclusive education. New challenges in the development of future specialists: collective monograph. Universitatea Dunarea de Jos Galati, Romania, 2021. P. 83-94.

12. Okhrimenko O., Semenikhina O., Shyshenko I. Readiness of future teachers for digital modernization of inclusive education. Innovative Approaches to Ensuring the Quality of Education, Scientific Research and Technological Processes : collective monograph. 2021. No 3.6.15. P. 694-700.

13. Omelyanenko, V., Kudrina, O., Semenikhina, O., Zihunov, V., Danilova, O. & Liskovetska, T. Conceptual aspects of modern innovation policy. European Journal of Sustainable Development. 2020. Volume 9 (2). P. 238-249. DOI:10.14207/ejsd.2020.v9n2p238

14. Ostroha M., Drushlyak M., Shyshenko I., Naboka O., Proshkin V., Semenikhina O. On the use of social networks in teachers' career guidance activities. Smyrnova-Trybulska E. (ed.). (2021) E-learning in COVID-19 Pandemic Time. "E-learning" Series. Vol. 13 (2021) (Pp. 113-124) Katowice-Cieszyn: Studio Noa for University of Silesia.

15. Petrenko S., Dehtiarova N. Increasing teachers' ict-competency level in the after-graduate education process. Інноваційна педагогіка. Вип. 21. Т. 3. 2020. С. 73-77.

16. Python on Android. URL: https://www.damonkohler.com/2008/12/python-on-android.html

17. Repenning, Alexander "Moving Beyond Syntax: Lessons from 20 Years of Blocks Programing in AgentSheets". Journal of Visual Languages and Sentient Systems. 2017. C. 68-91.

18. Rudenko Yu., Rozumenko A., Kryvosheya T., Karpenko O., Semenikhina O. Online Training during the COVID-19 Pandemic: Analysis of Opinions of Practicing Teachers in Ukraine Proceedings of 44 International convention on information and communication technology, electronics and microelectronics “MIPRO 2021”, Opatija (Croatia), 28 September – 1 October, 2021. DOI: 10.23919/MIPRO52101.2021.9596799

19. Rudenko Yu., Semenikhina O. Analysis of distance learning experience in colleges of Sumy region of Ukraine. Education during a pandemic crisis: problems and prospects / Eds. Tetyana Nestorenko & Tadeusz Pokusa Opole, 2020. P. 175-181

20. Rudenko Yuliia, Olha Naboka, Larysa Korolova, Khana Kozhukhova, Olena Kazakevych, Olena Semenikhina. Online Learning With the Eyes of Teachers and Students in Educational Institutions of Ukraine. TEM Journal. Volume 10, Issue 2, P. 922-931. DOI: 10.18421/TEM102-55.

21. Schildt Herbert. C++ The Complete Reference Third Edition. Osborne McGraw-Hill, 1998. 1008 c.

22. Scratch Statistics - Imagine, Program, Share. URL: https://scratch.mit.edu/statistics/ (дата звернення: 01.11.2019).

23. Semenikhina O. et al. The Formation of Skills to Visualize by the Tools of Computer Visualization. TEM Journal. 2020. Volume 9(4). P. 1704-1710. DOI: 10.18421/TEM94-51

24. Semenikhina O. V. The Using Interactive Methods In The Formation Of Conflictological Culture Of Specialist. International Scientific Journal «Future Science: Youth Innovations Digest». 2019. Volume 3, Issue 3. P. 44-48

25. Semenikhina O., Drushlyak M., Lynnyk S., Kharchenko I., Kyryliuk H., Honcharenko O. On Computer Support of the Course “Fundamentals of Microelectronics” by Specialized Software: the Results of the Pedagogical

Experiment. TEM Journal. 2020. Volume 9 (1). P. 309-316. DOI: 10.18421/TEM91-43

26. Semenikhina O., Drushlyak M., Yurchenko A., Udovychenko O., Budyanskiy D. The use of virtual physics laboratories in professional training: the analysis of the academic achievements dynamics. ICT in Research, Education and Industrial Applications (ICTERI-2020) : 16th International Conference. October, 06-10, 2020. Kharkiv. P. 423-429.

27. Semenikhina O., Proshkin V., Drushlyak M. Mathematical knowledge control automation within dynamic mathematics programs. E-learning and STEM Education / Scientific Editor Eugenia Smyrnova-Trybulska. Katowice–Cieszyn, 2019. P. 571-586. .

28. Semenikhina O., Proshkin V., Naboka O. Application of Computer Mathematical Tools in University Training of Computer Science and Mathematics Pre-service Teachers. International Journal of Research in E-Learning, 2020, 6(2), 1-23. https://doi.org/10.31261/IJREL.2020.6.2.06

29. Semenikhina O., Yurchenko A., Sbruieva A., Kuzminskyi A., Kuchai O., Bida O. The Open Digital Educational Resources In IT-Technologies: Quantity Analysis. Information technologies and learning tools. V. 75. Issue 1. P. 331-348 https://doi.org/10.33407/itlt.v75i1.3114

30. Semenikhina Olena V., Proshkin Volodymyr V. The main problems of using computer mathematical tools in university education. Інформаційні технології в освіті та науці: Збірник наукових праць. Випуск 12. Мелітополь: ФОП Однорог Т.В., 2021. 204 с. С.9-11.

31. Semenikhina, O., Yurchenko, A., Udovychenko, O., Petruk, V., Borozenets, N., Nekyslykh, K. Formation Of Skills To Visualize Of Future Physics Teacher: Results Of The Pedagogical Experiment. Revista Romaneasca Pentru Educatie Multidimensionala, 2021, 13(2), 476-497. https://doi.org/10.18662/rrem/13.2/432

32. Semenog O., Semenikhina O., Oleshko P., Prima R., Varava O., Pykaliuk R. Formation of Media Educational Skills of a Future Teacher in the Professional

Training. Revista Românească pentru Educaţie Multidimensională. 2020. Volume 12. Issue 3, P. 219-245. https://doi.org/10.18662/rrem/12.3/319.

33. Shamonia, V. H., Semenikhina, O. V., Proshkin, V. V., Lebid, O. V., Kharchenko, S. Y., & Lytvyn, O. S. Using the proteus virtual environment to train future IT professionals. CEUR Workshop Proceedings, 2547. P. 24-36.

34. Shishenko I. V., Shamonia V. H., Loboda V. S., Punko V. V., Khvorostina Yu. V. and Voitenko A. A. Studying dynamic mathematics software in the professional training of teachers of computer science, mathematics, and IT specialists. MIPRO 2020 : Proceedings of 43 International convention on information and communication technology, electronics and microelectronics, 28 вересня – 2 жовтня 2020, Opatija (Croatia). 2020. P. 683-687.

35. Stroustrup Bjarne. The C++ Programming Language. Addison-Wesley, 1997. 910c.

36. The Good and the Bad of Java Programming, 2019. URL: https://www.altexsoft.com/blog/engineering/pros-and-cons-of-java-programming/

37. The Making of Python. URL: https://www.artima.com/intv/python.html

38. TIOBE Index. URL: https://www.tiobe.com/tiobe-index/

39. Udovychenko O., Chkana Ya., Yurchenko A., Khvorostina Yu. Introduction of didactic games in the educational process. Фізико-математична освіта. 2019. Вип. 4(22). Частина 2. URL: https://fmo-journal.fizmatsspu.sumy.ua/publ/8-1-0-621.

40. Udovychenko, O. M., Ostroha, M. M., Chernysh, A. E., Kudrina, O. Y., Bondarenko, Y. A., & Kurienkova, A. V. (2020). The use of electronic textbooks in the learning process: A statistical analysis. MIPRO 2020 : Proceedings of 43 International convention on information and communication technology, electronics and microelectronics, 28 вересня – 2 жовтня 2020, Opatija (Croatia). 2020. P. 608-611. doi:10.23919/MIPRO48935.2020.9245146

41. Visual Basic – Вікіпедія. URL: https://uk.wikipedia.org/wiki/Visual_Basic.

42. Voitenko A., Semenikhina O. To the question about inclusive educational space in the training of informatics of children with intellectual disabilities. Education. Innovation. Practice. 2019. Issue 2 (6). P. 6-9.

43. Yurchenko A., Drushlyak M., Sapozhnykov S., Teplytska S., Koroliova L., Semenikhina O. Using online IT-industry courses in the computer sciences specialists' training. International Journal of Computer Science and Network Security. Vol. 21 No. 11 pp. 97-104. http://paper.ijcsns.org/07_book/202111/20211113.pdf

44. Yurchenko A., Semenikhina O., Rudenko Yu., Shamonia V. The Digital Technology in IT-Education: the View of Ukrainian University. Збірник наукових праць Національного університету кораблебудування імені адмірала Макарова, 2020. №4 (482). С. 129-133. https://doi.org/10.15589/znp2020.4(482).15

45. Yurchenko A., Shamonia V., Udovychenko O., Momot R., Semenikhina O. Improvement of Teacher Qualification in the Field of Computer Animation: Training or Master Class? Proceedings of 44 International convention on information and communication technology, electronics and microelectronics "MIPRO 2021", Opatija (Croatia), 28 September – 1 October, 2021. P. 683-687. DOI: 10.23919/MIPRO52101.2021.9596946

46. Yurchenko A.O., Udovychenko O.M., Rozumenko A.M., Chkana Y.O., Ostroha M.M. (2019). Regional Computer Graphics Competition as a Tool of Influence on the Profession Choice: Experience of Sumy Region of Ukraine. 42nd International Convention on Computers in Education (MIPRO) (May 20 – 24, 2019), Opatija, Croatia, 2019, pp. 909-914.

47. Абрамик М.В., Лещук С.О., Олексюк В.П. Використання хмарних технологій у процесі навчання майбутніх учителів інформатики основам програмування. Фізико-математична освіта. 2018. Випуск 4(18). С. 7-11.

48. Алексеев Е.Р., Чеснокова О.В., Кучер Т.В. Самоучитель по программированию на Free Pascal и Lazarus. Донецк.: ДонНТУ, Технопарк ДонНТУ УНИТЕХ, 2009. 503 с.

49. Атаманюк С.І., Шишенко І.В., Семеніхіна О.В. Інновації в освіті та специфічні принципи підготовки майбутніх фахівців їх використовувати. Фізико-математична освіта. Суми, 2020. Вип. 4(26). Ч. 2. С. 13-16.

50. Бобровицька С.Ф., Семеніхіна О.В. Стан розробленості проблеми підготовки майбутніх учителів початкової школи до застосування електронних освітніх ресурсів у професійній діяльності. Педагогіка та психологія. 2019. Вип. 62. С. 23-29.

51. Будянський Д.В., Друшляк М.Г., Семеніхіна О.В., Харченко І.В., Горбачук В.О., Чашечникова О.С. Типологія електронних ресурсів у формуванні риторичної культури фахівця. Інформаційні технології і засоби навчання. 2021. 81(1), С. 82-96. https://doi.org/10.33407/itlt.v81i1.4292

52. Вакал Ю.С., Шамоня В.Г. Організація педагогічного експерименту із використанням сучасних інформаційних технологій: навч. посіб. Суми: СумДПУ імені А. С. Макаренка, 2020. 156 с.

53. Вакалюк Т.А. Програмування мовою Pascal. Навчально-методичний посібник для студентів фізико-математичного факультету. Житомир: ФО-П Левковець Н.М., 2016. 232 с.

54. Ворожбит А.В., Рибак О.С. Огляд курсу за вибором «основи верстки та веб-програмування». Фізико-математична освіта. 2018. Випуск 1(15). С. 20-27.

55. Голиков Д. В. Scratch для юных программистов. СПб.: БХВ-Петербург, 2017. 192 с.: ил.

56. Горошко Ю., Костюченко А., Шкардибарда М. Використання ВПЗ у процесі вивчення основ програмування. Інформатика та інформаційні технології. 2012. №1. С. 22–25.

57. Дегтярьова Н., Мигаль В., Сасіна Ю. Особливості навчання візуальному програмуванню учнів старших класів в середовищі MIT APP INVENTOR.

58. Дегтярьова Н., Петренко С. Актуальні питання формування цифрових компетентностей вчителів різних дисциплін під час підвищення

кваліфікації. Актуальні питання гуманітарних наук: міжвузівський збірник наукових праць молодих вчених Дрогобицького державного педагогічного університету імені Івана Франка. Дрогобич: Видавничий дім «Гельветика», 2020. Вип. 27. Том 2. С. 167-170.

59. Дегтярьова Н.В., Петренко С.І. Змішане навчання як чинник формування навичок самоосвіти у майбутніх вчителів інформатики. Вісник Вінницького політехнічного інституту. 2(143). 2019. С. 117-122.

60. Дегтярьова Н.В., Руденко Ю.О., Вернидуб Г.О. Формування вміння у майбутніх учителів працювати над науковим текстом. Педагогіка формування творчої особистості у вищій і загальноосвітній школах: зб. наук. праць. Запоріжжя: КПУ, 2020. Вип. 68. Т.1. С. 240-243.

61. Дегтярьова Н.В., Руденко Ю.О., Шамоня В. Г., Семеніхіна О.В. Методика вирішення нечітких багатокритеріальних задач вибору варіантів. Збірник наукових праць Національного університету кораблебудування імені адмірала Макарова, 2020. № 3 (481). С. 124-128. https://doi.org/10.15589/znp2020.3(481).16

62. Друшляк М. Г., Юрченко А. О., Розуменко А. М., Розуменко А. О., Семеніхіна О. В. Ефективні форми підвищення кваліфікації вчителів у галузі комп'ютерної анімації. Відкрите освітнє е-середовище сучасного університету, 2021, 10 (1), С. 77-88. https://doi.org/10.28925/2414-0325.2021.108

63. Іщеряков С. Вчити програмування треба в школі чи університеті? Освітня політика. Портал громадських експертів. URL: http://osvita.ua/school/54063/

64. Ковальчук М.Б. Змістові аспекти алгоритмічного мислення. Фізико-математична освіта. 2018. Вип. 3(17). С. 61-66.

65. Мавлютов А.Р., Выдрин Д.Ф., Махнёва А.О. "Самые востребованные языки программирования" Academy, no. 1 (16), 2017, pp. 12-14.

66. Мартиненко О., Чкана Я., Удовиченко О. Управління самостійною роботою майбутніх учителів математики у віртуальному навчальному середовищі через використання електронної версії робочого зошиту.

Педагогічні науки: теорія, історія, інноваційні технології. 2020. № 2 (96). С. 144-153.

67. Навчальна програма з інформатики (профільний рівень) для 10-11 класів загальноосвітніх шкіл, затверджена Наказом Міністерства освіти і науки № 1407 від 23 жовтня 2017 року. URL: https://mon.gov.ua/ua/osvita/zagalna-serednya-osvita/navchalni-programi/navchalni-programi-dlya-10-11-klasiv

68. Навчальна програма з інформатики (рівень стандарту) для 10-11 класів загальноосвітніх шкіл, затверджена Наказом Міністерства освіти і науки № 1407 від 23 жовтня 2017 року. URL: https://mon.gov.ua/ua/osvita/zagalna-serednya-osvita/navchalni-programi/navchalni-programi-dlya-10-11-klasiv

69. Нова українська школа | Веб-ресурс НУШ. URL: https://nus.org.ua/

70. О Ruby. URL: https://www.ruby-lang.org/ru/about/

71. Острога М.М., Шамоня В.Г. Модель формирования готовности будущих бакалавров среднего образования к использованию цифровых технологий в профориентационной деятельности. Science and Education a New Dimension. Pedagogy and Psychology, IX (97), Issue: 246, 2021. P.25-28.

72. Павленко Л.В., Павленко М.П., Хоменко В.Г., Хоменко С.В., Скурська М.М. Інноваційні підходи до вивчення статистики майбутніми ІТ-фахівцями на основі використання мови програмування R. Фізико-математична освіта. 2020. Випуск 1(23). С. 97-105.

73. Петренко С., Петренко Л. Модель формування інформатичної компетентності майбутніх учителів інформатики в процесі фахової підготовки. Педагогічні науки: теорія, історія, інноваційні технології. Суми: СумДПУ імені А. С. Макаренка, 2020. № 2 (96) С. 154-164. DOI 10.24139/2312-5993/2020.02/154-164

74. Петренко С., Петренко Л. Формування готовності майбутніх учителів інформатики до професійної діяльності. Педагогічні науки: теорія,

історія, інноваційні технології. Суми: СумДПУ імені А. С. Макаренка, 2019. № 10 (94). С. 95-105. DOI 10.24139/2312-5993/2019.10/095-106.

75. Петренко С.І. Аналіз проблеми безпечної роботи учнів початкових класів у мережі Інтернет // Петренко С.І. / Вісник університету імені Альфреда Нобеля. Серія «Педагогіка і психологія». Педагогічні науки. 2020. № 1 (19) С. 85-92. DOI: 10.32342/2522-4115-2020-1-19-9

76. Петренко С.І., Дегтярьова Н.В. Формування ІКТ-компетентності викладачів на курсах підвищення кваліфікації. Наукові записки Серія: Педагогічні науки Випуск 186 - Кропивницький: РВВ ЦДПУ ім. В. Винниченка, 2020. с. 150-155.

77. Підручник Pascal - основи програмування. URL: http://pascal.dp.ua/rozdl-pershiy.html (дата звернення: 01.11.2019).

78. Притика О. В., Юрченко А. О. Про особливості мови програмування C+. Інформаційні технології в професійній діяльності : матеріали XIV Всеукраїнської науково-практичної конференції. Рівне : РВВ РДГУ. 2021. С. 155-156.

79. Про затвердження Державного стандарту базової і повної загальної середньої освіти. URL: https://zakon.rada.gov.ua/laws/show/1392-2011-%D0%BF

80. Прошкін В., Хоружа Л., Семеніхіна О. Теорія і практика професійної підготовки майбутніх учителів математики та інформатики засобами цифрових технологій. Теоретичні та практичні аспекти використання математичних методів та інформаційних технологій в освіті й науці: моногр. / за заг. ред. О. Литвин. К.: Київ. ун-т ім. Б. Грінченка, 2021. 332 с. С.48-74.

81. Руденко В. Д. Інформатика (профільний рівень) : підруч. для 10 кл. закл. загал. серед. освіти / В. Д. Руденко, Н. В. Речич, В. О. Потієнко. – Харків : Вид-во «Ранок», 2019. – 256 с.

82. Руденко Ю. О., Дегтярьова Н. В., Юрченко А. О., Семеніхіна О. В. Використання елементів нечіткої логіки у гуманітарних дослідженнях. Збірник наукових праць Національного університету кораблебудування імені

адмірала Макарова, 2020. № 1 (479). С. 130-134. https://doi.org/10.15589/znp2020.1(479).17

83. Руденко Ю.О., Дегтярьова Н.В. Електронні ресурси та сервіси інтернет в контексті реалізації електронного навчання. Професійна підготовка вчителя в умовах цифрового освітнього середовища / за заг. ред. О.В. Семеніхіної. Суми, 2020. С.56-86.

84. Семеніхіна О. В., Прошкін В. В., Друшляк М. Г. Використання прийомів мнемотехніки в процесі навчання математики. Математика в рідній школі. 2020. №5 (219). С. 2-7.

85. Семеніхіна О., Юрченко А. Професійна підготовка фахівця: організація онлайн-опитування для визначення потреб у зміні освітньої програми. Освіта. Інноватика. Практика. 2019. Issue 2(6). P. 36-43.

86. Семеніхіна О., Юрченко А., Удовиченко О. Формування умінь візуалізувати начальний матеріал у майбутніх учителів фізики: результати педагогічного експерименту. Професійна підготовка вчителя в умовах цифрового освітнього середовища / за заг. ред. О.В. Семеніхіної. Суми, 2020. С. 99-117.

87. Семеніхіна О.В., Бобровицька С.Ф. Особливості практичної підготовки вчителів до використання ЕОР у початковій школі. Фізико-математична освіта. 2020. Вип. 1(23). Частина 2. С. 72-77.

88. Семеніхіна О.В., Юрченко А.О., Удовиченко О.М. Формування умінь візуалізувати начальний матеріал у майбутніх учителів фізики: результати педагогічного експерименту. Фізико-математична освіта. 2020. Вип. 1(23). С. 122-128.

89. Семеног О., Семеніхіна О. Медіаосвітні уміння майбутнього вчителя та особливості їх формування у процесі професійної підготовки. Професійна підготовка вчителя в умовах цифрового освітнього середовища / за заг. ред. О.В. Семеніхіної. Суми, 2020. С.118-140.

90. Сучасна інформатика (сила практики в теорії): Структура програми. URL: https://alextexnok.blogspot.com/p/normal-0-21-false-false-false-uk-x-none_29.html

91. Удовиченко О.М. Критерії та показники рівнів готовності майбутніх учителів інформатики до професійної діяльності. Вісник Черкаського національного університету. Серія «Педагогічні науки». Черкаси, 2020. Вип. 2.2020. С. 142-147.

92. Харченко І.І., Удовиченко О.М. Результати експериментального формування культури професійної комунікації майбутніх фахівців з економіки. Вісник Черкаського національного університету. Серія «Педагогічні науки». Черкаси, 2020. Вип. 1.2020. С. 146-150.

93. Хворостіна Ю.В., Удовиченко О.М., Юрченко А.О. Особливості використання дидактичних ігор на уроках математики. Інноваційна педагогіка. 2019. Вип. 19. Том 3. С. 141-146. https://doi.org/10.32843/2663-6085-2019-19-3-29

94. Чередник І.В., Руденко Ю.О., Семеніхіна О.В. Труднощі навчання учнів системам числення і кодуванню інформації та шляхи їх запобігання. Фізико-математична освіта. 2020. Випуск 2(24). Частина 2. С. 21-27.

95. Шамоня В., Семеніхіна О. Комп'ютерна візуалізація роботи логічних елементів інформаційної системи на базі PROTEUS. Професійна підготовка вчителя в умовах цифрового освітнього середовища / за заг. ред. О.В. Семеніхіної. Суми, 2020. С. 87-98.

96. Шамшина Н.В. Методичні аспекти вивчення СУБД ACCESS: створення інформаційних систем. Професійна підготовка вчителя в умовах цифрового освітнього середовища / за заг. ред. О.В. Семеніхіної. Суми, 2020. С. 140-178.

97. Юрченко А. А., Хворостина Ю. В., Острога М. М., Пунько В. В. Изучение программирования: анализ программ по информатике для старшей школы в Украине. The 3rd International scientific and practical conference

“Perspectives of world science and education”: Conference proceedings, (November 27-29, 2019) CPN Publishing Group, Osaka, Japan. 2019. P. 587-595.

98. Юрченко А.О., Самойленко Л.О. Мови програмування, які вивчаються у ЗЗСО. Діджиталізація в Україні: інновації в освіті, науці, бізнесі: Матеріали міжнародної науково-практичної конференції, 16-18 вересня 2019 року, Бердянськ, 2019. С. 38-47.

99. Юрченко А.О., Семеніхіна О.В., Хворостіна Ю.В., Удовиченко О.М. Навчання програмувати в старшій школі крізь призму чинних навчальних програм. Фізико-математична освіта. 2019. Випуск 2(20). Ч.2. С. 47-54.

100. Юрченко А.О., Семеніхіна О.В., Хворостіна Ю.В., Удовиченко О.М., Петренко С.І. Навчання програмувати в старшій школі крізь призму чинних навчальних програм. Фізико-математична освіта. 2019. Вип. 2(20). Ч. 2. С. 48-55. DOI 10.31110/2413-1571-2019-022-4-021.

101. Юрченко А.О., Удовиченко О.М., Хворостіна Ю.В., Петренко С.І. Дослідження рівня знань майбутніх учителів фізики при використанні цифрових лабораторій. Фізико-математична освіта. 2019. Вип. 4(22). С. 137-141. DOI 10.31110/2413-1571-2019-022-4-021.

102. Язык программирования Ruby. URL: https://www.internet-technologies.ru/articles/yazyk-programmirovaniya-ruby.html

103. Язык программирования Паскаль. URL: http://informatics-lesson.ru/pascal/index.php

ДОДАТКИ

Додаток А

TIOBE Index for November 2021

November Headline: PHP about to lose its top 10 position

Since the start of the TIOBE index, more than 20 years ago, PHP has been a permanent top 10 player. Recently, we saw PHP struggling to stay in that top 10. PHP was once the master of web programming, but now it is facing a lot of competition in this field. This is not to say that PHP is dead. There are still a lot of small and medium enterprises relying on PHP. So I expect PHP to decline further but in a very slow pace. Two of PHP's competitors, Ruby and Groovy, gain both 3 positions this month. Ruby from #16 to #13 and Groovy from #15 to #12. Other interesting moves this month are Lua (from #32 to #26), Dart (from #40 to #31), and Kotlin (from #38 to #33). -- *Paul Jansen CEO TIOBE Software*

The TIOBE Programming Community index is an indicator of the popularity of programming languages. The index is updated once a month. The ratings are based on the number of skilled engineers world-wide, courses and third party vendors. Popular search engines such as Google, Bing, Yahoo!, Wikipedia, Amazon, YouTube and Baidu are used to calculate the ratings. It is important to note that the TIOBE index is not about the *best* programming language or the language in which *most lines of code* have been written.

The index can be used to check whether your programming skills are still up to date or to make a strategic decision about what programming language should be adopted when starting to build a new software system. The definition of the TIOBE index can be found here.

Nov 2021	Nov 2020	Change	Programming Language	Ratings	Change
1	2	↑	Python	11.77%	-0.35%
2	1	↓	C	10.72%	-5.49%
3	3		Java	10.72%	-0.96%
4	4		C++	8.28%	+0.69%
5	5		C#	6.06%	+1.39%
6	6		Visual Basic	5.72%	+1.72%

Nov 2021	Nov 2020	Change	Programming Language	Ratings	Change
7	7		JavaScript	2.66%	+0.63%
8	16	⇈	Assembly language	2.52%	+1.35%
9	10	↑	SQL	2.11%	+0.58%
10	8	↓	PHP	1.81%	+0.02%
11	21	⇈	Classic Visual Basic	1.56%	+0.83%
12	11	↓	Groovy	1.51%	-0.00%
13	15	↑	Ruby	1.43%	+0.22%
14	14		Swift	1.43%	+0.08%
15	9	⇊	R	1.28%	-0.36%
16	12	⇊	Perl	1.22%	-0.29%
17	18	↑	Delphi/Object Pascal	1.22%	+0.36%
18	13	⇊	Go	1.21%	-0.16%
19	34	⇈	Fortran	1.19%	+0.79%
20	**17**	↓	**MATLAB**	**1.17%**	**+0.07%**

Other programming languages

The complete top 50 of programming languages is listed below. This overview is published unofficially, because it could be the case that we missed a language. If you have the impression there is a programming language lacking, please notify us at tpci@tiobe.com. Please also check the overview of all programming languages that we monitor.

Position	Programming Language	Ratings
21	(Visual) FoxPro	1.06%
22	SAS	0.98%
23	Prolog	0.77%
24	Scratch	0.77%
25	COBOL	0.68%
26	Lua	0.58%
27	PL/SQL	0.58%
28	Objective-C	0.55%
29	Rust	0.54%
30	Lisp	0.49%
31	Dart	0.42%
32	Ada	0.41%
33	Kotlin	0.40%
34	D	0.40%
35	Scala	0.36%
36	Julia	0.35%
37	ABAP	0.34%
38	PowerShell	0.28%
39	Clojure	0.24%
40	Haskell	0.24%
41	Ladder Logic	0.24%
42	VBScript	0.24%
43	VHDL	0.23%
44	LabVIEW	0.23%
45	Scheme	0.23%

Position	Programming Language	Ratings
46	TypeScript	0.22%
47	Apex	0.18%
48	Transact-SQL	0.18%
49	Logo	0.16%
50	Erlang	0.15%

The Next 50 Programming Languages

The following list of languages denotes #51 to #100. Since the differences are relatively small, the programming languages are only listed (in alphabetical order).

- ABC, Algol, APL, Awk, B4X, Ballerina, Bash, BCPL, Bourne shell, CL (OS/400), Clipper, Dylan, Eiffel, Elixir, Emacs Lisp, F#, Forth, Haxe, Icon, Inform, Io, J#, Korn shell, Lingo, LiveCode, M4, Maple, ML, MQL4, NXT-G, Oberon, OCaml, OpenCL, Pure Data, Q, Racket, Raku, RPG, S, sed, Simulink, Solidity, SPARK, SPSS, Stata, Tcl, Vala/Genie, Verilog, Xojo, Zig