Міністерство освіти і науки України
Донецький національний університет імені Василя Стуса
Факультет інформаційних і прикладних технологій
Кафедра інформаційних технологій

З В І Т

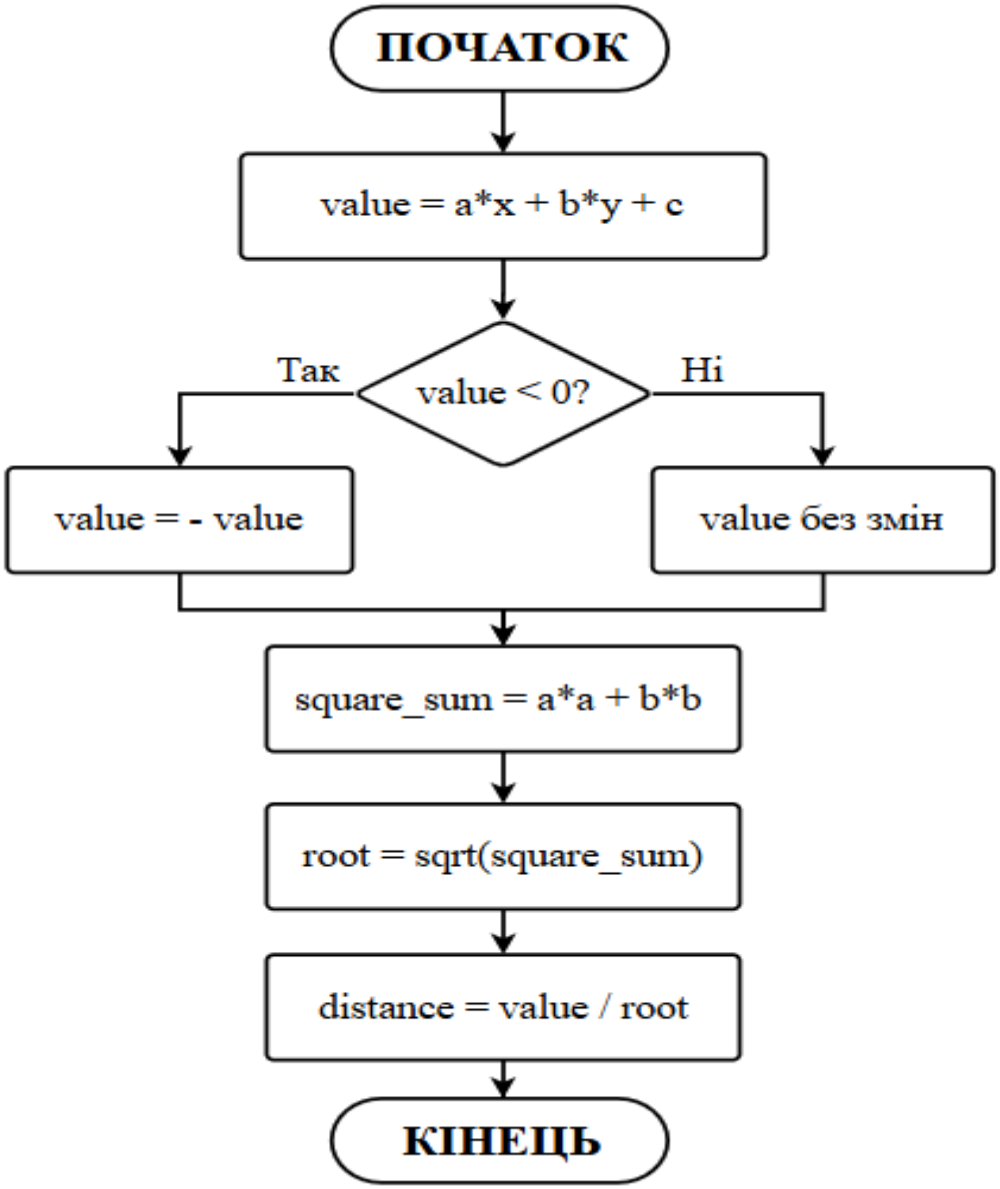
з лабораторної роботи № 7
з дисципліни «Основи програмування»
на тему:
«Створення користувацьких функцій у мові Python»

Виконав: студент гр. Б25_д/F3 (Б)
Сауляк Н. Б.
Перевірив: доц. Бабаков Р. М.

Вінниця – 2025

Завдання 1

№	Завдання
12	На декартовій площині задана пряма a*x+b*y+c=0. Знайти відстань від точки M до цієї прямої.



Функція `distance_to_line(x, y, a, b, c)` приймає п’ять аргументів:
Координати точки: (x, y) – координати точки, від якої ми шукаємо відстань.
Параметри прямої: (a, b, c) – коефіцієнти, які визначають пряму, задану загальним рівнянням:

$$ax + by + c = 0$$

Формула відстані D від точки (x_0, y_0) до прямої $ax + by + c = 0$ має такий вигляд:

$$D = \frac{|ax_0 + by_0 + c|}{\sqrt{a^2 + b^2}}$$

Обчислення виразу чисельника (value):

```
value = a*x + b*y + c
```

На цьому кроці обчислюється значення чисельника формули. По суті, ми підставляємо координати заданої точки (x, y) у ліву частину рівняння прямої. Отримане значення (value) може бути додатним, від'ємним або нулем.

Застосування модуля до чисельника (value):

```
if value < 0:
    value = -value
```

Відстань завжди є невід'ємною величиною. Тому ми застосовуємо операцію модуля (абсолютного значення) до обчисленого на попередньому кроці value. Якщо value від'ємне, воно множиться на -1, стаючи додатним.

Обчислення суми квадратів коефіцієнтів (square_sum):

```
square_sum = a*a + b*b
```

Починається обчислення знаменника. Тут ми знаходимо суму квадратів коефіцієнтів a та b прямої: $a^2 + b^2$.

Обчислення квадратного кореня (root):

```
root = square_sum ** 0.5
```

Обчислення фінальної відстані (distance):

```
distance = value / root
```

Виконується ділення модуля чисельника (крок 2) на квадратний корінь знаменника (крок 4). Це дає кінцеве значення перпендикулярної відстані D.

Повернення результату:

```
return distance
```

Лістинг програми:

```
def distance_to_line(x, y, a, b, c):


    # крок 1: обчислюємо вираз
    value = a*x + b*y + c
```

```
    # крок 2: модуль
    if value < 0:
        value = -value
    # крок 3: обчислюємо a^2 + b^2
    square_sum = a*a + b*b
    # крок 4: корінь квадратний
    root = square_sum ** 0.5
    # крок 5: ділимо
    distance = value / root
    # крок 6: повертаємо результат
    return distance

# приклади виклику:
print(distance_to_line(3, 4, 2, -1, -1))
print(distance_to_line(0, 2, 1, 2, -3))
print(distance_to_line(x=1, y=5, a=3, b=4, c=-10))
```

Виведення результату:

```
0.4472135954999579
0.4472135954999579
2.6
```

Завдання 2

Функція 1:

1. Отримати координати точки A і B.
2. Перевірити, чи дорівнюють обидві координати нулю.Якщо так, то точка є центром квадрата, і алгоритм завершується.
3. Якщо хоча б одна координата дорівнює нулю, то точка лежить на одній із осей координат і не може бути кутом квадрата.
4. Якщо модулі координат рівні між собою (|A| = |B|), то точка є кутом квадрата з центром у (0,0).
5. Якщо жодна з попередніх перевірок не виконана, то точка не є кутом квадрата.
6. Вивести відповідне повідомлення про результат.

Лістинг програми:

```
def check_square_corner(A, B):
    # Перевірка 1: центр квадрата
    if A == 0 and B == 0:
        return "Це центр квадрата, а не кут."

    # Перевірка 2: точка лежить на осі
    if A == 0 or B == 0:
        return "Точка лежить на осі координат і не є кутом
квадрата."

    # Перевірка 3: умова кута квадрата |A| = |B|
    if abs(A) == abs(B):
        return "Так, ця точка є кутом квадрата з центром у
(0, 0)."

    # Інший випадок
    return "Ні, ця точка не є кутом квадрата."
```

```
# приклади виклику
print(check_square_corner(3, 3))
print(check_square_corner(0, 5))
print(check_square_corner(2, -3))
```

Виведення:

```
Так, ця точка є кутом квадрата з центром у (0, 0).
Точка лежить на осі координат і не є кутом квадрата.
Ні, ця точка не є кутом квадрата.
```

Функція 2:

1. Отримати п’ять цілих чисел: x1, x2, x3, x4, x5.
2. Створити змінну-лічильник повторів і присвоїти їй значення 0.
3. Перевірити, чи повторюється число x1 серед інших чотирьох чисел. Якщо так, збільшити лічильник на 1.
4. Перевірити, чи повторюється число x2 серед чисел x3, x4, x5. Якщо так, збільшити лічильник на 1.
5. Аналогічно перевірити, чи повторюється число x3 серед x4 та x5. Якщо так, збільшити лічильник на 1.
6. Перевірити, чи дорівнює число x4 числу x5. Якщо так, збільшити лічильник на 1.
7. Після завершення перевірок вивести значення лічильника. Це число показує, скільки значень зустрічаються більше одного разу.

Лістинг програми:

```
def count_repeated(x1, x2, x3, x4, x5):
    count = 0

    # Перевіряємо, чи x1 повторюється
    if x1 == x2 or x1 == x3 or x1 == x4 or x1 == x5:
        count += 1
```

```
    # Для x2
    if x2 == x3 or x2 == x4 or x2 == x5:
        count += 1

    # Для x3
    if x3 == x4 or x3 == x5:
        count += 1

    # Для x4
    if x4 == x5:
        count += 1

    return count



# приклади виклику
print(count_repeated(1, 2, 3, 2, 5))
print(count_repeated(7, 7, 7, 7, 7))
print(count_repeated(1, 2, 3, 4, 5))
```

Виведення:

```
1
4
0
```

Функція 3:

1. Отримати 10 цифр, які складають номер телефону: 5 цифр числа A та 5 цифр числа B.
2. Об’єднати всі цифри в один рядок, сформувавши повний номер телефону.
3. Визначити оператора мобільного зв’язку за першими трьома цифрами:

- Якщо номер починається з 050, 095 або 099 → оператор Vodafone.
- Якщо номер починається з 067, 068, 096, 097 або 098 → оператор Kyivstar.
- Якщо номер починається з 063, 073 або 093 → оператор Lifecell.
- Якщо жоден з варіантів не підходить → оператор невідомий.

4. Перевірити «елітність» номера. Номер вважається елітним, якщо він містить будь-яку з комбінацій:
   000, 111, 222, 333, 444, 555, 666, 777, 888, 999.
5. Якщо хоча б одна така послідовність є в номері, позначити номер як елітний. Інакше номер вважається звичайним.
6. Вивести сформований номер, визначеного оператора та результат перевірки на елітність.

Лістинг програми:

```
def phone_info(a1, a2, a3, a4, a5, b1, b2, b3, b4, b5):
    # Формуємо номер
    number = f"{a1}{a2}{a3}{a4}{a5}{b1}{b2}{b3}{b4}{b5}"

    # Визначення оператора
    operator = ""
    if number.startswith("050") or number.startswith("095")
or number.startswith("099"):
        operator = "Vodafone"

    if number.startswith("067") or number.startswith("068")
or number.startswith("096") or number.startswith("097") or
number.startswith("098"):
        operator = "Kyivstar"

    if number.startswith("063") or number.startswith("073")
or number.startswith("093"):
```

```
        operator = "Lifecell"

    # Перевірка елітності
    elite = False
    if "000" in number or "111" in number or "222" in number
or \
       "333" in number or "444" in number or "555" in number
or \
       "666" in number or "777" in number or "888" in number
or \
       "999" in number:
        elite = True

    # Результат
    return number, operator, elite



# приклади виклику
print(phone_info(0,5,0,1,2, 3,4,5,6,7))
print(phone_info(0,6,7,1,2, 3,3,3,4,4))
print(phone_info(0,9,9,1,1, 1,1,1,1,1))
```

Виведення:

```
('0501234567', 'Vodafone', False)
('0671233344', 'Kyivstar', True)
('0991111111', 'Vodafone', True)
```

Висновок

У ході виконання роботи було розроблено три окремі функції, кожна з яких реалізує індивідуальне завдання відповідно до варіанта. Під час виконання завдання було закріплено навички роботи з умовними операторами, логічними виразами, обробкою даних та побудовою простих алгоритмів.