Міністерство освіти і науки України
Центральноукраїнський національний технічний університет
Механіко-технологічний факультет
Кафедра кібербезпеки та програмного забезпечення
Дисципліна: Скриптові мови програмування (Python)

Лабораторна робота №2
Тема: «ФУНКЦІЇ У PYTHON»

Виконав: ст. гр. КН-24
Куріщенко П. В.
Перевірив: ассистент
Ткаченко О.С.

Кропивницький 2024

Варіант - 2

Мета роботи - навчитися створювати власні функції у мові Python.

ЗАВДАННЯ ДО ЛАБОРАТОРНОЇ РОБОТИ

Розробити наступні функції:

1. Функцію, що приймає 1 аргумент — сторону квадрата, і повертає 3 значення: периметр квадрата, площу квадрата та діагональ квадрата;

Опис принципу роботи / проектних рішень функції `squareParams():`

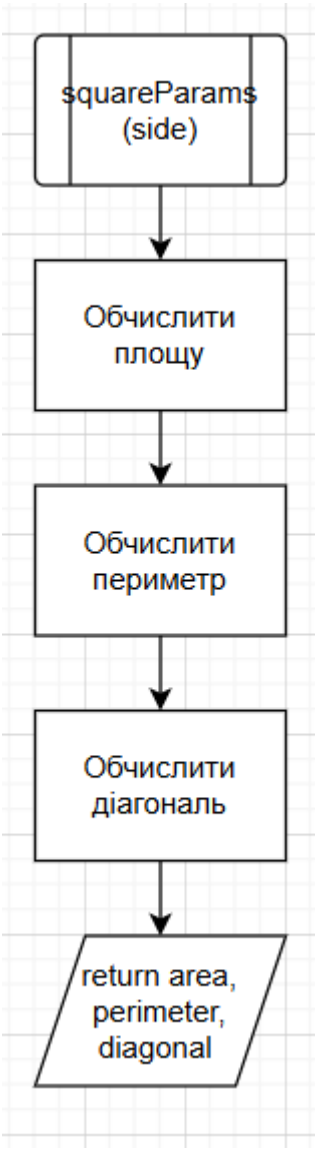
Призначення:

Обчислює площу, периметр і довжину діагоналі квадрата за заданою довжиною сторони.

Принцип роботи / проектне рішення:

- Площа знаходиться як **сторона²**.
- Периметр — це **4 × сторона**.
- Діагональ обчислюється за теоремою Піфагора: **сторона × √2**.
- Використовується модуль math для обчислення квадратного кореня.
- Повертається кортеж із трьома значеннями.

Блок-схема:



Результат запуску:

```
Enter the length of the side of the square: 12
Area: 144.0, Perimeter: 48.0, Diagonal: 16.970562748477143
```

2. Функцію, яка приймає 1 аргумент — число від 0 до 1000, і повертає True, якщо воно просте, і False — інакше;

Опис принципу роботи / проектних рішень функції `isNumberPrime()`:

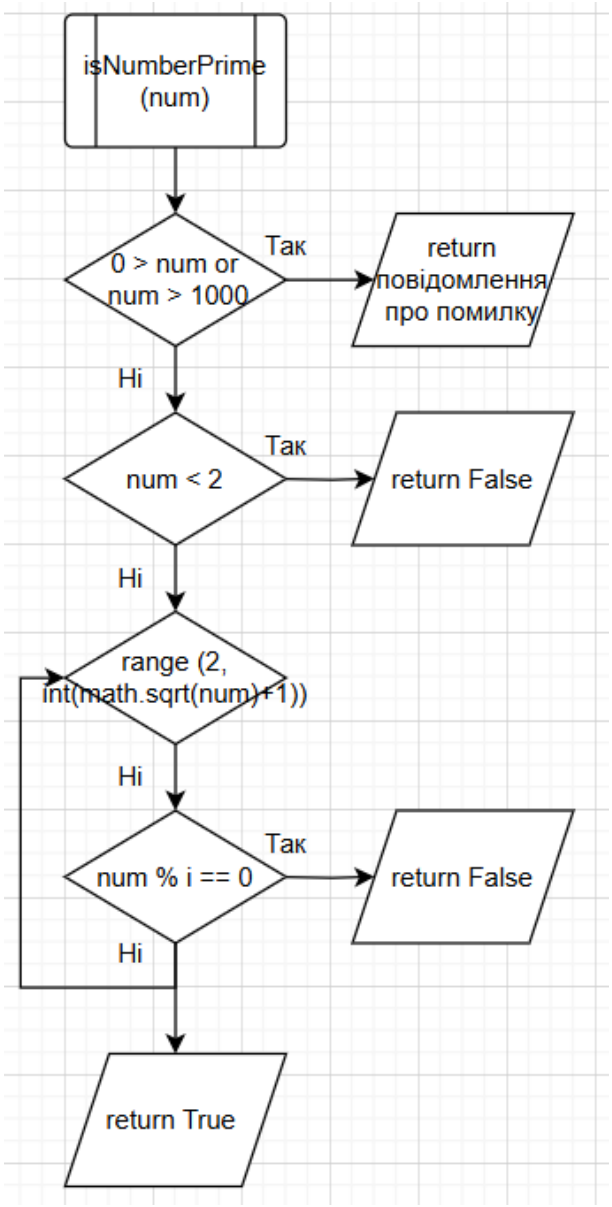
Призначення:

Перевіряє, чи є число простим.

Принцип роботи / проектне рішення:

- Якщо число менше 0 або більше 1000 — виводиться повідомлення про обмеження.
- Просте число визначається як число більше 1, яке ділиться тільки на 1 і на себе.
- Перевірка ділиться від **2 до √num**, що оптимізує перевірку.
- Повертає `True` для простих чисел, `False` — для непростих.

Блок-схема:



Результат запуску:

```
Enter a number between 0 and 1000: 67
Is your number prime? True
```

```
Enter a number between 0 and 1000: -10
Is your number prime? Please enter a number between 0 and 1000
```

```
Enter a number between 0 and 1000: 111
Is your number prime? False
```

3. Функцію, що як аргумент приймає ціле число N, створює список довжиною N, заповнює його випадковими цілими числами та

повертає цей список. Примітка: Для генерації випадкових чисел використати бібліотеку random та її функцію randint;

Опис принципу роботи / проектних рішень функції `randomList()`:

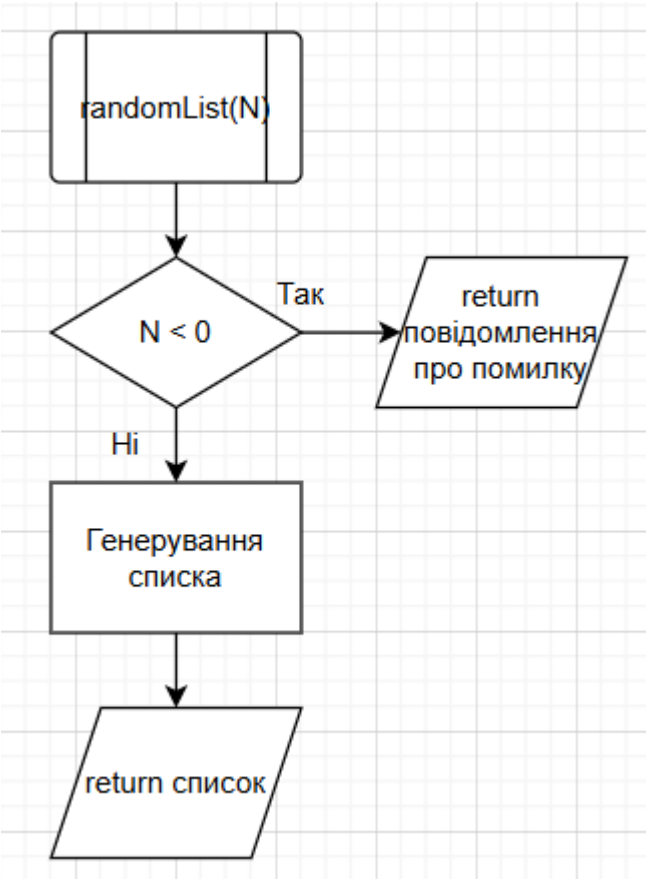
Призначення:

Генерує список з **N випадкових цілих чисел** в діапазоні від -100 до 100.

Принцип роботи / проектне рішення:

- Перевіряється, чи N додатне — якщо ні, повертається повідомлення про помилку.
- Інакше створюється список за допомогою генератора списку та random.randint().
- Використовується модуль random для генерації чисел.

Блок-схема:



Результат запуску:

```
Enter the number of elements in the list: 6
Your random list: [-15, 64, -82, 88, 97, -63]
```

```
Enter the number of elements in the list: -1
Your random list: Please enter a whole number greater than 0
```

4. Функцію, що отримує 2 аргументи — список чисел lst та число n і повертає всі числа зі списку lst, що більші числа n;

Опис принципу роботи / проектних рішень функції `listOfBiggerNumbers()`:

Призначення:

Фільтрує елементи списку, які **більші за задане число num**.

Принцип роботи / проектне рішення:

- Якщо список порожній — виводиться повідомлення про помилку.
- Інакше використовується генератор списку з умовою int(i) > num.
- Елементи попередньо приводяться до int, щоб уникнути помилок при порівнянні.

Блок-схема:

Результат запуску:

```
Enter a number: 8
Enter the numbers in the list (space-separated): 1 -29 9 233 8 9 10 4 0
All numbers in your list that bigger than your number: [9, 233, 9, 10]
```

```
Enter a number: 1000
Enter the numbers in the list (space-separated): 2 3 320 999 21 -1001
All numbers in your list that bigger than your number: []
```

```
Enter a number: 12
Enter the numbers in the list (space-separated):
All numbers in your list that bigger than your number: Please enter a list of numbers
```

5. Функцію, що отримує 2 аргументи — список чисел lst та число x і повертає значення, скільки разів число x зустрічається у списку lst;

Опис принципу роботи / проектних рішень функції `amountOfSameNumbers(num, lst):`

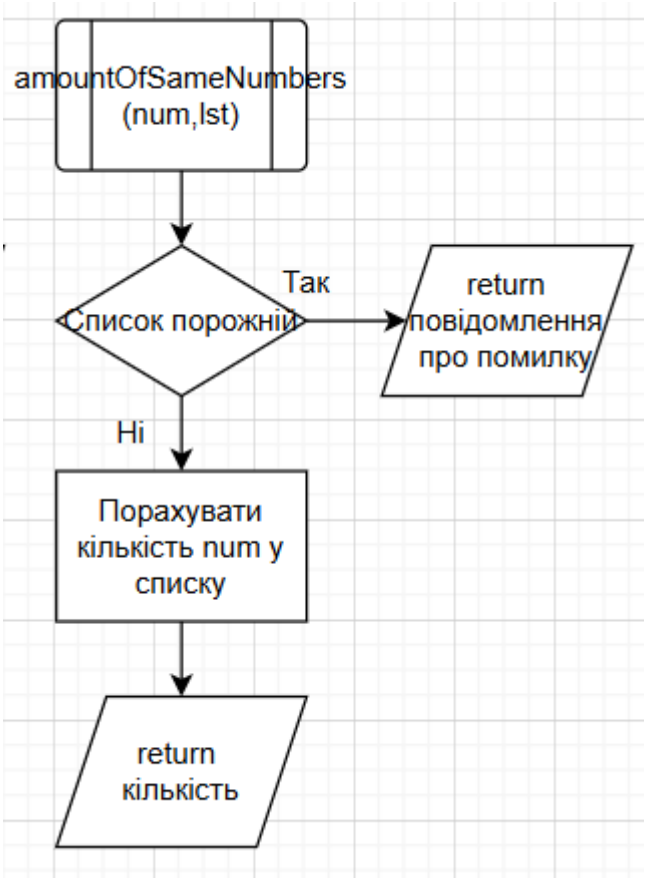
Призначення:

Підраховує, скільки разів задане число num зустрічається у списку lst.

Принцип роботи / проектне рішення:

- Якщо список порожній — виводиться повідомлення про помилку.
- Інакше використовується вбудований метод count() для обчислення кількості входжень.

Блок-схема:



Результат запуску:

```
Enter a number: 1
Enter the numbers in the list (space-separated): 1 1 1 1 1 21 11 -1 0
Amount of same numbers in your list: 5
```

```
Enter a number: -12
Enter the numbers in the list (space-separated):
Amount of same numbers in your list: Please enter a list of numbers
```

```
Enter a number: 128238
Enter the numbers in the list (space-separated): 3 30 21 309 12
Amount of same numbers in your list: 0
```

6. Функцію, що як аргументи отримує 2 списки, а повертає один список, що містить усі спільні елементи списків-аргументів;

Опис принципу роботи / проектних рішень функції `listOfRepeatedNumbers(lst1, lst2):`

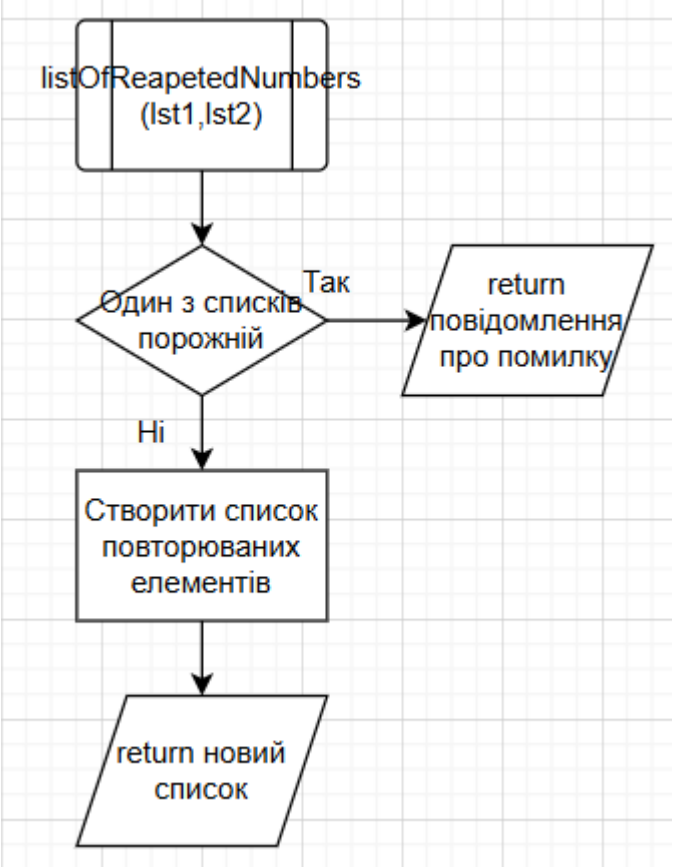
Призначення:

Знаходить **спільні елементи** в обох списках.

Принцип роботи / проектне рішення:

- Перевіряється наявність обох списків.
- Генератор списку проходить по lst1, залишаючи лише ті елементи, які є в lst2.
- Зберігає порядок елементів з lst1.

Блок-схема:



Результат запуску:

```
Enter the numbers in the first list (space-separated): 2 203 76 32 -1010 30 6 38
Enter the numbers in the second list (space-separated): 6 34 75 -1010 6 1 0
New list with all elements that repeat: [-1010, 6]
```

```
Enter the numbers in the first list (space-separated): 1 2 3 4 5 6 7 8
Enter the numbers in the second list (space-separated): 0 0 0 0 0 0 0 0 0 0
New list with all elements that repeat: []
```

7. Функцію, яка приймає як аргументи 2 списки та виводить усі елементи першого списку, яких немає у другому.

Опис принципу роботи / проектних рішень функції `listOfUniqueElements(lst1, lst2):`

Призначення:

Створює список з елементів першого списку, **які не зустрічаються** в другому.

Принцип роботи / проектне рішення:

- Перевіряється, чи обидва списки не порожні.
- Генератор списку виключає елементи, які є в lst2.
- Це дозволяє дізнатися унікальні елементи лише з lst1.

Блок-схема:

Результат запуску:

```
Enter the numbers in the first list (space-separated): 2 2 2 29 32 -10
Enter the numbers in the second list (space-separated): 8 9 10 2 23 29
New list with all elements of the first list that don't appear in the second list: [32, -10]
```

```
Enter the numbers in the first list (space-separated): 3 2 3 -2
Enter the numbers in the second list (space-separated):
New list with all elements of the first list that don't appear in the second list: Please enter two lists of numbers
```

```
Enter the numbers in the first list (space-separated): 2 38 291 53 -37 -67 -73
Enter the numbers in the second list (space-separated): 647 -3 27 1000 23
New list with all elements of the first list that don't appear in the second list: [2, 38, 291, 53, -37, -67, -73]
```

Помістіть функції в окремий модуль. Реалізуйте програму, яка використовує всі функції зі створеного модуля. Зробіть описи `Doc strings` для кожної реалізованої функції.

Лістинг розробленого модуля `utils.py`:

```
import math
import random


def squareParams(side):
```

```
    """
    Calculate the area, perimeter, and diagonal of a square.

    Args:
        side (float): The length of the side of the square.

    Returns:
        tuple: A tuple containing the area, perimeter, and diagonal.
    """
    area = side ** 2
    perimeter = 4 * side
    diagonal = math.sqrt(2) * side
    return area, perimeter, diagonal

def isNumberPrime(num):
    """
    Check if a number is prime.

    Args:
        num (int): The number to check.

    Returns:
        bool or str: True if the number is prime, False otherwise.
                     Returns a string if the number is out of range.
    """
    if num < 0 or num > 1000:
        return "Please enter a number between 0 and 1000"
    elif num < 2:
        return False
    for i in range(2, int(math.sqrt(num)) + 1):
        if num % i == 0:
            return False
    return True

def randomList(N):
    """
    Generate a list of N random integers between -100 and 100.
```

```
    Args:
        N (int): The number of elements in the list.

    Returns:
        list: A list of N random integers.
    """
    if N < 0:
        return "Please enter a whole number greater than 0"
    else:
        return [random.randint(-100, 100) for _ in range(N)]

def listOfBiggerNumbers(num, lst):
    """
    Create a new list with all elements from the input list that are
greater than the given number.

    Args:
        num (int): The number to compare against.
        lst (list): The list of numbers to filter.

    Returns:
        list: A new list containing elements greater than num.
    """
    if len(lst) == 0:
        return "Please enter a list of numbers"
    else:
        return [int(i) for i in lst if int(i) > num]

def amountOfSameNumbers(num, lst):
    """
    Count the occurrences of a number in a list.

    Args:
        num (int): The number to count.
        lst (list): The list of numbers.
```

```
    Returns:
        int: The count of occurrences of num in lst.
    """
    if len(lst) == 0:
        return "Please enter a list of numbers"
    else:
        return lst.count(num)

def listOfRepeatedNumbers(lst1, lst2):
    """
    Create a new list with all elements that are present in both
input lists.

    Args:
        lst1 (list): The first list of numbers.
        lst2 (list): The second list of numbers.

    Returns:
        list: A new list containing elements that are present in
both lst1 and lst2.
    """
    if len(lst1) == 0 or len(lst2) == 0:
        return "Please enter two lists of numbers"
    else:
        return [i for i in lst1 if i in lst2]

def listOfUniqueElements(lst1, lst2):
    """
    Create a new list with all elements of the first list that don't
appear in the second list.

    Args:
        lst1 (list): The first list of numbers.
        lst2 (list): The second list of numbers.

    Returns:
```

```
        list: A new list containing elements from lst1 that are not
in lst2.
    """
    if len(lst1) == 0 or len(lst2) == 0:
        return "Please enter two lists of numbers"
    else:
        return [i for i in lst1 if i not in lst2]
```

Лістинг `main.py`:

```
from utils import squareParams, isNumberPrime, randomList,
listOfBiggerNumbers, amountOfSameNumbers, listOfRepeatedNumbers,
listOfUniqueElements

def main():

    # first function
    print("Area: {0}, Perimeter: {1}, Diagonal:
{2}".format(*squareParams(float(input("Enter the length of the side
of the square: ")))))

    # second function
    print("Is your number prime?",
        isNumberPrime(int(input("Enter a number between 0 and 1000:
"))))

    # third function
    print("Your random list:",
        randomList(int(input("Enter the number of elements in the
list: "))))

    # fourth function
    print("All numbers in your list that bigger than your number:",
      listOfBiggerNumbers(
          int(input("Enter a number: ")),
          list(map(int, input("Enter the numbers in the list (space-
separated): ").split())))
```

```
      ))

    # fifth function
    print("Amount of same numbers in your list:",
       amountOfSameNumbers(
           int(input("Enter a number: ")),
           list(map(int, input("Enter the numbers in the list
(space-separated): ").split()))
       ))

    # sixth function
    print("New list with all elements that repeat:",
          listOfRepeatedNumbers(
              list(map(int, input("Enter the numbers in the first
list (space-separated): ").split())),
              list(map(int, input("Enter the numbers in the second
list (space-separated): ").split()))
          ))

    # seventh function
    print("New list with all elements of the first list that don't
appear in the second list:",
          listOfUniqueElements(
              list(map(int, input("Enter the numbers in the first
list (space-separated): ").split())),
              list(map(int, input("Enter the numbers in the second
list (space-separated): ").split()))
          ))

if __name__ == "__main__":
    main()
```

Контрольні питання:

1. Як створити функцію у мові Python?

Функція створюється за допомогою інструкції def, після якої вказується ім’я функції, список аргументів у дужках, та двокрапка. В тілі функції може бути присутня інструкція return, яка повертає результат.

Приклад:

```
def add(x, y):
    return x + y
```

Функції можуть приймати:

- позиційні та іменовані аргументи;
- необов’язкові аргументи з параметрами за замовчуванням;
- змінну кількість позиційних аргументів (*args);
- змінну кількість іменованих аргументів (**kwargs).

2. Що таке модулі та пакети?

Модуль — це файл з розширенням .py, який містить визначення функцій, змінних та класів. Його можна підключати до інших програм за допомогою import.

Пакет — це каталог з набором модулів і файлом __init__.py, що дозволяє Python розпізнати цей каталог як пакет. Пакети дозволяють організовувати модулі в логічну структуру.

3. Які бувають способи підключення модулів та пакетів?

- **Звичайне імпортування:**

```
import math
```

- **Імпортування з псевдонімом:**

```
import math as m
```

- **Імпортування окремих атрибутів модуля:**

```
from math import sqrt, pi
```

- **Імпортування з псевдонімами:**

```
from math import ceil as c
```

- **Імпортування всіх атрибутів модуля:**

```
from math import *
```

4. Що таке анонімні функції та інструкція lambda?

Анонімні функції — це функції, що не мають імені. Вони створюються за допомогою інструкції lambda і використовуються для коротких обчислень, зазвичай "на льоту".

Синтаксис:

```
lambda аргументи: вираз
```

Приклад:

```
f = lambda x: x**2 + 4
# те саме що:
def f(x):
    return x**2 + 4
```

5. Чи можна використати модуль як самостійну програму?

Так, модуль можна використовувати як самостійну програму. Щоб код виконувався лише при безпосередньому запуску, а не при імпорті, слід перевіряти значення змінної __name__:

Приклад:

```
if __name__ == "__main__":
     print("Програма запущена напряму")
```

При імпорті значення `__name__` буде дорівнювати імені модуля, а при прямому запуску — `"__main__"`.