ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД УКООПСПІЛКИ
«ПОЛТАВСЬКИЙ УНІВЕРСИТЕТ ЕКОНОМІКИ І ТОРГІВЛІ»

ІНСТИТУТ ЕКОНОМІКИ, УПРАВЛІННЯ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
ФОРМА НАВЧАННЯ ДЕННА
КАФЕДРА МАТЕМАТИЧНОГО МОДЕЛЮВАННЯ ТА СОЦІАЛЬНОЇ ІНФОРМАТИКИ

Допускається до захисту

Завідувач кафедри _____О.О. Ємець
(підпис)

«_____»_____2020 р.

ПОЯСНЮВАЛЬНА ЗАПИСКА ДО ДИПЛОМНОЇ РОБОТИ

на тему

РОЗРОБКА ТРЕНАЖЕРУ ДЛЯ ВИВЧЕННЯ ОСНОВ МОВИ ПРОГРАМУВАННЯ PYTHON

зі спеціальності «Комп'ютерні науки та інформаційні технології»

Виконавець роботи Мельницький Ян Віталійович
_____ «___»_____2020р.
(підпис)
Науковий керівник к.ф.-м.н., доц. Олексійчук Юрій Федорович
_____ «___»_____2020р.
(підпис)

ПОЛТАВА 2020р.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ, ТЕРМІНІВ

Умовні позначення, символи, скорочення, терміни	Пояснення умовних позначень, скорочень, символів
тренажер	комп’ютерна програма, призначена для вивчення і закріплення різноманітних практичних навичок
Python	мова програмування високого рівня, що має додатки в багатьох сферах, включаючи веб-програмування, сценарії, наукові обчислення та штучний інтелект
JavaFX	платформа та набір інструментів для створення насичених інтернет-застосунків з можливістю підвантаження медіа та змісту
FXML	це мова розмітки користувальницького інтерфейсу на основі XML

ВСТУП

Стрімка глобалізація особливо явно проявляється не тільки в світовій економіці, а і в глобальній інформатизації суспільства. У свою чергу, інформатизація суспільства ініціює формування інформаційно-комунікативної сфери, роблячи доступною інформацію будь-якого виду для кожної людини. Забезпечується така можливість засобами інформаційних технологій, завдяки яким людина здатна накопичувати, зберігати інформацію, працювати з будь-якою інформацією, застосовувати її в професійній діяльності.

З розвитком інформаційних технологій зростає їх роль та використання у сфері освіти. Світовим трендом у сфері освіти стають відкриті онлайн-курси і медіа-освіта. Онлайн-курси стали сьогодні дуже популярним засобом навчання, Така форма навчання дає змогу інтерактивного спілкування студентів та викладачів, а також прийому іспитів в режимі онлайн. Це одна із найновіших форм дистанційного навчання, яка активно розвивається у світовій освіті.

Важливим аспектом дистанційної освіти є використання спеціальних тренажерів та інтерактивних завдань для практичних дисциплін, що містять завдання розрахункового характеру. Сюди належать, зокрема, завдання математичних дисциплін.

Метою роботи є програмна реалізація тренажеру для вивчення основ мови програмування Python.

Об’єктом розробки в даній роботі є процес дистанційного навчання математичним дисциплінам.

Предмет розробки – безпосередньо програмний продукт, який реалізує тренажер для вивчення основ мови програмування Python на мові програмування JavaFX.

Головне завдання – розробити алгоритм роботи тренажеру та створити його програмну реалізацію.

Методи розробки – програмні засоби мови JavaFX.

При реалізації тренажеру використано мову програмування Java та середовище розробки NetBeans.

Робота складається з чотирьох розділів. У першому розділі розглянуто постановку основних вимог до змісту і постановку задачі для реалізації тренажеру. У другому розділі описано основні переваги використання тренажерів у дистанційному навчанні та складові дистанційної освіти. У третьому розділі розглянуто основні поняття мови Python, описано алгоритм роботи тренажеру та її блок-схема. У четвертому розділі – описано обґрунтування вибору програмних засобів, процес програмної реалізації, програму та інструкцію по використанню тренажеру.

Обсяг пояснювальної записки: 83 стор., в т.ч. основна частина - 52 стор., джерела - 11 назв.

1. ПОСТАНОВКА ЗАДАЧІ

1.1. Постановка основних вимог до змісту

Зміст роботи повинен відповідати завданню затвердженим керівником та завідувачем кафедри.

Розглянемо основні положення викладення матеріалу [1]:

- чіткість та логічна послідовність викладення матеріалу;
- переконливість аргументації;
- стислість і точність формулювань, які виключають можливість неоднозначного тлумачення; коректність викладення результатів дослідження;
- обґрунтованість рекомендацій та пропозицій.

У роботі повинні бути відображеними [2]:

- актуальність тематики та відповідність до сучасного стану науки, техніки і питань виробництва;
- обґрунтування вибраного напрямку досліджень, методів розв’язку задачі та їх порівнянь оцінки;
- аналіз та узагальнення існуючих результатів;
- розробка загальної методики проведення досліджень;
- характер і зміст виконаних теоретичних досліджень та розрахунків, методи досліджень;
- обґрунтування необхідності проведення експериментальних досліджень, принцип дії розроблених програм, характеристики цих програм, оцінка похибок розрахунків, отримані експериментальні дані; теоретичний (практичний) аналіз розроблених алгоритмів (програм);
- блок-схема програми (алгоритму);
- контрольні приклади;

- оцінка повноти розв’язку поставленої задачі;
- наукова або практична цінність виконаної роботи, виклад наукової новизни, якщо вона є.

1.2. Постановка задачі для реалізації тренажеру

Розглянемо основні завдання роботи:

- підготовити матеріал, що буде використовуватися при реалізації тренажеру;
- розробити алгоритм тренажеру для вивчення основ мови програмування Python;
- скласти блок-схему алгоритму роботи тренажеру;
- розробити навчальний тренажер з даної теми.

Оскільки, тренажер розробляється з метою вивчення основ мови програмування Python, то пропонується наступний перелік тем для ознайомлення:

- Що таке Python?
- Ваша перша програма;
- Прості операції;
- Числа з плаваючою точкою;
- Інші числові операції;
- Рядки;
- Прості функції введення і виведення;
- Строкові операції;
- Перетворення типу;
- Змінні;
- Операції «на місті»;
- Використання редактора.

Розроблений тренажер повинен мати зрозумілий для користувача інтерфейс, тому слід зберегтис таку структуру:

- стартова сторінка:
  - тема тренажеру;
  - перехід до тренування;
  - інформація про розробника
- сторінка з вибором теми:
  - перелік тем;
  - перехід до виконання обраної теми;
- сторінка з матеріалом:
  - теоретичний матеріал;
  - приклад для засвоєння матеріалу;
  - повернутися на попередній крок;
  - перейти на наступний крок.

2. ІНФОРМАЦІЙНИЙ ОГЛЯД

2.1. Основні переваги використання тренажерів у дистанційному навчанні

Сучасні телекомунікаційні, інформаційні і комп’ютерні технології, в першу чергу – електронні тренажери широко використовуються для навчання.

Програми-тренажери використовуються для тренування навичок з розв’язання типових задач.

Вони забезпечують:

- послідовне виведення на екран завдань заданої складності з вибраної теми;
- контроль за діями користувача з розв’язання запропонованого завдання;
- миттєву реакцію на неправильні дії;
- виправлення помилок користувача;
- демонстрацію правильного розв’язання завдання;
- виведення підсумкового повідомлення про результати роботи користувача (можливо, з рекомендаціями чи порадами).

За допомогою тренажерів можлива реалізація індивідуального підходу в організації навчання. Так, до переваг даної форми роботи відносимо наступне:

- генерація навчальних задач та прикладів;
- можливість покрокового контролю виконання завдання;
- можливість проведення роботи з тренажером у поза-навчальний час;
- формування завдань різного рівня складності;

- відсутність контролю зі сторони викладача (контроль надається інформаційній системі), що значно підвищує ефективність роботи самого викладача.

2.2. Складові дистанційної освіти

Дистанційна освіта є однією з найперспективніших форм освіти на теперішній день. Являючись формою заочної форми освіти, дистанційна освіта значно ефективніша, оскільки передбачає значно більше очне спілкування між викладачем та студентом, навіть при тому, якщо вони розділені територіальне. Це можливо стало за допомогою сучасних телекомунікаційних технологій та Інтернет. Саме це дозволяє значно підвищити ефективність навчання і залучити значно більшу кількість студентів для навчання.

Будь-яке навчання потребує певної організаційно-інформаційної підтримки. Складовими дистанційної освіти є:

- Навчальний центр, який здійснює необхідні функції організаційної підтримки;
- Інформаційні ресурси - учбові курси, довідкові, методичні та інші матеріали;
- Засоби забезпечення технології дистанційного навчання (організаційні, технічні, програмні та ін.);
- Викладачі-консультанти, які курують дистанційні курси і називаються тьюторами;
- Студенти.

Звичайно навчальний центр містить у собі й викладачів з додатковим персоналом, і інформаційні ресурси, і окремо організаційно-адміністративну систему (свого роду "електронний деканат"). Також для простоти можна припустити, що він містить у собі технічні й програмні

засоби, хоча, по суті, вони стосуються всіх аспектів системи дистанційної освіти.

Для організації й правильного функціонування системи дистанційної освіти необхідно виконувати наступні основні функції:

- підтримка навчальних курсів;
- доставка навчального матеріалу студентам;
- підтримка довідкових матеріалів (бібліотека);
- консультації;
- контроль знань;
- організація спілкування студентів (колективні форми навчання).

Елементи дистанційного навчального курсу:

- інформаційні ресурси;
- засоби спілкування;
- система тестування;
- система адміністрування.

Найважливішим компонентом дистанційного курсу є інформаційні ресурси, тому що в них зосереджена змістовна частина - контент (content).

Контент включає:

- навчальний матеріал (конспекти лекцій, демонстраційні матеріали і т.д.);
- додаткові інформаційні матеріали (коментарі викладача, відповіді на питання, що задають часто, і т.п.);
- бібліотеку ресурсів (рекомендована література, списки Web-ресурсів по темі курсу, тощо);
- предметний й/або тематичний словник (глосарій);
- програму навчання (академічний календар).

Всі ці компоненти можуть бути реалізовані різними засобами. Це і застосування потужних мов програмування, і застосування спеціалізованих

авторських систем, тощо. Також дистанційна освіта може використовуватися в денній формі освіти. Наприклад, для студентів, що пропустили заняття або не можуть бути присутніми на заняттях з певної причини, тощо [3].

3. ТЕОРЕТИЧНА ЧАСТИНА

3.1. Основні поняття мови Python

Що таке Python?

Python - мова програмування високого рівня, що має додатки в багатьох сферах, включаючи веб-програмування, сценарії, наукові обчислення та штучний інтелект.

Він дуже популярний і використовується такими організаціями, як Google, NASA, CIA та Disney.

Три основні версії Python - це 1.x, 2.x та 3.x. Вони підрозділяються на незначні версії, такі як 2.7 та 3.3.

Код, написаний для Python 3.x, гарантовано працює у всіх майбутніх версіях.

На даний момент використовуються Python версії 2.x та 3.x.

Цей курс охоплює Python 3.x, але його важко змінити від однієї версії до іншої.

У Python є кілька різних реалізацій, написаних різними мовами.

Версія, що використовується в цьому курсі, CPython, є найбільш популярною на сьогоднішній день.

Ваша перша програма

Почнемо із створення короткої програми, що відображає «Привіт, світ!».

У Python ми використовуємо оператор print для виведення тексту:

```
>>> print('Hello world!')
Hello world!
```

print може також використовуватися для виведення декількох рядків тексту.

Наприклад:

```
>>> print('Hello world!')
Hello world!
>>> print('Hello world!')
Hello world!
>>> print('Spam and eggs...')
Spam and eggs...
```

Прості операції

Python має можливість проводити обчислення.

Введіть обчислення безпосередньо в консоль Python, і він видасть відповідь.

```
>>> 2 + 2
4
>>> 5 + 4 - 3
6
```

Python також здійснює множення та ділення, використовуючи зірочку для позначення множення, а пряму косу рису вказує на ділення.

Використовуйте дужки, щоб визначити, які операції виконуються першими.

```
>>> 2 * (3 + 4)
14
>>> 10 / 2
5.0
```

Знак мінус вказує на від’ємне число.

Операції виконуються на від’ємних числах так само, як і на додатних.

```
>>> -7
-7
>>> (-7 + 2) * (-4)
20
```

Поділ на нуль у Python створює помилку, оскільки відповідь не може бути обчислена.

```
>>> 11 / 0
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
ZeroDivisionError: division by zero
```

Числа з плаваючою точкою

Числа з плаваючою точкою використовуються в Python для представлення чисел, які не є цілими числами.

Деякі приклади чисел, які представлені у вигляді чисел з плаваючою точкою, становлять 0,5 та -7,8237591.

Їх можна створити безпосередньо, ввівши число з десятковою точкою, або за допомогою таких операцій, як поділ на цілі числа. Додаткові нулі в кінці числа ігноруються.

```
>>> 3/4
0.75
>>> 9.8765000
9.8765
```

Як ви бачили раніше, ділення будь-яких двох цілих чисел дає число з плаваючою точкою.

Число з плаваючою точкою також виробляється шляхом виконання операції на двох числах з плаваючою точкою, або на числі з плаваючою точкою і на цілому.

```
>>> 8 / 2
4.0
>>> 6 * 7.0
42.0
>>> 4 + 1.65
5.65
```

Інші числові операції

Крім додавання, віднімання, множення та ділення, Python також підтримує експоненцію, яка є підняттям одного числа до сили іншого. Ця операція виконується за допомогою двох зірочок.

```
>>> 2**5
32
>>> 9 ** (1/2)
3.0
```

Щоб визначити коефіцієнт та решту поділу, використовуйте відповідно ділення підлоги та модуля.

Поділ підлоги робиться за допомогою двох передніх косих.

Оператор модуля виконується символом відсотка (%).

Ці оператори можна використовувати як з плавцями, так і з цілими числами.

Цей код показує, що 6 тричі переходить у 20, а решта, коли 1,25 ділиться на 0,5, - 0,25.

```
>>> 20 // 6
3
>>> 1.25 % 0.5
0.25
```

Рядки

Якщо ви хочете використовувати текст у Python, ви повинні використовувати рядок.

Рядок створюється введенням тексту між двома одинарними або подвійними лапками.

Коли на консолі Python відображається рядок, вона зазвичай використовує одиничні лапки. Розмежувач, який використовується для рядка, ніяк не впливає на те, як він поводиться.

```
>>> "Python is fun!"
'Python is fun!'
>>> 'Always look on the bright side of life'
'Always look on the bright side of life'
```

Деякі символи не можуть бути безпосередньо включені в рядок. Наприклад, подвійні лапки не можуть бути безпосередньо включені в подвійний рядок цитат; це призведе до того, що вона закінчиться передчасно.

Таких символів необхідно уникнути, поставивши перед ними зворотний нахил.

Інші поширені символи, які потрібно уникнути, - це нові рядки та зворотні риски.

Подвійні лапки потрібно уникати лише у подвійних рядках цитат, і те саме стосується рядків з цитатами.

```
>>> 'Brian\'s mother: He\'s not the Messiah. He\'s a very naughty boy!'
'Brian's mother: He's not the Messiah. He's a very naughty boy!'
```

Python пропонує простий спосіб уникнути написання "\ n" вручну, щоб уникнути нових рядків у рядку. Створіть рядок із трьома наборами лапок, і нові рядки, створені натисканням клавіші Enter, автоматично переходять для вас.

>>> """Customer: Good morning.

Owner: Good morning, Sir. Welcome to the National Cheese Emporium."""

'Customer: Good morning.\nOwner: Good morning, Sir. Welcome to the National Cheese Emporium.'

Прості функції введення і виведення.

Зазвичай програми беруть введення і обробляють його, щоб отримати виведення.

У Python ви можете використовувати функцію print для отримання виводу. Тут відображається текстове зображення на екрані.

>>> print(1 + 1)

2

>>> print("Hello\nWorld!")

Hello

World!

Щоб отримати вхід від користувача в Python, ви можете використовувати інтуїтивно названу функцію введення.

Функція пропонує користувачеві ввести і повертає те, що вони вводяться як рядок (із вмістом автоматично виходить).

>>> input("Enter something please: ")

Enter something please: This is what\nthe user enters!

'This is what\\nthe user enters!'

Строкові операції

Як і у цілих чисел і чисел з плаваючою точкою, рядки в Python можна додавати, використовуючи процес, званий конкатенацією, який можна зробити на будь-яких двох рядках.

При об'єднанні рядків не має значення, чи створені вони з одинарними або подвійними лапки.

```
>>> "Spam" + 'eggs'
'Spameggs'
>>> print("First string" + ", " + "second string")
First string, second string
```

Навіть якщо ваші рядки містять числа, вони все одно додаються як рядки, а не цілі числа. Додавання рядка до числа створює помилку, оскільки, хоч вони і можуть виглядати схоже, це дві різні сутності.

```
>>> "2" + "2"
'22'
>>> 1 + '2' + 3 + '4'
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
TypeError: unsupported operand type(s) for +: 'int' and 'str'
```

Рядки також можна помножити на цілі числа. Це створює повторну версію початкового рядка. Порядок рядка та цілого числа не має значення, але рядок зазвичай стає першим.

Рядки не можна помножити на інші рядки. Рядки також не можна помножити на числа з плаваючою точкою, навіть якщо числа з плаваючою точкою - цілі числа.

```
>>> print("spam" * 3)
spamspamspam
>>> 4 * '2'
'2222'
>>> '17' * '87'
TypeError: can't multiply sequence by non-int of type 'str'
>>> 'pythonisfun' * 7.0
```

TypeError: can't multiply sequence by non-int of type 'float'

Перетворення типу

У Python неможливо виконати певні операції через задіяні типи. Наприклад, ви не можете додати два рядки, що містять числа 2 і 3 разом, щоб створити ціле число 5, оскільки операція буде виконуватися над рядками, що робить результат "23".

Рішенням цього є перетворення типів.

У цьому прикладі ви б використали функцію int.

```
>>> "2" + "3"
'23'
>>> int("2") + int("3")
5
```

Іншим прикладом перетворення типів є перетворення користувачем даних (що є рядком) до чисел (цілих чисел або чисел з плаваючою точкою), щоб забезпечити виконання обчислень.

```
>>> float(input("Enter a number: ")) + float(input("Enter another number: "))
Enter a number: 40
Enter another number: 2
42.0
```

Змінні

Змінні відіграють дуже важливу роль у більшості мов програмування, і Python не є винятком. Змінна дозволяє зберігати значення шляхом присвоєння йому імені, яке можна використовувати для посилання на значення пізніше в програмі.

Щоб призначити змінну, використовуйте один знак рівності. На відміну від більшості рядків коду, які ми розглянули до цього часу, він не дає жодного результату на консолі Python.

```
>>> x = 7
>>> print(x)
7
>>> print(x + 3)
10
>>> print(x)
7
```

Змінні можна перепризначати стільки разів, скільки вам потрібно, щоб змінити їх значення.

У Python змінні не мають конкретних типів, тому ви можете призначити рядок змінній, а пізніше призначити цілу цілу величину.

```
>>> x = 123.456
>>> print(x)
123.456
>>> x = "This is a string"
>>> print(x + "!")
This is a string!
```

Певні обмеження застосовуються щодо символів, які можуть використовуватися в іменах змінних Python. Дозволені лише символи - це літери, цифри та підкреслення. Крім того, вони не можуть починати з цифр.

Недотримання цих правил призводить до помилок.

```
>>> this_is_a_normal_name = 7
>>> 123abc = 7
SyntaxError: invalid syntax
```

```
>>> spaces are not allowed
SyntaxError: invalid syntax
```

Спробу посилання на змінну, яку ви не призначили, викликає помилку.

Ви можете використовувати оператор del для видалення змінної, що означає, що посилання з імені на значення видаляється, а спроба використання змінної викликає помилку. Видалені змінні можна переназначити пізніше як звичайні.

```
>>> foo = "a string"
>>> foo
'a string'
>>> bar
NameError: name 'bar' is not defined
>>> del foo
>>> foo
NameError: name 'foo' is not defined
```

Ви також можете взяти значення змінної з даних користувача.

```
>>> foo = input("Enter a number: ")
Enter a number: 7
>>> print(foo)
7
```

Операції «на місті»

Місцеві оператори дозволяють писати код типу "x = x + 3" більш стисло, як "x + = 3".

Те ж саме можливо і з іншими операторами, такими як -, *, / і%.

```
>>> x = 2
>>> print(x)
```

```
2
>>> x += 3
>>> print(x)
5
```

Ці оператори можуть використовуватися і для інших типів, крім чисел, таких як рядки.

```
>>> x = "spam"
>>> print(x)
spam
>>> x += "eggs"
>>> print(x)
Spameggs
```

Використання редактора

Поки ми використовували лише Python з консоллю, вводячи та запускаючи по одному рядку коду за один раз.

Фактичні програми створюються по-різному; багато рядків коду записуються у файл, а потім виконуються за допомогою інтерпретатора Python.

У IDLE це можна зробити, створивши новий файл, ввівши якийсь код, збереживши файл та запустивши його. Це можна зробити або за допомогою меню, або за допомогою комбінацій клавіш Ctrl-N, Ctrl-S і F5.

Кожен рядок коду у файлі інтерпретується так, ніби ви ввели його по черзі на консолі [4].

```
x = 7
x = x + 2
print(x)
```

Результат:

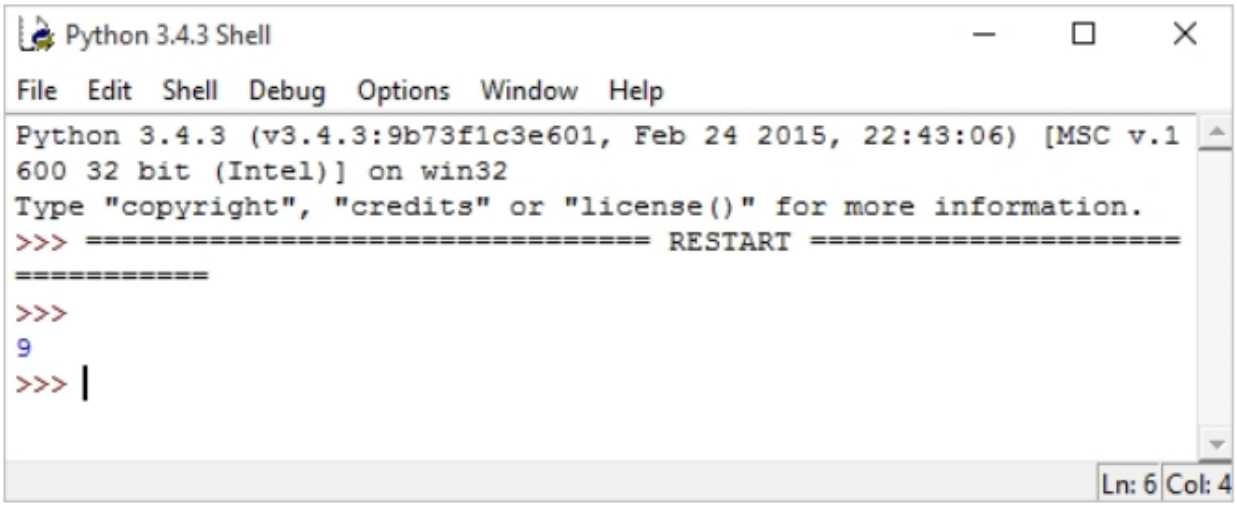

Рисунок 3.1 – Результат виконання коду

3.2. Алгоритм роботи тренажеру

Крок 0. Пропонується вибрати одну з перелічених тем для подальшого ознайомлення з нею:

- Що таке Python?
- Ваша перша програма;
- Прості операції;
- Числа з плаваючою точкою;
- Інші числові операції;
- Рядки;
- Прості функції введення і виведення;
- Строкові операції;
- Перетворення типу;
- Змінні;
- Операції «на місті»;
- Використання редактора.

Виводиться спочатку теоретичне вікно з поясненням відносно обраної теми, а потім тестове. В ньому перевіряється засвоєння матеріалу у вигляді завдання з вибором одного з варіантів відповідей або заповнення пустої комірки.

Що таке Python?

Крок 1. Виводиться теорія: «Python - мова програмування високого рівня, що має додатки в багатьох сферах, включаючи веб-програмування, сценарії, наукові обчислення та штучний інтелект.

Він дуже популярний і використовується такими організаціями, як Google, NASA, CIA та Disney.». Перехід на наступний крок.

Крок 2. Виводиться завдання: «Python - це:» і варіанти відповіді:

- Мова програмування;
- Набір інструментів редагування;
- Середовище розвитку.

Якщо вибрано перший варіант, то перехід на наступний крок. Інакше – виводиться повідомлення про помилку.

Крок 3. Виводиться теорія: «Три основні версії Python - це 1.x, 2.x та 3.x. Вони підрозділяються на незначні версії, такі як 2.7 та 3.3.

Код, написаний для Python 3.x, гарантовано працює у всіх майбутніх версіях. На даний момент використовуються Python версії 2.x та 3.x.

У Python є кілька різних реалізацій, написаних різними мовами.

Версія, що використовується в цьому курсі, CPython, є найбільш популярною на сьогоднішній день.». Перехід на наступний крок.

Крок 4. Виводиться завдання: «Яке з цих тверджень є правдивим?» і варіанти відповіді:

- CPython - це реалізація Python;
- Python 1.7 - це найбільш широко використовувана версія;
- Код Python повинен бути завжди компільований.

Якщо вибрано перший варіант, то перехід на крок 0. Інакше – виводиться повідомлення про помилку.

Ваша перша програма

Крок 1. Виводиться теорія: «Почнемо із створення короткої програми, що відображає «Привіт, світ!».

У Python ми використовуємо оператор print для виведення тексту:» і приклад. Перехід на наступний крок.

Крок 2. Виводиться завдання: «Заповніть пробіли, щоб надрукувати "Hi"», слід заповнити комірку:

>>>_____("Hi")

Якщо вказано «print», то перехід на наступний крок. Інакше – виводиться повідомлення про помилку.

Крок 3. Виводиться теорія: «print може також використовуватися для виведення декількох рядків тексту.» і приклад. Перехід на наступний крок.

Крок 4. Виводиться завдання: «Заповніть пробіл для виведення "ni ni ni".», слід заповнити комірки:

>>> print ('ni ni ni'___

Якщо вказано «)», то перехід на крок 0. Інакше – виводиться повідомлення про помилку.

Прості операції

Крок 1. Виводиться теорія: «Python має можливість проводити обчислення. Введіть обчислення безпосередньо в консоль Python, і він видасть відповідь.» і приклад. Перехід на наступний крок.

Крок 2. Виводиться завдання: «Що виводить цей код?», слід заповнити комірку:

>>> 1 + 2 + 3

____

Якщо вказано «6», то перехід на наступний крок. Інакше – виводиться повідомлення про помилку.

Крок 3. Виводиться теорія: «Python також здійснює множення та ділення, використовуючи зірочку для позначення множення, а пряму косу рису вказує на ділення.

Використовуйте дужки, щоб визначити, які операції виконуються першими.» і приклад. Перехід на наступний крок.

Крок 4. Виводиться завдання: «Який варіант виводиться цим кодом?

>>> (4 + 8) / 2» і варіанти відповіді:

- 6
- 8
- 6.0

Якщо вибрано третій варіант, то перехід на наступний крок. Інакше – виводиться повідомлення про помилку.

Крок 5. Виводиться теорія: «Знак мінус вказує на від'ємне число.

Операції виконуються на від'ємних числах так само, як і на додатних.» і приклад. Перехід на наступний крок.

Крок 6. Виводиться завдання: «Заповніть комірку, щоб зробити цей код правильним.», слід заповнити комірку:

>>> (____5 - 1) * 3

-18

Якщо вказано «-», то перехід на наступний крок. Інакше – виводиться повідомлення про помилку.

Крок 7. Виводиться теорія: «Поділ на нуль у Python створює помилку, оскільки відповідь не може бути обчислена.» і приклад. Перехід на наступний крок.

Крок 8. Виводиться завдання: «Вкажіть відповідь, яка спричинить ZeroDivisionError.

>>> 20 / (____ - 5)» і варіанти відповіді:

- 0

- 5
- -5

Якщо вибрано другий варіант, то перехід на крок 0. Інакше – виводиться повідомлення про помилку.

Числа з плаваючою точкою

Крок 1. Виводиться теорія: «Числа з плаваючою точкою використовуються в Python для представлення чисел, які не є цілими числами.

Деякі приклади чисел, які представлені у вигляді чисел з плаваючою точкою, становлять 0,5 та -7,8237591.

Їх можна створити безпосередньо, ввівши число з десятковою точкою, або за допомогою таких операцій, як поділ на цілі числа. Додаткові нулі в кінці числа ігноруються.» і приклад. Перехід на наступний крок.

Крок 2. Виводиться завдання: «Які з них не зберігатимуться як число з плаваючою точкою?» і варіанти відповіді:

- 2/4
- 7
- 7.0

Якщо вибрано другий варіант, то перехід на наступний крок. Інакше – виводиться повідомлення про помилку.

Крок 3. Виводиться теорія: «Як ви бачили раніше, ділення будь-яких двох цілих чисел дає число з плаваючою точкою.

Число з плаваючою точкою також виробляється шляхом виконання операції на двох числах з плаваючою точкою, або на числі з плаваючою точкою і на цілому.» і приклад. Перехід на наступний крок.

Крок 4. Виводиться завдання: «Заповніть пробіл виводом цього коду.», слід заповнити комірку:

>>> 1 + 2 + 3 + 4.0 + 5

____

Якщо вказано «15.0», то перехід на крок 0. Інакше – виводиться повідомлення про помилку.

Інші числові операції

Крок 1. Виводиться теорія: «Крім додавання, віднімання, множення та ділення, Python також підтримує експоненцію, яка є підняттям одного числа до сили іншого. Ця операція виконується за допомогою двох зірочок.» і приклад. Перехід на наступний крок.

Крок 2. Виводиться завдання: «Заповніть комірку, щоб зробити цей код правильним.», слід заповнити комірку:

>>> (1 + ____) ** 2

16

Якщо вказано «3», то перехід на наступний крок. Інакше – виводиться повідомлення про помилку.

Крок 3. Виводиться теорія: «Щоб визначити коефіцієнт та решту поділу, використовуйте відповідно ділення підлоги та модуля.

Поділ підлоги робиться за допомогою двох передніх косих. Оператор модуля виконується символом відсотка (%). Ці оператори можна використовувати як з плавцями, так і з цілими числами.

Цей код показує, що 6 тричі переходить у 20, а решта, коли 1,25 ділиться на 0,5, - 0,25.» і приклад. Перехід на наступний крок.

Крок 4. Виводиться завдання: «Який результат цього коду?

>>> 7%(5 // 2)» і варіанти відповіді:

- 1
- 0
- 7

Якщо вибрано перший варіант, то перехід на крок 0. Інакше – виводиться повідомлення про помилку.

Рядки

Крок 1. Виводиться теорія: «Якщо ви хочете використовувати текст у Python, ви повинні використовувати рядок.

Рядок створюється введенням тексту між двома одинарними або подвійними лапками.

Коли на консолі Python відображається рядок, вона зазвичай використовує одиничні лапки. Розмежувач, який використовується для рядка, ніяк не впливає на те, як він поводиться.» і приклад. Перехід на наступний крок.

Крок 2. Виводиться завдання: «Заповніть код, щоб створити рядок, що містить “Hello world”.», слід заповнити комірку:

>>>"Hello _____"

Якщо вказано «world», то перехід на наступний крок. Інакше – виводиться повідомлення про помилку.

Крок 3. Виводиться теорія: «Деякі символи не можуть бути безпосередньо включені в рядок. Наприклад, подвійні лапки не можуть бути безпосередньо включені в подвійний рядок цитат; це призведе до того, що вона закінчиться передчасно. Таких символів необхідно уникнути, поставивши перед ними зворотний нахил.

Інші поширені символи, які потрібно уникнути, - це нові рядки та зворотні риски. Подвійні лапки потрібно уникати лише у подвійних рядках цитат, і те саме стосується рядків з цитатами.» і приклад. Перехід на наступний крок.

Крок 4. Виводиться завдання: «Заповніть код, щоб створити рядок, що містить подвійну лапку.», слід заповнити комірку:

>>>" _____"

Якщо вказано «\"», то перехід на наступний крок. Інакше – виводиться повідомлення про помилку.

Крок 5. Виводиться теорія: «Python пропонує простий спосіб уникнути написання "\ n" вручну, щоб уникнути нових рядків у рядку. Створіть рядок із трьома наборами лапок, і нові рядки, створені натисканням клавіші Enter, автоматично переходять для вас.» і приклад. Перехід на наступний крок.

Крок 6. Виводиться завдання: «Заповніть пропущену частину результату.», слід заповнити комірку:

>>> """First line

second line"""

'First line ____ second line'

Якщо вказано «\n», то перехід на крок 0. Інакше – виводиться повідомлення про помилку.

Прості функції введення і виведення.

Крок 1. Виводиться теорія: «Зазвичай програми беруть введення і обробляють його, щоб отримати виведення.

У Python ви можете використовувати функцію print для отримання виводу. Тут відображається текстове зображення на екрані.» і приклад. Перехід на наступний крок.

Крок 2. Виводиться завдання: «Що виведе цей код?

>>> print('print("print")')» і варіанти відповіді:

- An error message
- 'print("print")'
- print("print")

Якщо вибрано третій варіант, то перехід на наступний крок. Інакше – виводиться повідомлення про помилку.

Крок 3. Виводиться теорія: «Щоб отримати вхід від користувача в Python, ви можете використовувати інтуїтивно названу функцію введення.

Функція пропонує користувачеві ввести і повертає те, що вони вводяться як рядок (із вмістом автоматично виходить).» і приклад. Перехід на наступний крок.

Крок 4. Виводиться завдання: «Заповніть комірку для запиту користувача.», слід заповнити комірку:

>>>_____("Enter a number:")

Якщо вказано «input», то перехід на крок 0. Інакше – виводиться повідомлення про помилку.

Строкові операції

Крок 1. Виводиться теорія: «Як і у цілих чисел і чисел з плаваючою точкою, рядки в Python можна додавати, використовуючи процес, званий конкатенацією, який можна зробити на будь-яких двох рядках.

При об'єднанні рядків не має значення, чи створені вони з одинарними або подвійними лапки.» і приклад. Перехід на наступний крок.

Крок 2. Виводиться завдання: «Заповніть пропущений рядок.», слід заповнити комірку:

>>> "Hello " + '_____

'Hello world'

Якщо вказано «world'», то перехід на наступний крок. Інакше – виводиться повідомлення про помилку.

Крок 3. Виводиться теорія: «Навіть якщо ваші рядки містять числа, вони все одно додаються як рядки, а не цілі числа. Додавання рядка до числа створює помилку, оскільки, хоч вони і можуть виглядати схоже, це дві різні сутності.» і приклад. Перехід на наступний крок.

Крок 4. Виводиться завдання: «Який рядок коду видає помилку?» і варіанти відповіді:

- 3 + 4
- "one" + "2"

- "7" + 'eight'
- '5' + 6

Якщо вибрано четвертий варіант, то перехід на наступний крок. Інакше – виводиться повідомлення про помилку.

Крок 5. Виводиться теорія: «Рядки також можна помножити на цілі числа. Це створює повторну версію початкового рядка. Порядок рядка та цілого числа не має значення, але рядок зазвичай стає першим.

Рядки не можна помножити на інші рядки. Рядки також не можна помножити на числа з плаваючою точкою, навіть якщо числа з плаваючою точкою - цілі числа.» і приклад. Перехід на наступний крок.

Крок 6. Виводиться завдання: «Заповніть пробіл виводом цього коду.», слід заповнити комірку:

>>> print(3 * '7')

____

Якщо вказано «777», то перехід на крок 0. Інакше – виводиться повідомлення про помилку.

Перетворення типу

Крок 1. Виводиться теорія: «У Python неможливо виконати певні операції через задіяні типи. Наприклад, ви не можете додати два рядки, що містять числа 2 і 3 разом, щоб створити ціле число 5, оскільки операція буде виконуватися над рядками, що робить результат "23".

Рішенням цього є перетворення типів.

У цьому прикладі слід використати функцію int.» і приклад. Перехід на наступний крок.

Крок 2. Виводиться завдання: «Що виводить цей код?

>>> int("3" + "4")» і варіанти відповіді:

- 34
- "34"
- "7"

Якщо вибрано перший варіант, то перехід на наступний крок. Інакше – виводиться повідомлення про помилку.

Крок 3. Виводиться теорія: «Іншим прикладом перетворення типів є перетворення користувачем даних (що є рядком) до чисел (цілих чисел або чисел з плаваючою точкою), щоб забезпечити виконання обчислень.» і приклад. Перехід на наступний крок.

Крок 4. Виводиться завдання: «Що виводить цей код?

>>> float("210" * int(input("Enter a number:")))

Enter a number: 2» і варіанти відповіді:

- 420
- "420.0"
- "210210"
- 210210.0

Якщо вибрано четвертий варіант, то перехід на крок 0. Інакше – виводиться повідомлення про помилку.

Змінні

Крок 1. Виводиться теорія: «Змінні відіграють дуже важливу роль у більшості мов програмування, і Python не є винятком. Змінна дозволяє зберігати значення шляхом присвоєння йому імені, яке можна використовувати для посилання на значення пізніше в програмі.

Щоб призначити змінну, використовуйте один знак рівності. На відміну від більшості рядків коду, які ми розглянули до цього часу, він не дає жодного результату на консолі Python.» і приклад. Перехід на наступний крок.

Крок 2. Виводиться завдання: «Що виводить цей код?

>>> spam = "eggs"

>>> print(spam * 3)» і варіанти відповіді:

- eggseggseggs
- spamspamspam

- "spamspamspam"

Якщо вибрано перший варіант, то перехід на наступний крок. Інакше – виводиться повідомлення про помилку.

Крок 3. Виводиться теорія: «Змінні можна перепризначати стільки разів, скільки вам потрібно, щоб змінити їх значення.

У Python змінні не мають конкретних типів, тому ви можете призначити рядок змінній, а пізніше призначити цілу цілу величину.» і приклад. Перехід на наступний крок.

Крок 4. Виводиться завдання: «Вкажіть правильну відповідь, щоб код працював правильно.

```
>>> x = 5
>>> y = ____
>>> print(x + y)
```

12» і варіанти відповіді:

- 7
- "7"
- = 7

Якщо вибрано перший варіант, то перехід на наступний крок. Інакше – виводиться повідомлення про помилку.

Крок 5. Виводиться теорія: «Певні обмеження застосовуються щодо символів, які можуть використовуватися в іменах змінних Python. Дозволені лише символи - це літери, цифри та підкреслення. Крім того, вони не можуть починати з цифр.

Недотримання цих правил призводить до помилок.» і приклад. Перехід на наступний крок.

Крок 6. Виводиться завдання: «Яке з них є дійсним ім'ям змінної в Python?» і варіанти відповіді:

- a-variable-name
- A_VARIABLE_NAME

- a variable name

Якщо вибрано другий варіант, то перехід на наступний крок. Інакше – виводиться повідомлення про помилку.

Крок 7. Виводиться теорія: «Спробу посилання на змінну, яку ви не призначили, викликає помилку.

Ви можете використовувати оператор del для видалення змінної, що означає, що посилання з імені на значення видаляється, а спроба використання змінної викликає помилку. Видалені змінні можна переназначити пізніше як звичайні.

Ви також можете взяти значення змінної з даних користувача.» і приклад. Перехід на наступний крок.

Крок 8. Виводиться завдання: «Що виводить цей код?», слід заповнити комірку:

```
>>> spam = 2
>>> eggs = 3
>>> del spam
>>> eggs = 4
>>> spam = 5
>>> print(spam * eggs)
```

____

Якщо вказано «20», то перехід на крок 0. Інакше – виводиться повідомлення про помилку.

Операції «на місті»

Крок 1. Виводиться теорія: «Місцеві оператори дозволяють писати код типу "x = x + 3" більш стисло, як "x + = 3".

Те ж саме можливо і з іншими операторами, такими як -, *, / і%.» і приклад. Перехід на наступний крок.

Крок 2. Виводиться завдання: «Що виводить цей код?», слід заповнити комірку:

```
>>> x = 4
>>> x *= 3
>>> print(x)
```

____

Якщо вказано «12», то перехід на наступний крок. Інакше – виводиться повідомлення про помилку.

Крок 3. Виводиться теорія: «Ці оператори можуть використовуватися і для інших типів, крім чисел, таких як рядки.» і приклад. Перехід на наступний крок.

Крок 4. Виводиться завдання: «Який результат цього коду?», слід заповнити комірку:

```
>>> x = "a"
>>> x *= 3
print(x)
```

____

Якщо вказано «ааа», то перехід на крок 0. Інакше – виводиться повідомлення про помилку.

Використання редактора

Крок 1. Виводиться теорія: «Поки ми використовували лише Python з консоллю, вводячи та запускаючи по одному рядку коду за один раз.

Фактичні програми створюються по-різному; багато рядків коду записуються у файл, а потім виконуються за допомогою інтерпретатора Python.

У IDLE це можна зробити, створивши новий файл, ввівши якийсь код, збереживши файл та запустивши його. Це можна зробити або за допомогою меню, або за допомогою комбінацій клавіш Ctrl-N, Ctrl-S і F5.

Кожен рядок коду у файлі інтерпретується так, ніби ви ввели його по черзі на консолі.» і приклад. Перехід на наступний крок.

Крок 2. Виводиться завдання: «Які рядки виконуються під час запуску файлу Python?» і варіанти відповіді:

- Перші 100 рядків
- Кожен рядок
- Лише перший рядок

Якщо вибрано другий варіант, то перехід на крок 0. Інакше – виводиться повідомлення про помилку.

3.3. Блок-схема алгоритму роботи тренажеру

На рисунках 3.2 – 3.3 зображена блок-схема алгоритму роботи тренажеру.

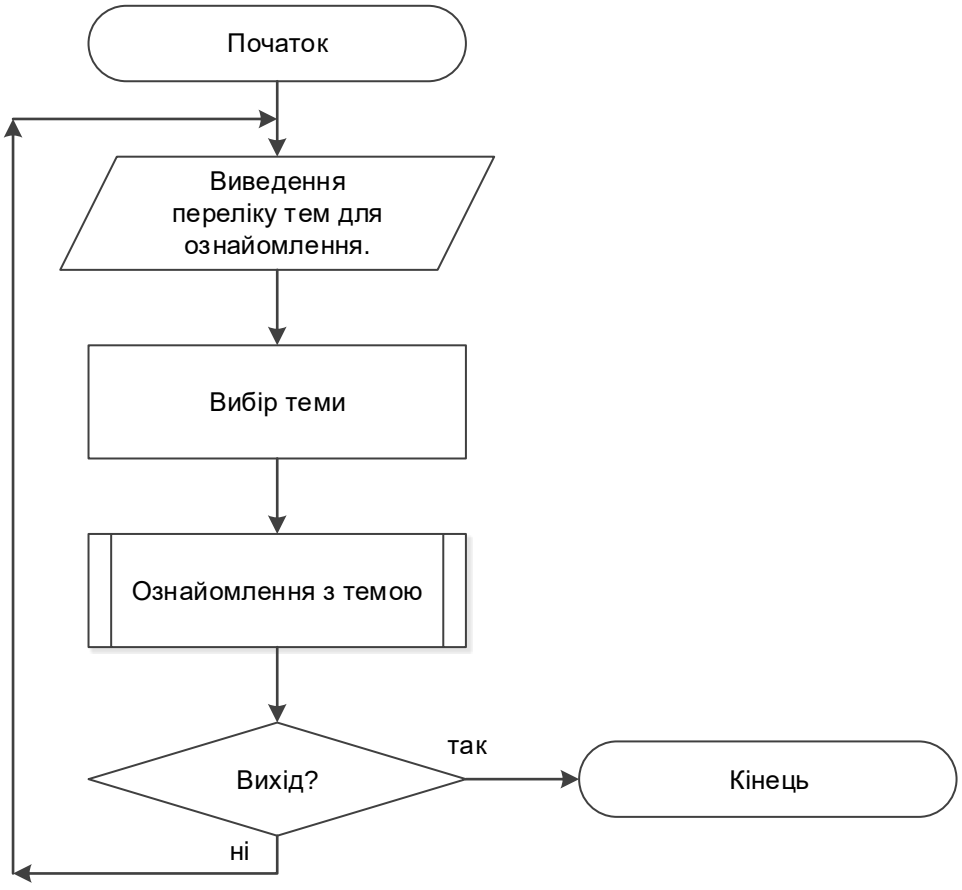

Рисунок 3.2 – Блок-схема алгоритму роботи тренажеру

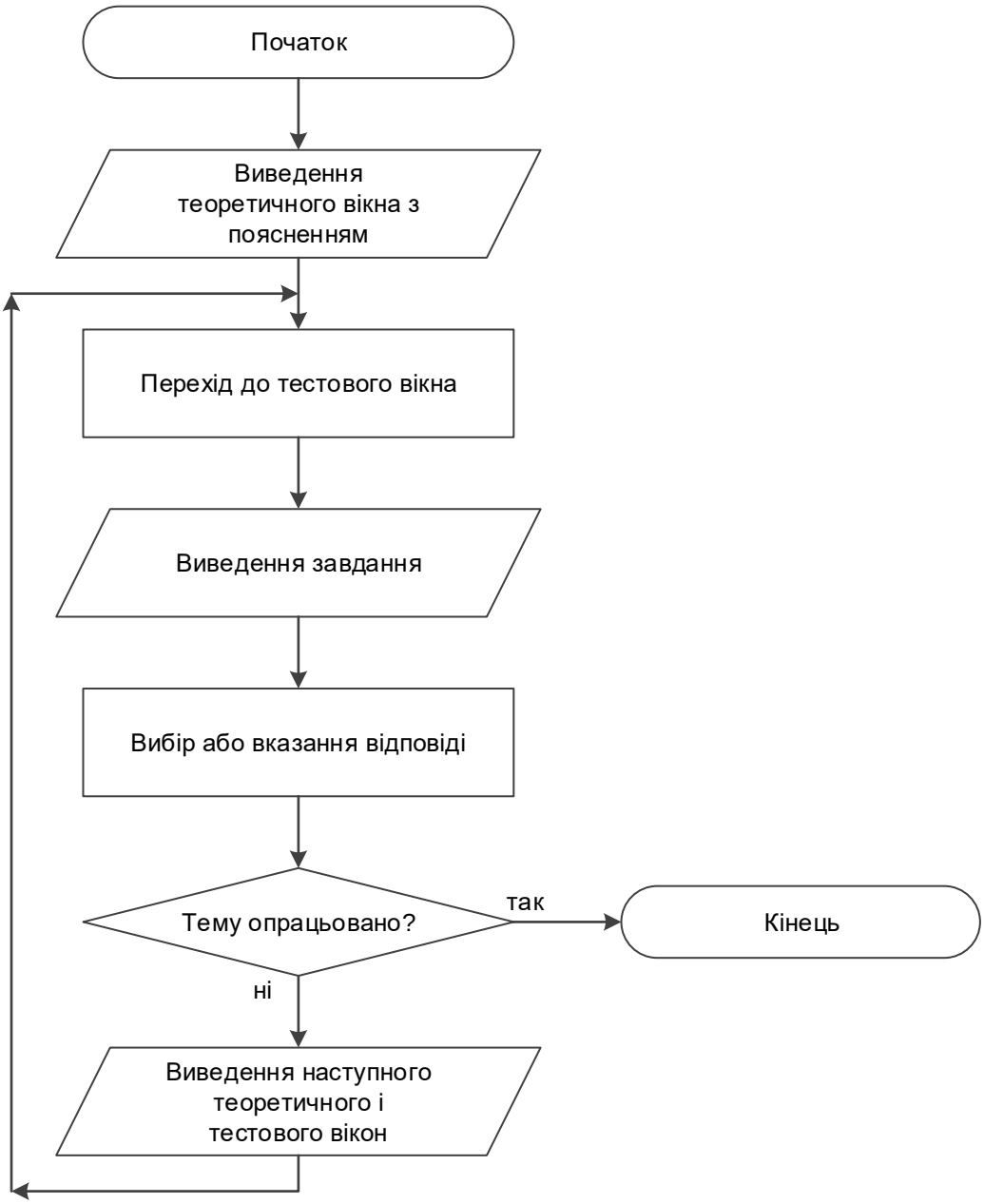

Рисунок 3.3 – Блок-схема процесу «Ознайомлення з темою»

4. ПРАКТИЧНА ЧАСТИНА

4.1. Обґрунтування вибору програмних засобів

JavaFX — платформа та набір інструментів для створення насичених інтернет-застосунків (англ. Rich Internet Applications, RIA) з можливістю підвантаження медіа та змісту. Вперше продемонстровано Sun Microsystems на Міжнародній конференції Java-розробників JavaOne у травні 2007. JavaFX включає в себе набір утиліт, за допомогою яких веб-розробники та дизайнери можуть швидко створювати та надавати розвинуті інтернет-застосунки для десктопів, мобільних пристроїв, телебачення та інших платформ.

JavaFX складається з JavaFX Script і JavaFX Mobile. Починаючи з випуску JavaFX 2.0 забезпечено можливість створення JavaFX-застосунків, написаних цілком мовою Java. Для розробки застосунків доступний багатий графічний і мультимедійний API, що спрощує створення візуальних програм [5].

FXML - це мова розмітки користувальницького інтерфейсу на основі XML, створена корпорацією Oracle для визначення користувальницького інтерфейсу програми JavaFX.

Він надає зручну альтернативу для побудови таких графіків у процедурному коді та ідеально підходить для визначення користувальницького інтерфейсу програми JavaFX, оскільки ієрархічна структура документа XML тісно паралельна структурі графічного сценарію JavaFX. Однак все, що створено або реалізовано в FXML, можна виразити безпосередньо за допомогою JavaFX [6].

4.2. Опис процесу програмної реалізації

Спочатку оглянуто основні принципи роботи JavaFX, FXML [7-10].

За запуск тренажеру і підключення таблиці стилів відповідає наступний код:

```
@Override
public void start(Stage stage) throws Exception {
    Parent root =
FXMLLoader.load(getClass().getResource("FXMLDocument.fxml"));

    Scene scene = new Scene(root);

scene.getStylesheets().add(getClass().getResource("StyleSheet.css").t
oExternalForm());

    stage.setScene(scene);
    stage.show();
}
```

Всі панелі описано у fxml-файлі. Так стартова сторінка виводиться завдяки:

```
<AnchorPane prefHeight="550" prefWidth="500">
    <children>
        <Label styleClass="theme" layoutX="20" layoutY="50"
prefWidth="450" alignment="CENTER"  text="Тренажер для вивчення"  />
        <Label styleClass="theme" layoutX="20" layoutY="90"
prefWidth="450" alignment="CENTER"  text="основ мови програмування
Python"  />
        <Button styleClass="button" layoutX="140" layoutY="260"
text="Розпочати тренування" onAction="#start" />
        <Label styleClass="creator" layoutX="40" layoutY="470"
prefWidth="450"  text="Розробник: Мельницький Я.В."  />
    </children>
</AnchorPane>
```

Для виведення сторінки з переліком тем служить:

```
<AnchorPane fx:id="s0" prefWidth="500">
    <children>
        <VBox layoutX="85" layoutY="15" spacing="10.0"
prefWidth="350">
            <children>
                <Label styleClass="head" prefWidth="350"
alignment="CENTER"  text="Оберіть тему"  />
                <RadioButton text="Що таке Python?"
toggleGroup="$rbg" fx:id="r1" />
                <RadioButton text="Ваша перша програма"
toggleGroup="$rbg" fx:id="r2"/>
                <RadioButton text="Прості операції"
toggleGroup="$rbg" fx:id="r3"/>
                <RadioButton text="Числа з плаваючою точкою"
toggleGroup="$rbg" fx:id="r4"/>
```

```
                <RadioButton text="Інші числові операції"
toggleGroup="$rbg" fx:id="r5"/>
                <RadioButton text="Рядки" toggleGroup="$rbg"
fx:id="r6"/>
                <RadioButton text="Прості функції введення і
виведення" toggleGroup="$rbg" fx:id="r7"/>
                <RadioButton text="Строкові операції"
toggleGroup="$rbg" fx:id="r8"/>
                <RadioButton text="Перетворення типу"
toggleGroup="$rbg" fx:id="r9"/>
                <RadioButton text="Змінні" toggleGroup="$rbg"
fx:id="r10"/>
                <RadioButton text="Операції «на місті»"
toggleGroup="$rbg" fx:id="r11"/>
                <RadioButton text="Використання редактора"
toggleGroup="$rbg" fx:id="r12"/>
            </children>
        </VBox>
        <Button prefWidth="150" styleClass="button" layoutX="295"
layoutY="475" text="Далі" onAction="#choose" />
    </children>
</AnchorPane>
```

Теоретичні та тестові панелі відображаються аналогічним чином (див. Додаток А).

Стилі всіх кнопок, написів прописуються у файлі StyleSheet.css.

```
.root {
    -fx-font-size: 15px;
    -fx-font-family: sans-serif;
}
.theme {
    -fx-font-size: 24px;
    -fx-font-weight: bold;
    -fx-font-style: italic;
}
.creator {
    -fx-font-size: 16px;
    -fx-font-style: italic;
    -fx-text-decoration: underline;
}
.button {
    -fx-padding: 20px;
    -fx-background-color: grey;
    -fx-cursor: hand;
    -fx-font-size: 18px;
    -fx-font-weight: bold;
}
.head {
    -fx-font-size: 24px;
    -fx-font-weight: bold;
    -fx-padding: 0 0 30px 0;
}
.theory {
    -fx-wrap-text: true;
    -fx-text-align: justify;
    -fx-direction: rtl;
    -fx-padding: 0 0 5px 0;
}
.task {
```

```
    -fx-wrap-text: true;
    -fx-padding: 0 0 15px 0;
}
.error {
    -fx-text-fill: red;
    -fx-font-size: 18px;
}

.example {
    -fx-wrap-text: true;
    -fx-padding: 0 0 0 35px;
}
```

Наступним кроком оголошено змінні, що використовуються у панелях, і глобальні змінні step (поточний крок алгоритму) та theme (номер теми).

```
@FXML
private Region s0, s1, s2, s3;
@FXML
private Pane stackConsole;
@FXML
private ToggleGroup rbg;
@FXML
private Label texthead, theory1, example1, theory2, example2,
taskhead1, task1, taskhead2, task2, h1text, h2text, v1text, v2text,
error1, error2;
@FXML
private RadioButton t1, t2, t3, t4, r1, r2, r3, r4, r5, r6 ,r7 ,r8,
r9, r10, r11, r12;
@FXML
private TextField hfield, vfield;
@FXML
private HBox hbox, vbox;
int step = 1, theme = 1;
```

Було створено функції для виведення вмісту панелі для кожної теми відповідно до поточного кроку алгоритму: showStep1(), showStep2(), showStep3(), showStep4(), showStep5(), showStep6(), showStep7(), showStep8(), showStep9(), showStep10(), showStep11(), showStep12().

```
private void showStep1() {
    switch(step) {
        case 1:
            theory1.setText("\tPython - мова програмування високого
рівня, що має додатки в багатьох сферах, включаючи веб-програмування,
сценарії, наукові обчислення та штучний інтелект.\n" +
                    "\tВін дуже популярний і використовується такими
організаціями, як Google, NASA, CIA та Disney.");
            example1.setText("");
            theory2.setText("");
            example2.setText("");
            stackConsole.getChildren().setAll(s1);
            break;
        case 2:
```

```
                task1.setText("Python - це:");
                t1.setText("Мова програмування");
                t2.setText("Набір інструментів редагування");
                t3.setText("Середовище розвитку");
                t4.setText("");
                t4.setVisible(false);
                stackConsole.getChildren().setAll(s2);
                break;
            case 3:
                theory1.setText("\tТри основні версії Python - це 1.x,
2.x та 3.x. Вони підрозділяються на незначні версії, такі як 2.7 та
3.3.\n" +
                        "\tКод, написаний для Python 3.x, гарантовано працює
у всіх майбутніх версіях. На даний момент використовуються Python
версії 2.x та 3.x.\n" +
                        "\tУ Python є кілька різних реалізацій, написаних
різними мовами.\n" +
                        "\tВерсія, що використовується в цьому курсі,
CPython, є найбільш популярною на сьогоднішній день.");
                example1.setText("");
                theory2.setText("");
                example2.setText("");
                stackConsole.getChildren().setAll(s1);
                break;
            case 4:
                task1.setText("Яке з цих тверджень є правдивим?");
                t1.setText("CPython - це реалізація Python");
                t2.setText("Python 1.7 - це найбільш широко
використовувана версія");
                t3.setText("Код Python повинен бути завжди
компільований");
                t4.setText("");
                t4.setVisible(false);
                stackConsole.getChildren().setAll(s2);
                break;
        }
    }
```

Також щоб виводити вміст в залежності від номеру теми реалізовано showStep().

```
    private void showStep() {
        switch(theme) {
            case 1:
                showStep1();
                break;
            case 2:
                showStep2();
                break;
            case 3:
                showStep3();
                break;
            case 4:
                showStep4();
                break;
            case 5:
                showStep5();
                break;
            case 6:
                showStep6();
                break;
```

```
            case 7:
                showStep7();
                break;
            case 8:
                showStep8();
                break;
            case 9:
                showStep9();
                break;
            case 10:
                showStep10();
                break;
            case 11:
                showStep11();
                break;
            case 12:
                showStep12();
                break;
        }
    }
```

Для роботи всіх кнопок було розроблено події. Кнопка на стартовій сторінці працює завдяки start(ActionEvent event).

```
    private void start(ActionEvent event) {
        r1.setSelected(true);
        stackConsole.getChildren().setAll(s0);
        hbox.setAlignment(Pos.CENTER_LEFT);
        vbox.setAlignment(Pos.CENTER_LEFT);
    }
```

Вибір теми здійснюється за допомогою події choose(ActionEvent event). При цьому в заголовку тренажеру вказується назва теми і відбувається перехід до теоретичної панелі.

```
@FXML
private void choose(ActionEvent event) {
    RadioButton rb = (RadioButton) rbg.getSelectedToggle();
    error1.setVisible(false);
    error2.setVisible(false);
    v1text.setText("");
    vfield.setVisible(false);
    v2text.setText("");
    switch(rb.getId()) {
        case "r1":
            texthead.setText("Що таке Python?");
            taskhead1.setText("Що таке Python?");
            taskhead2.setText("Що таке Python?");
            theme = 1;
            break;
        case "r2":
            texthead.setText("Ваша перша програма");
            taskhead1.setText("Ваша перша програма");
            taskhead2.setText("Ваша перша програма");
            theme = 2;
            break;
```

```
            .
            .
            .
            case "r12":
                texthead.setText("Використання редактора");
                taskhead1.setText("Використання редактора");
                taskhead2.setText("Використання редактора");
                theme = 12;
                break;
        }
        showStep();
        stackConsole.getChildren().setAll(s1);
    }
```

Кнопка «Назад» здійснює перехід до попереднього кроку і виконується завдяки back(ActionEvent event). Якщо це перший крок, то перехід до вибору теми.

```
    @FXML
    private void back(ActionEvent event) {
        error1.setVisible(false);
        error2.setVisible(false);
        r1.setSelected(true);
        hfield.setText("");
        if(step == 1) {
            stackConsole.getChildren().setAll(s0);
        } else {
            if(theme == 12) { step = 2; }
            step--;
            showStep();
        }
    }
```

Кнопка «Далі» відповідає за перехід до наступного кроку. Якщо на поточному кроці відображається теоретичне вікно, то просто відбувається перехід. Якщо виводиться практичне вікно, то спочатку перевіряється відповідь: вірно – перехід до наступного, невірно – повідомлення про помилку. На останьому кроці відбувається перехід до вибору теми (див. Додаток А).

```
    @FXML
    private void next(ActionEvent event) {
        error1.setVisible(false);
        error2.setVisible(false);
        if(theme != 12 && (step == 1 || step == 3 || step == 5 || step ==
    7)) {
            step++;
            showStep();
        } else {
            switch(theme) {
                case 1:
                    switch(step) {
                        case 2:
```

```
                if(t1.isSelected()) {
                    step++;
                    t1.setSelected(false);
                    showStep();
                } else {
                    error1.setVisible(true);
                    error2.setVisible(true);
                }
                break;
            ...
        }
        break;
.
.
.
```

4.3. Інструкція по використанню тренажеру

Стартове вікно містить тему тренажеру, інформацію про розробника та кнопку «Розпочати тренування» (рис. 4.1), при натисненні відображається вікно з назвами тем.

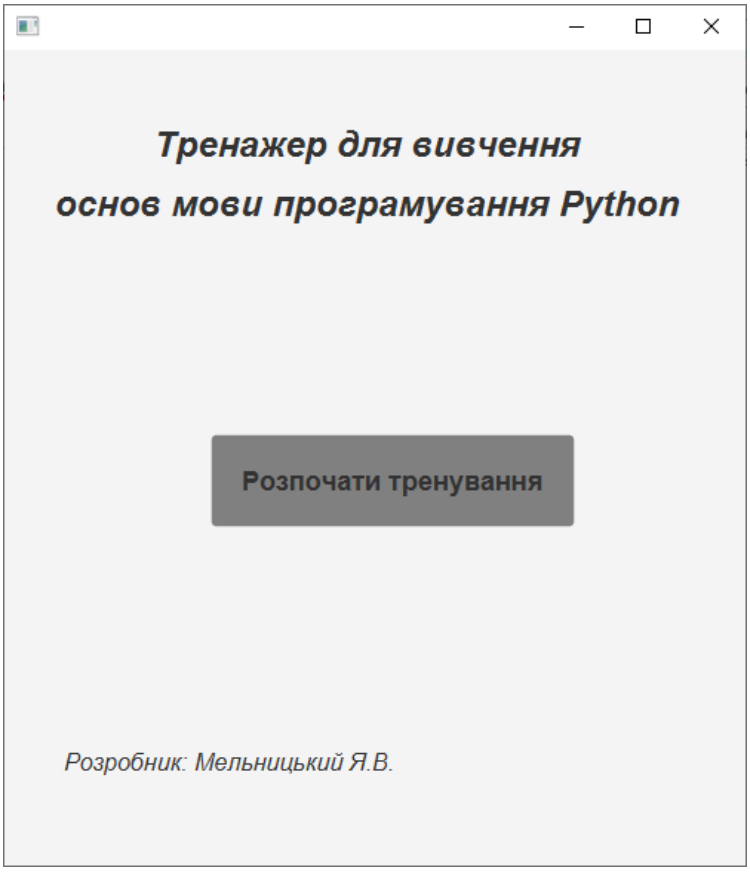

Рисунок 4.1 – Стартове вікно

Вікно тем має їх перелік і кнопку «Далі» для переходу до теоретичного вікна обраної теми (рис. 4.2). Автоматично вибирається перша тема.

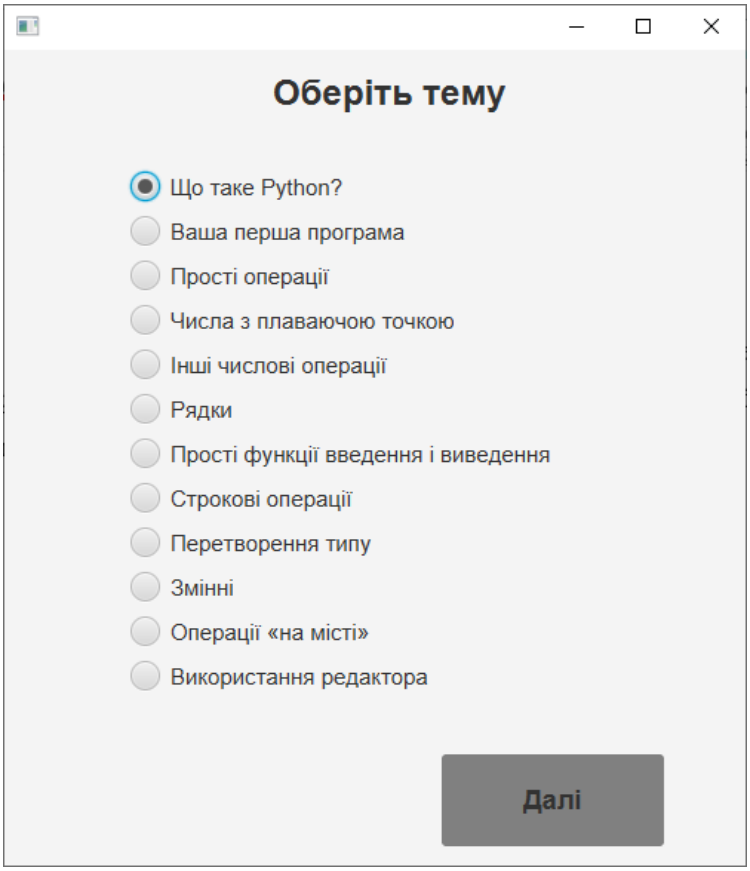

Рисунок 4.2 – Вікно тем

На теоретичному вікні з’являється пояснення відносно обраної теми. На кожному вікні завжди є дві кнопки “Далі” та “Назад” для навігації по програмі (рис. 4.3). По натисненні кнопки “Далі” – текст зникає і з’являється тест за щойно поясненою темою.

Рисунок 4.3 – Теоретичне вікно

В тестовому вікні відображаються завдання (рис. 4.4).

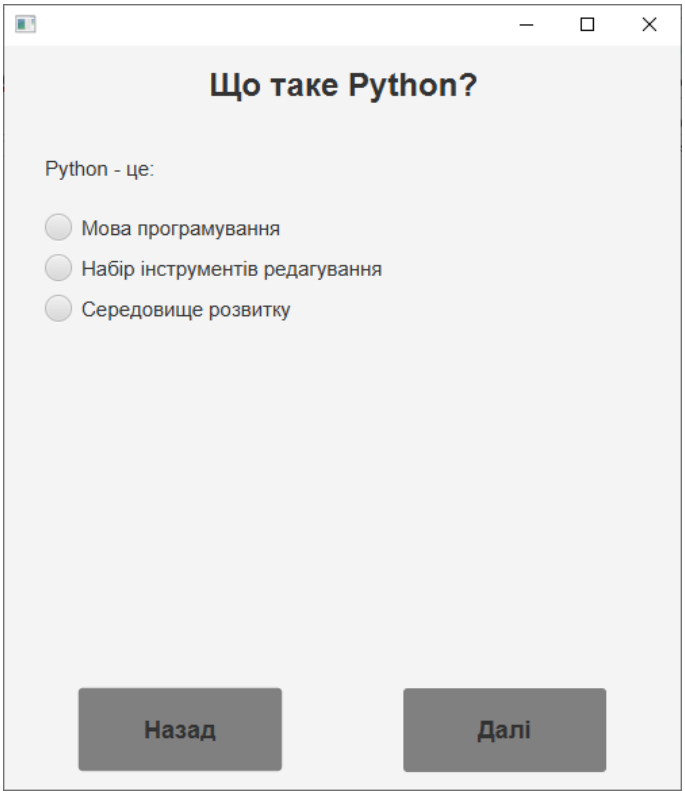

Рисунок 4.4 – Тестове вікно

При обиранні невірної відповіді на вікні з’явиться повідомлення щодо повернення до попереднього вікна з теоретичною частиною для повторного вивчення (рис. 4.5), а при обиранні правильної відповіді – вміст вікна змінюється на теоретичну частину наступного кроку (рис. 4.6).

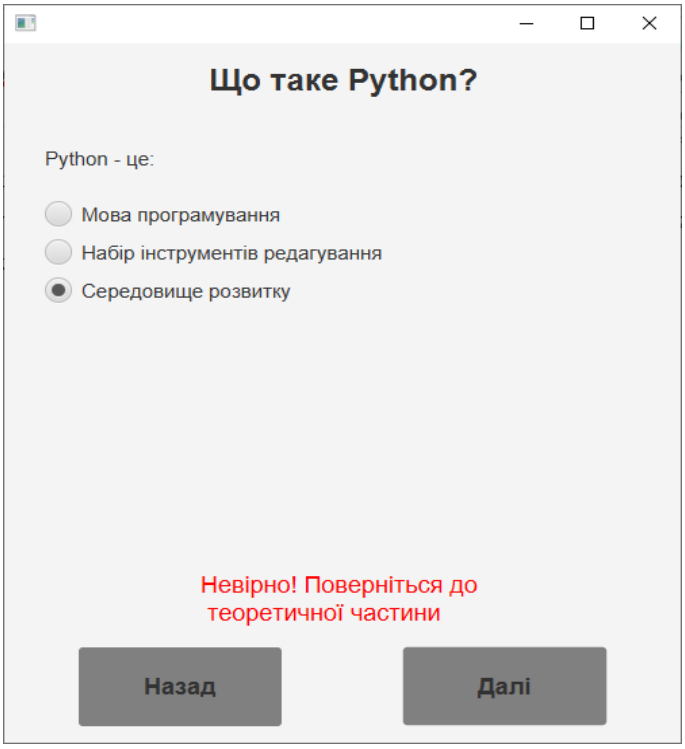

Рисунок 4.5 – Повідомлення про помилку

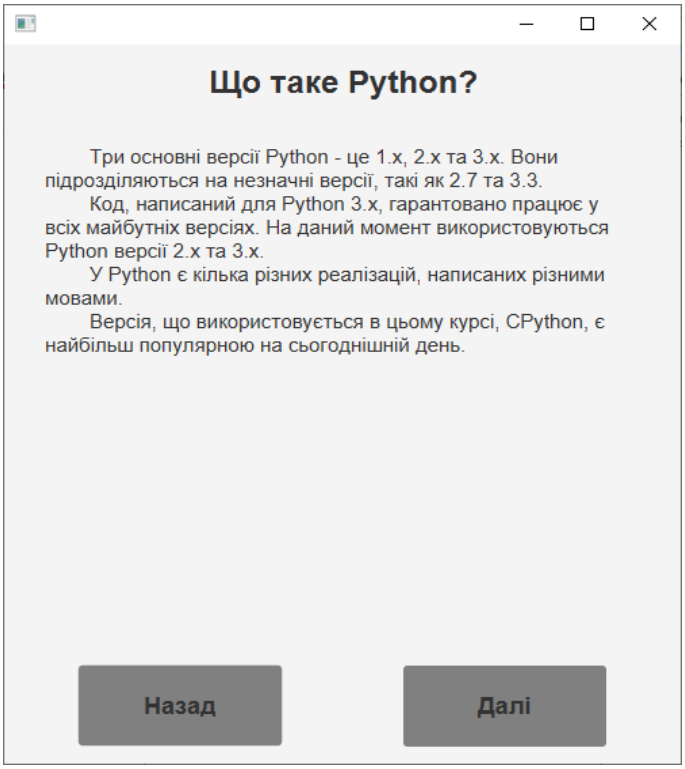

Рисунок 4.6 – Теоретична частина наступного кроку

Якщо тему опрацьовано повністю, то відбувається перехід до вікна з темати і вибирається наступна (рис. 4.7).

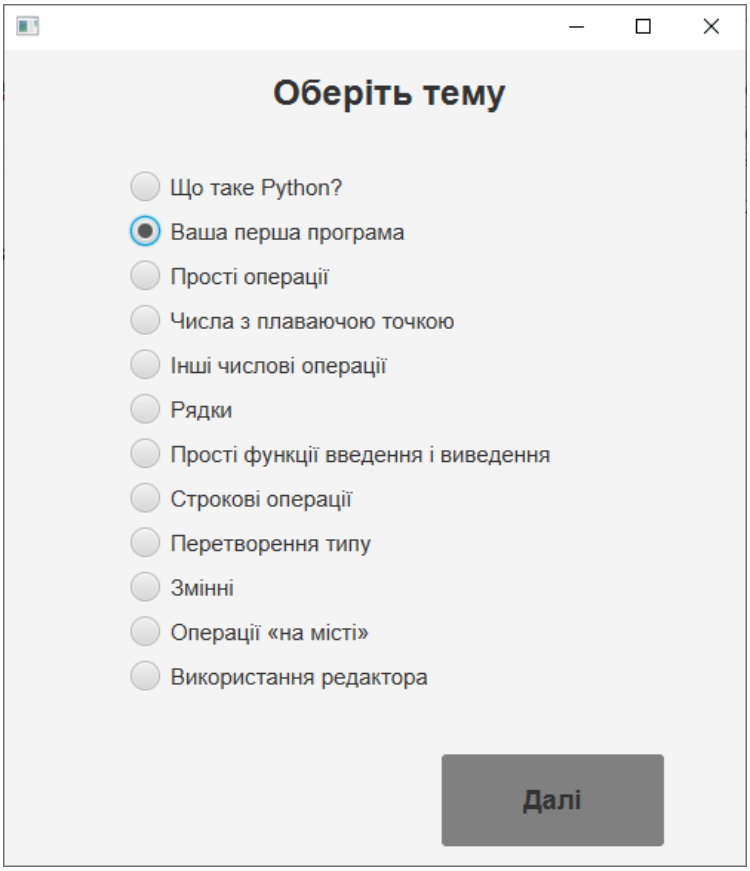

Рисунок 4.7 – Автоматичний вибір наступної теми

Серед тестових завдань в темах крім вибору одного з варіантів відповідей є також тести, в яких потрібно заповнити пусту комірку (рис. 4.8). В даному випадку слід вводити лише англійські літери, розділювач в числах вказувати через «.». Якщо хоч один символ буде відрізнятися від вірної відповіді, то виведеться повідомлення про помилку.

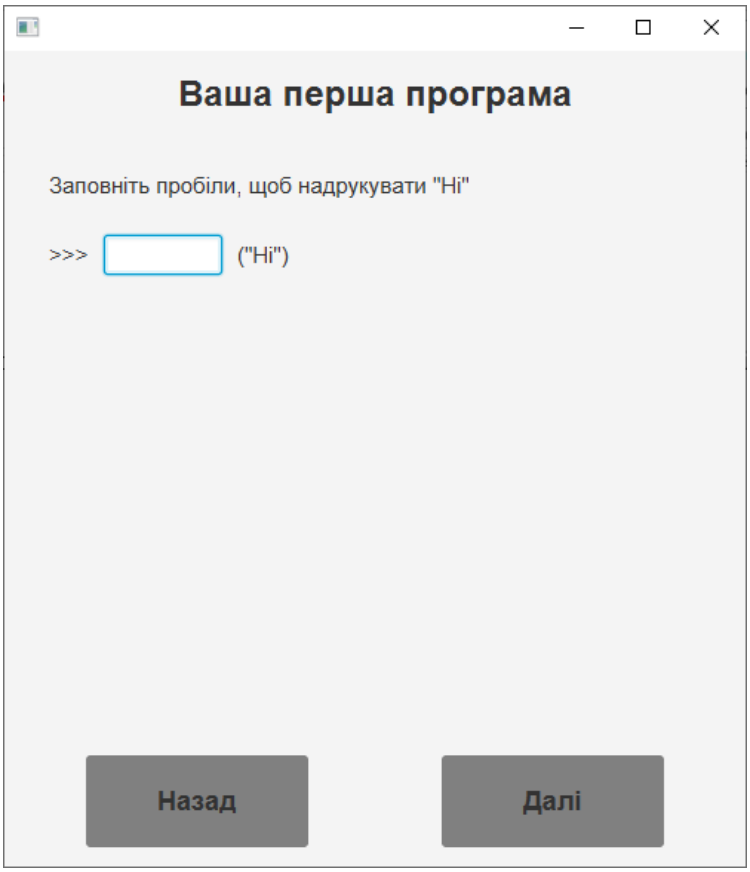

Рисунок 4.8 – Інший тип тестових завдань

ВИСНОВКИ

Впровадження інформаційно-комунікаційних технологій в навчальний процес дозволяє підвищити ефективність навчання та якість формування вмінь студентів за допомогою віртуальних тренажерів. Саме тому в рамках дипломної роботи ставилися задачі по розробці тренажера для вивчення основ мови програмування Python.

Основні результати роботи:

- підготовлено матеріал, що використовується при реалізації тренажеру;
- розроблено алгоритм тренажеру для вивчення основ мови програмування Python;
- складено блок-схему алгоритму роботи тренажеру;
- розроблено навчальний тренажер з даної теми.

В алгоритмі описується процес ознайомлення з такими темами:

- Що таке Python?
- Ваша перша програма;
- Прості операції;
- Числа з плаваючою точкою;
- Інші числові операції;
- Рядки;
- Прості функції введення і виведення;
- Строкові операції;
- Перетворення типу;
- Змінні;
- Операції «на місті»;
- Використання редактора.

Виводиться спочатку теоретичне вікно з поясненням відносно обраної теми, а потім тестове. В ньому перевіряється засвоєння матеріалу у

вигляді завдання з вибором одного з варіантів відповідей або заповнення пустої комірки.

При обиранні невірної відповіді на вікні з’явиться повідомлення щодо повернення до попереднього вікна з теоретичною частиною для повторного вивчення, а при обиранні правильної відповіді – вміст вікна змінюється на теоретичну частину наступної теми.

Розроблений тренажер має таку структуру:

- стартова сторінка:
  - тема тренажеру;
  - перехід до тренування;
  - інформація про розробника
- сторінка з вибором теми:
  - перелік тем;
  - перехід до виконання обраної теми;
- сторінка з матеріалом:
  - теоретичний матеріал;
  - приклад для засвоєння матеріалу;
  - повернутися на попередній крок;
  - перейти на наступний крок.

Всі панелі програми описано у fxml-файлі. Стилі всіх кнопок, написів прописуються у файлі StyleSheet.css.

Після цього оголошено змінні, що використовуються у панелях, і глобальні змінні step та theme.

Було створено функції для виведення вмісту панелі для кожної теми відповідно до поточного кроку алгоритму Також щоб виводити вміст в залежності від номеру теми реалізовано showStep().

Для роботи всіх кнопок було розроблено події.

Всі вимоги, описані в постановці задачі, були виконані.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Ємець О. О. Методичні рекомендації до виконання кваліфікаційної бакалаврської роботи для студентів 4 курсу напряму 6.040302 «Інформатика» / О. О. Ємець, Ол-ра О. Ємець. – Полтава : ПУЕТ, 2010. – 74 с.
2. Ємець О. О. Методичні рекомендації щодо оформлення пояснювальних записок до курсових проектів (робіт) для студентів за освітньою програмою «Комп'ютерні науки» спеціальності 122 «Комп'ютерні науки та інформаційні технології», «Комп'ютерні науки» галузь знань - 12 «Інформаційні технології» / О. О. Ємець . – Полтава : РВВ ПУЕТ, 2017. – 71 с.
3. Мазур М.П. Розвиток дистанційного навчання в Україні як складової інформатизації сучасного суспільства / М.П.Мазур // Інформатика та інформаційні технології в навчальних закладах. – 2007, №1 . – с. 71-75.
4. Python 3 Tutorial Module 1: Basic Concepts [Електронний ресурс] / SoloLearn. – Режим доступу: https://www.sololearn.com/Course/Python/
5. JavaFX [Електронний ресурс] / Матеріал з Вікіпедії — вільної енциклопедії. – Режим доступу: https://uk.wikipedia.org/wiki/JavaFX
6. FXML [Електронний ресурс] / From Wikipedia, the free encyclopedia. – Режим доступу: https://en.wikipedia.org/wiki/FXML
7. Учебник по JavaFX: Привет, мир // Перевод статті «JavaFX Tutorial: Hello world». – Vojtech Ruzicka. – Режим доступу: https://habr.com/ru/post/474498/
8. Учебник по JavaFX: FXML и SceneBuilder // Перевод статті «JavaFX Tutorial: FXML and SceneBuilder». – Vojtech Ruzicka. – Режим доступу: https://habr.com/ru/post/474982/
9. Учебник по JavaFX: основные макеты // Перевод статті «JavaFX Tutorial: Basic layouts» ». – Vojtech Ruzicka. – Режим доступу: https://habr.com/ru/post/475882/

10. Учебник по JavaFX: CSS-стилизация // Перевод статті «JavaFX Tutorial: CSS Styling» ». – Vojtech Ruzicka. – Режим доступу: https://habr.com/ru/post/477924/
11. Бібліографічний запис. Бібліографічний опис. Загальні вимоги та правила складання: ДСТУ 7.1-2006. – [Чинний від 2007-07-01]. – К. : Держспоживстандарт України, 2007. – 47 с.

ДОДАТОК А. КОД ПРОГРАМИ