**Міністерство освіти і науки України**
**Національний технічний університет України**
**«Київський політехнічний інститут імені Ігоря Сікорського»**
**Факультет інформатики та обчислювальної техніки**
**Кафедра обчислювальної техніки**

# Лабораторна робота №2
**з дисципліни**
**«Основи розробки програмного забезпечення мовою програмування Python»**
**на тему**
**“Використання операторів умовного виконання та циклів у мові програмування Python”**

Виконав:
студент групи ІМ-42
Жила Іван Дмитрович
номер варіанту: 8

Перевірив:
Фещенко К. Ю

Київ 2026

## Мета роботи

Ознайомитися з алгоритмічними структурами керування в мові Python: операторами умовного виконання if, elif, else, а також циклами fori while. Набути практичних навичок створення програм з розгалуженням та повторенням дій.

**Короткі теоретичні відомості:**

*Умовні оператори:* Використовуються для розгалуження алгоритму. if перевіряє умову, elif додає альтернативні перевірки, а else виконується, якщо жодна умова не справдилася.

*Цикл for:* Найкраще підходить для перебору послідовностей (наприклад, діапазону чисел range()), коли кількість повторень відома наперед.

*Цикл while:* Повторює дії, доки виконується певна логічна умова.

*Керування циклами:* break негайно зупиняє цикл, а continue пропускає поточну ітерацію і переходить до наступної.

*PEP 8:* Код має бути читабельним, з відступами у 4 пробіли та зрозумілими іменами змінних.

**Загальне завдання:**
Написати програму, яка:
− вводить початкові дані;
− виконує розрахунки згідно індивідуального завдання;
− виводить результатю

**Індивідуальне завдання -** Побудувати таблицю множення для заданого числа. (8 варіант)

**Текст програми:**

```
# Лабораторна робота No2
# Тема: Використання операторів умовного виконання та
циклів у мові програмування Python.
# Варіант 8 Побудувати таблицю множення для заданого
```

```
числа.

# Введення цілого числа
num = int(input("Введіть число: "))

# Виведення повідомлення
print(f"Таблиця множення для числа {num}:")

# Використовуємо range(0, 11), щоб включити числа від 1 до 10
for i in range(0, 11):
    # Виведення результатів
    print(f"{num} * {i} =", num * i)
```

**Скріншоти:**

```
Введіть число: 5
Таблиця множення для числа 5:
5 * 0 = 0
5 * 1 = 5
5 * 2 = 10
5 * 3 = 15
5 * 4 = 20
5 * 5 = 25
5 * 6 = 30
5 * 7 = 35
5 * 8 = 40
5 * 9 = 45
5 * 10 = 50

Process finished with exit code 0
```

```
Введіть число: 14
Таблиця множення для числа 14:
14 * 0 = 0
14 * 1 = 14
14 * 2 = 28
14 * 3 = 42
14 * 4 = 56
14 * 5 = 70
14 * 6 = 84
14 * 7 = 98
14 * 8 = 112
14 * 9 = 126
14 * 10 = 140

Process finished with exit code 0
```

**Висновки:**
Під час виконання роботи я опанував використання циклічних структур у Python. Для побудови таблиці множення було обрано цикл for, оскільки кількість ітерацій (від 0 до 10) була відома заздалегідь, що робить код компактнішим та читабельнішим. Також було дотримано стандартів оформлення PEP 8 щодо відступів та іменування змінних