

**Укладачі:**

Стець Андрій Михайлович – викладач Вінницького технічного коледжу, керівник гуртків інформатики Вінницького обласного Центру технічної творчості учнівської молоді.

Стець Ірина Іванівна – викладач ДПТНЗ «Вінницьке вище професійно-технічне училище», керівник гуртків інформатики Вінницького обласного Центру технічної творчості учнівської молоді.

**Рецензенти:**

Косовець О.П., кандидат педагогічних наук, керівник гуртка-методист;

Свята З.І., завідуюча відділом Вінницького обласного Центру технічної творчості учнівської молоді.

.

**ПОЯСНЮВАЛЬНА ЗАПИСКА**

З розвитком інформаційних технологій і систем зростає й потреба у спеціалістах з розробки програмного забезпечення.

Сучасні підходи до створення програмних продуктів шляхом використання різного роду фреймворків, що визначають загальний каркас застосування і надають можливість використовувати готові рішення для реалізації багатьох важливих і складних функцій програми, у значній мірі спрощують процес створення веб-застосувань, програм для мобільних пристроїв тощо.

Поєднання вивчення основ програмування з використанням фреймворків дозволяє швидко отримувати проект прикладної програми, який забезпечує виконання потрібних функцій, а потім експериментувати на його основі, вивчаючи програмні засоби та архітектурні рішення, що у ньому використовуються. Але таке поєднання має свої особливості: для роботи з проектом потрібно бути ознайомленим із засобами та способами структуризації програмного коду, архітектурними та структурними рішеннями, які використовуються у проекті.

Характерною особливістю структури даної навчальної програми є те, що вона складається з двох паралельних змістовних ліній: основ алгоритмізації та програмування і розробки програмного забезпечення. Обидві ці лінії тематично взаємопов’язані і послідовно узгоджені. Тематика практичних завдань обох ліній націлена на вирішення наскрізної задачі з реалізації програм накопичення, редагування, збереження і обробки даних. Це дозволяє підготувати вихованців до сприйняття нового матеріалу взаємопов’язаних частин обох паралельних змістовних ліній і не втрачати в часі актуальність вивчення тем обох курсів.

Особливий акцент у змісті програми зроблено на вивченні засобів структуризації програми і використанні ефективних структурних рішень, які покращують читабельність програми, забезпечують можливість повторного використання коду, спрощують модернізацію програми і полегшують вивчення і освоєння популярних фреймворків, що призначені для реалізації каркасів програм різного типу.

Пропонована програма побудована на основі особистісно-орієнтованого, діяльнісного та компетентнісного підходів.

В основу навчальної програми покладені авторські матеріали, які були розроблені для забезпечення викладання дисциплін «Алгоритми і методи обчислень», «Програмування», «Основи програмної інженерії», «Сучасні мережеві технології» для підготовки молодших спеціалістів зі спеціальності 5.05010201 «Обслуговування комп’ютерних систем і мереж», а також матеріали, які були розроблені для забезпечення занять гуртка «Інформатика та обчислювальна техніка» за програмою курсу «Основи програмування на C#».

У визначенні загальної структури програми використані ідеї, що закладені у навчальній програмі з основ візуального програмування, рекомендованій Міністерством освіти і науки України [10].

Дана навчальна програма реалізується у гуртку науково-технічного напряму позашкільних та загальноосвітніх навчальних закладів і спрямована на вихованців віком від 12 років.

***Мета і завдання курсу***

Метою навчальної програми є формування ключових компетентностей особистості в процесі вивчення основ програмування та реалізації проектів програм.

Основні завдання полягають у формуванні таких компетентностей:

* пізнавальної – забезпечує ознайомлення з поняттями та знаннями, що стосуються призначення та особливостей використання основних засобів мов програмування, основ проектування програм та використання каркасів програм;
* практичної – сприяє оволодінню навичками і способами діяльності:
  + виконання декомпозиції задачі – розбивати рішення задачі на кілька функцій, створювати функції для багаторазового використання; використовувати відповідні структури даних і алгоритми; створювати загальний / об'єктно-орієнтований код, який інкапсулює ті умови задачі, які можуть бути змінені;
  + виконання декомпозиції системи – виконувати декомпозицію задачі і проектувати систему в межах однієї чи декількох платформ або технологій;
  + організації коду у файлі – групувати методи логічно і за викликами, розділювати код на регіони, описувати хороші коментарі, здійснювати несуперечливу розстановку пробілів і табуляції, естетично оформляти код;
  + організації коду між файлами – схожі файли групувати в папки, використовувати кожен фізичний файл для опису однієї структури даних чи реалізації одного функціоналу, організовувати код на фізичному рівні у точній відповідності до структури проекту;
  + організації дерев файлів вихідного коду – від простого поділу коду в логічні підкаталоги до повної відповідності дерева вихідного коду відповідає логічній ієрархії та організації коду в проекті;
  + забезпечення читабельності коду – використовувати хороші імена файлів, змінних, класів, методів і т. д.; описувати функції короткого розміру, нестандартний код і допущення в коді пояснювати коментарями, не використовувати глибокої вкладеності умов або методів;
  + реалізації безпечного програмування – перевіряти всі аргументи, значення, що повертаються, обробляти виключення в коді, у якому потенційно можуть бути помилки, писати модульні тести, які емулюють збої;
  + здійснення обробки помилок – обробляти помилки в коді, який або кидає виняток, або генерує помилку; слідкувати за тим, щоб після виникнення помилки програма продовжувала працювати, а непотрібні більш ресурси були коректно звільнені оброблювачем помилки; писати код так, щоб визначати можливі помилки на ранньому етапі;
  + ефективного використання середовища програмування (IDE) – використовувати IDE для редагування тексту, ефективно користуватися меню в IDE, знати деякі тонкощі середовища, гарячі клавіші функцій, що найчастіше використовуються;
  + ефективного використання програмного інтерфейсу системи чи платформи (API) – швидко знаходити потрібні засоби і запам’ятовувати ті з них, що найчастіше використовуються; створювати бібліотеки, які обертають API, для спрощення завдань, що найчастіше зустрічаються;
  + використання фреймворків – використовувати популярні фреймворки (каркаси програм);
  + реалізації вимог до системи – розуміти виставлені вимоги і писати код відповідно до технічного завдання;
  + ефективного використання засобів мови програмування – ефективного використовувати засоби імперативного, об'єктно-орієнтованого та декларативного програмування, види типізації, що підтримуються мовою, ледачі обчислення;
  + використання структур даних – використовувати на практиці масиви, зв'язані списки, словники та інші складні структури даних; аргументовано вибирати для реалізації конкретної задачі ті чи інші структури даних на основі розуміння їх переваг та недоліків;
  + використання стандартних алгоритмів – використовувати методи сортування, обходу та пошуку в структурах даних;
  + використання засобів автоматизації тестування – створювати модульні тести для написаного коду;
  + використання технічних засобів та новітніх інформаційних технологій – застосовувати їх при виконанні поставлених завдань та оформленні звітів (у текстовій формі та формі таблиць, презентацій, тощо.); використовувати різноманітне програмне забезпечення, як офісне, так і спеціалізоване;
* творчої – забезпечує формування творчих здібностей вихованців у процесі оволодіння навичками і способами діяльності:
  + здійснення пошукової та дослідницької діяльності з залученням різних типів джерел та матеріалів, не залежно від рівня складності завдання;
  + поєднання знань з різних дисциплін, зразком чого є використання знань з суміжних дисциплін для реалізації програмних засобів і аналіз предметних областей для обробки даних яких створюється програмне забезпечення;
  + формулювання вимог до системи – задавати питання, що стосуються не розглянутих у завданні випадків; розуміти картину в цілому і пропонувати кращі альтернативи та додаткові вимоги;
* соціальної – сприяє вихованню самодисципліни та самовідповідальності, самореалізації особистості в соціумі, професійному самовизначенню, розвитку навичок і способів діяльності:
  + формулювання і обґрунтовування власної думки – спілкуватись на теми реалізації програм, розуміти інших і зрозуміло пояснювати свою думку, мати хороші правопис і граматику;
  + організації командної роботи – співпрацювати з іншими людьми, проявляти толерантність, не втрачаючи при цьому своєї особистості та власного погляду на речі;
  + правильної організації часу – дотримуватись встановлених термінів виконання роботи.

На опрацювання навчального матеріалу відводиться 216 годин (основний рівень – 6 год. на тиждень).

Програма курсу розрахована на проведення занять за умови постійного доступу вихованців до комп'ютерів з відповідним програмним забезпеченням.

Програму побудовано шляхом поєднання лінійно-концентричного та блочного принципів з дотриманням дидактичних принципів доступності, послідовності, системності, доцільності, достатності, наступності у навчанні від простого до складного, зокрема подача нового матеріалу базується на уже відомому та в тісному зв'язку з ним, допускається заміна одного із змістовних блоків програми в залежності від запитів вихованців.

Програма передбачає:

* обрання, за бажанням керівника гуртка мови програмування (C#, Java, Python тощо) та середовища розробки. Також програма враховує бажання вихованців – якщо вони бажають, то наступний рік вони можуть навчатися за цією ж самою програмою, але вивчаючи іншу мову програмування;
* ділення на окремі модулі (розділи) таким чином, щоб:
  + навчання у першому модулі було спрямовано на знайомство з основними засобами вибраної мови програмуванням та принципами реалізації консольного і віконного інтерфейсів користувача;
  + навчання у другому модулі було спрямовано на ознайомлення вихованців з видами динамічних та файлових структур даних, перевагами і недоліками їх використання та з основними засобами їх реалізації, які надає вибрана мова програмування;
  + навчання у третьому було спрямовано на вивчення основ роботи з довільним фреймворком, що визначає каркас програми. Тип проекту, який реалізує фреймворк також може бути довільним: веб-застосування, віконне застосування, програма для планшетів і мобільних пристроїв тощо. Авторський варіант програми визначає орієнтовну структуру модуля у випадку використання фрейморка, що призначений для створення веб-застосування. Здібні і обдаровані вихованці можуть з дозволу керівника гуртка обрати для вивчення інший фреймворк, ніж той, над вивченням якого працює група, за умови забезпечення курсу відповідним навчальним матеріалом;
  + навчання у четвертому модулі спрямоване на практичне використання вивчених засобів програмування і технологій розробки програм: вихованці розробляють власний проект програми, а наприкінці приймаюсь участь у конкурсі кращих робіт;
* впровадження дистанційної форми у навчальний процес, що дозволить вирішити питання опрацювання матеріалів курсу за індивідуальними варіантами програми, забезпечення неперервного навчання під час карантину, надання можливості навчання інвалідам та вихованцям, які з поважних причин не можуть відвідувати заклади освіти.

Навчальний матеріал програми адаптований до занять з вихованцями різного рівня підготовленості, але передбачається наявність базових знань користувача персонального комп’ютера.

Програма є орієнтовною. За необхідності керівник гуртка може внести до програми певні зміни на свій розсуд, які не повинні впливати на загальний зміст навчальної програми та кількість навчальних годин. Незмінними мають залишатися мета, завдання і прогнозований результат освітньої діяльності.

Загальними принципами організації навчально-виховного процесу є: науковість, синтез інтелектуальної і практичної діяльності, індивідуальний підхід, послідовність і поступовість викладення матеріалу.

Найбільш вдалою формою організації навчального процесу в гуртку є лекційно-практична система, в якій виділяються лекції та практикум з розв’язування задач. Причому більше уваги приділяється організації різних видів самостійної роботи вихованців. Керівник допомагає вибрати і визначити постановку задачі, контролює і направляє розробку прикладної програми, бере участь у її тестуванні.

**Навчально-методичне забезпечення програми**

Для ***навчально-методичного забезпечення програми*** окрім відповідних підручників і навчальних посібників необхідні такі програмні та додаткові технічні засоби:

1. комп'ютерний клас;
2. операційна система: Microsoft Windows, Linux або Mac OS;
3. середовище розробки програм, що підтримує вибрану мову програмування (C#, Java, Python тощо): MS Visual Studio, Mono, Net Beans, Idea, WingWare, Eclipse або інше;
4. мережа Інтернет;
5. мультимедійний проектор для демонстрації на екран або локальна мережа для демонстрації занять і прикладів на комп'ютери вихованців (бажано).

## Основний рівень, перший рік навчання

## НАВЧАЛЬНО-ТЕМАТИЧНИЙ ПЛАН

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№ теми** | **Розділ, тема** | Кількість годин | | |
| теоретичних | практичних | усього |
| 1. | Вступне заняття | 2 | 1 | 3 |
| *2.* | *Організація програм* | *31* | *53* | *84* |
| 2.1 | Основні засоби мови | 14 | 19 | 33 |
| 2.2. | Структуровані типи | 4 | 8 | 12 |
| 2.3. | Основи об'єктно-орієнтованого програмування | 6 | 12 | 18 |
| 2.4. | Основи створення віконних програм | 4 | 8 | 12 |
| 2.5. | Програмування графіки | 3 | 6 | 9 |
| *3.* | *Організація даних* | *9* | *15* | *24* |
| 3.1. | Динамічні структури даних | 6 | 9 | 15 |
| 3.2. | Файлові структури даних | 3 | 6 | 9 |
| *4.* | *Використання каркасів програм* | *12* | *21* | *33* |
| 5. | Реалізація індивідуальних проектів | 14 | 55 | 69 |
| 5.1. | Планування проекту | 4 | 8 | 12 |
| 5.2. | Розробка і реалізація програми | 11 | 34 | 45 |
| 5.3. | Представлення та захист проектів | 2 | 10 | 12 |
| 6. | Підсумокове заняття | 3 |  | 3 |
|  | **Разом** | **74** | **142** | **216** |

# ЗМІСТ ПРОГРАМИ

#### 1. Вступне заняття (3 год )

*Теоретична частина*. Мета, завдання та зміст роботи гуртка. Організаційні питання. Правила поведінки в навчальному закладі та кабінеті інформатики. Ознайомлення з програмою навчання та технологіями, що вивчаються.

*Практична частина.* Вибір предметної області (ПО) для виконання індивідуальних завдань.

## 2. Організація програм (84 год.)

**2.1. Основні засоби мови (30 год.)**

*Теоретична частина*. Інформаційні моделі. Алгоритми. Властивості алгоритмів. Способи представлення алгоритмів. Базові алгоритмічні структури. Консольні застосування. Засоби структурування коду. Форматування виведення. Елементи мови програмування: типи даних, змінні, константи, літерали. Перетворення типів. Підпрограми. Функції. Методи. Передача параметрів. Функції що повертають значення. Виконання обчислень. Операції. Вирази. Засоби для виконання математичних обчислень. Перевантаження функцій. Оператори розгалуження. Умови. Оператор умови. Умовна операція. Винятки та їх обробка. Оператор вибору. Перерахування. Оператори циклу. Цикли з параметром. Вкладені цикли. Символи. Обробка натискання клавіш. Обробка тексту. Обробка рядків. Регулярні вирази. Введення і виведення тексту.

*Практична частина.* Створення інформаційної моделі.Реалізація консольного виведення. Реалізація консольного введення даних. Реалізація підпрограм для введення даних. Реалізація підпрограм для виконання обчислень. Реалізація алгоритмів з розгалуженнями. Реалізація алгоритмів вибору варіантів. Вдосконалення підпрограм введення даних. Реалізація алгоритмів з повтореннями. Реалізація підпрограм обробки символів. Робота з текстом.

**2.2. Структуровані типи (18 год.)**

*Теоретична частина*. Масиви. Види масивів. Одновимірні масиви. Багатовимірні масиви. Типи користувача. Об'єкти. Поля. Екземплярні дані та методи. Доступ до елементів поточного об'єкта. Ініціалізація об'єктів. Посилання. Узагальнене програмування. Засоби узагальненого програмування. Обмеження на параметри-типи. Опосередкований виклик методів. Лямбда-вирази.

*Практична частина.* Робота з одновимірними масивами. Робота з двовимірними масивами. Робота з типами користувача. Використання засобів узагальненого програмування.

**2.3. Основи об'єктно-орієнтованого програмування (15 год.)**

*Теоретична частина*. Об'єктно-орієнтоване програмування. Принципи ООП. Інкапсуляція. Успадкування. Поліморфізм. Інтерфейси. Інтерфейсні посилання. Успадкування інтерфейсів. Масиви об'єктів. Порівняння об'єктів. Опосередкований виклик методів. Поняття шаблонів проектування. Тестування програмного забезпечення.

*Практична частина.* Реалізація ієрархії класів. Реалізація інтерфейсів. Робота з масивами об'єктів. Використання опосередкованих викликів методів. Використання шаблонів проектування. Реалізація модульних тестів.

**2.4. Основи створення віконних програм (12 год.)**

*Теоретична частина*. Графічний інтерфейс користувача. Компоненти для реалізації графічного інтерфейсу. Властивості компонентів. Розміщення компонентів вікна. Події та методи компонентів. Організація взаємодії вікон. Редагування даних.

*Практична частина.*  Реалізація віконної програми. Використання подій та методів компонентів. Робота з декількома вікнами. Реалізація вікон редагування даних.

**1.5. Програмування графіки (9 год.)**

*Теоретична частина*. Засоби малювання. Малювання графічних примітивів. Малювання тексту. Операції із зображеннями.

*Практична частина.* Реалізація малювання графічних примітивів. Реалізація малювання тексту. Реалізація операцій із зображеннями.

## 3. Організація даних (24 год.)

**3.1. Динамічні структури даних (15 год.)**

*Теоретична частина*. Базові структури даних. Зв'язані списки, дерева, хеш-таблиці. Ефективність виконання операцій. Колекції. Типи колекцій. Засоби обробки даних. Стеки та черги. Колекції з доступом за ключем.

*Практична частина.*  Аналіз реалізації базових структур даних. Робота з колекціями. Реалізація обробки даних. Використання стеків та черг. Використання колекцій з доступом за ключем.

**3.2. Файлові структури даних (9 год.)**

*Теоретична частина*. Файлове введення-виведення. Читання та запис файлів. Формати передачі даних. Серіалізація. Види серіалізації.

*Практична частина.* Реалізація файлового введення-виведення. Використання форматів передачі даних. Використання серіалізації.

## 4. Використання каркасів програм (33 год.)

*Теоретична частина*. Поняття каркасу програми. Каркаси для реалізації веб-застосувань. Архітектурний шаблон MVC. Структура проекту веб-застосування. Моделі. Засоби генерації розмітки HTML. Маршрутизація. Навігація за даними. AJAX. Редагування даних. Валідація моделі. Авторизація та аутентифікація. Розгортання застосувань. Способи розгортання застосувань.

*Практична частина.* Створення веб-застосування. Робота з даними. Робота з засобами генерації розмітки HTML. Робота з шаблонами URL. Реалізація навігації. Використання AJAX. Реалізація засобів редагування. Реалізація засобів валідації. Реалізація засобів авторизації та аутентифікації.

## 5. Реалізація індивідуальних проектів (72 год.)

**5.1. Планування проекту (15 год.)**

*Теоретична частина*. Визначення теми проекту. Інтернет та авторське право. Проектування інтерфейсу користувача.

*Практична частина.* Перегляд готових проектів. Пошук інформації. Створення інформаційної моделі. Створення макету інтерфейсу користувача.

**5.2. Розробка програми (42 год.)**

*Теоретична частина*. Процес створення програмного забезпечення. Прототипування програмних систем. Розробка через тестування. Ієрархія класів предметної області. Організація даних ПО. Організація сховища даних. Організація взаємодії елементів інтерфейсу користувача. ілісність даних. Підтримка цілісності даних. Фільтрація даних та отримання підсумкової інформації. Техніки тест дизайну. План розробки тест кейсів.

*Практична частина.*  Реалізація класів для представлення сутностей ПО. Реалізація класів наборів даних. Реалізація засобів введення-виведення. Реалізація інтерфейсу користувача. Реалізація засобів редагування даних. Реалізація засобів обробки даних. Розробка тест кейсів. Тестування програми.

**5.3. Представлення та захист проектів (12 год.)**

*Теоретична частина*. Документація програмного забезпечення. Порядок захисту проектів.

*Практична частина.* Підготовка документації до програми. Підготовка демонстраційних матеріалів. Захист проектів.

**6.Підсумкове заняття (3 год.)**

# Прогнозований результат

*Вихованці мають знати і розуміти:*

* базові поняття програмування: концепцію типу, операції, оператори, принципи та правила їх застосування;
* засоби керування порядком обчислень;
* принципи організації та застосування складених структур даних;
* основи програмування графіки;
* суть процедурного підходу до проектування програм;
* суть об'єктно-орієнтованого підходу до проектування програм;
* основи методології розробки програм;
* принципи управління пам’яттю;
* принципи організації та застосування динамічних структур даних;
* принципи та методи роботи з файловими структурами даних;
* призначення та структуру автоматизованих тестів;

*Вихованці мають вміти і застосовувати:*

* складати алгоритми, використовуючи базові структури;
* реалізовувати алгоритми структурними програмами;
* ефективно вибирати типи та структури даних для зберігання інформації;
* структурувати програму, використовуючи сучасні методології розробки;
* використовувати динамічні структури даних для роботи з інформацією;
* застосовувати операції роботи з файлами;
* проектувати прикладні програми, що призначені для зберігання та редагування інформації;
* створювати модульні тести;

*У вихованців мають сформуватися компетентності:*

* виконання декомпозиції задачі;
* виконання декомпозиції системи;
* організації коду у файлі;
* організації коду між файлами;
* організації дерев файлів вихідного коду;
* забезпечення читабельності коду;
* реалізації безпечного програмування;
* здійснення обробки помилок;
* ефективного використання середовища програмування;
* ефективного використання програмного інтерфейсу системи чи платформи;
* реалізації вимог до системи;
* ефективного використання засобів мови програмування;
* використання структур даних;
* використання стандартних алгоритмів;
* використання засобів автоматизації тестування;
* використання технічних засобів та новітніх інформаційних технологій;
* здійснення пошукової та дослідницької діяльності;
* поєднання знань з різних дисциплін;
* формулювання вимог до системи;
* формулювання і обґрунтовування власної думки;
* організації командної роботи;
* правильної організації часу.

# ЛІТЕРАТУРА

1. [ПРИКЛАДНЕ ПРОГРАМУВАННЯ (автори Міловідов Ю.О., Бєлоглазова Н.В.) «Рекомендовано Міністерством освіти і науки України» (лист Міністерства освіти і науки, молоді та спорту України від 15.09.2014 №1/11-14728)](https://drive.google.com/file/d/0B3m2TqBM0APKTHN1bWtKbWZzYWs/view?usp=sharing).
2. Албахари Дж., Албахари Б. С# 6.0. Справочник. Полное описание языка, 6-е изд. Пер. с англ. Москва: ООО “И.Д. Вильямс”, 2016. 1040 с.
3. Бріггс Дж. Р. PYTHON для дітей. Веселий вступ до програмування. Пер. з англ. Львів: Видавництво Старого Лева, 2018. 400 с.
4. Васильев А. Н. Python на примерах. Практический курс по программироанию. *Просто о сложном*. Санкт-Петербург: Наука и Техника, 2016. 432 с.
5. Гамма Э., Хелм Р., Джонсон Р. и др. Приемы объектно-ориентированного проектирования. Паттерны проектирования. Пер. с англ. Санкт-Петербург: Питер, 2011. 796 с.: ил.
6. Гослинг Дж. и др. Язык программирования Java SE 85-е издание. Пер. с англ. Москва: ООО “И.Д.Вильямс”, 2015. 672 c.
7. Доусон М. Программируем на Python. Пер. с англ. Санкт-Петербург: Питер, 2014. 418 с.
8. Дрейер М. C# для школьников: Учебное пособие. Москва: Интернет-Университет Информационных Технологий; БИНОМ. Лаборатория знаний, 2009. 128 с.
9. Мартин Р. Чистый код.: Создание, анализ и рефакторинг. *Библиотека программиста*. Пер. с англ. Санкт-Петербург: Питер, 2010. 796 c.
10. Мэтиз Э. Изучаем Python. Программирование игр, визуализация данных, веб-приложения. *Библиотека программиста*. Пер. с англ. Санкт-Петербург: Питер, 2017. 406 с.
11. Навчальна програма з основ візуального програмування. *Навчальні програми з позашкільної освіти науково-технічного напряму* / за ред. Биковського Т.В., Шкури Г. А. Київ: УДЦПО, 2014. Вип. 2. С. 152-158.
12. Свейгарт Э. Автоматизация рутинных задач с помощью Python: практическое руководство для начинающих. Пер. с англ. Москва: ООО “И.Д.Вильямс”, 2017. 592 с.
13. Сэнд У. Сэнд К. Hello World! Занимательное программирование. *Вы и ваш ребенок.* Пер. с англ. Санкт-Петербург: Питер, 2016. 400 с.
14. Руденко В. Д., Жугастров О. О. Вивчаємо Java у школі: навч. посіб. у 2 ч.: Ч. 1. Синтаксис мови. Харків: Вид-во “Ранок”, 2016. 96 с.
15. Руденко В. Д., Жугастров О. О. Вивчаємо Java у школі: навч. посіб. у 2 ч.: Ч. 2. Класи, об’єкти, методи Харків: Вид-во “Ранок”, 2016. 96 с.
16. Симан М. Внедрение зависимостей в .NET. Пер. с англ. Санкт-Петербург: Питер, 2014. 464 с.
17. Тиддвелл Дж. Разработка пользовательских интерфейсов. Пер. с англ. 2-e изд. Санкт-Петербург: Питер, 2011. 480 с.
18. Троелсен, Э. Язык программирования C# 5.0 и платформа .NET 4.5, 6-е изд. Пер. с англ. Москва: ООО “И.Д.Вильямс”, 2013. 1312 с.
19. Эккель Б. Философия Java 4-е издание. Пер. с англ. Санкт-Петербург: Питер, 2009. 640 с.
20. Интерактивный учебник языка Python. URL: http://pythontutor.ru/ (дата звернення: 08.08.2018).
21. Матрица компетентности программиста ч. I. URL: https://docs.google.com/ document/d/1FVvoSY35YD4BfAkv-XYGRITFbE17pA7A-R6S76UVsBQ/ pub (дата звернення: 08.08.2018).
22. Матрица компетентности программиста ч. II. URL: https://docs.google.com/ document/d/1FVvoSY35YD4BfAkv-XYGRITFbE17pA7A-R6S76UVsBQ/pub (дата звернення: 08.08.2018).
23. Освоюємо Java. *Вікіпідручник*. URL: https://uk.wikibooks.org/wiki/ \_Java (дата звернення: 08.08.2018).
24. Пориньте у Python 3. *Вікі-підручник.* URL: https://uk.wikibooks.org/ wiki/Пориньте\_у\_Python\_3 (дата звернення: 08.08.2018).
25. Сайт о программировании. URL: https://metanit.com/ (дата звернення: 08.08.2018).