Звіт з практичної роботи No10

Тема роботи: Рекурсивні виклики

Виконав: Кравченко Сергій, ТЦР-12

Дата: 02.06.2025

Мета роботи:

Розробити набір рекурсивних функцій на мові Python для розв'язання

різних задач, розуміючи основні принципи рекурсії, включаючи базовий

випадок та рекурсивний виклик.

Хід роботи:

Постановка завдання:

Завдання 1

1. Рекурсивна функція для обчислення факторіалу:

Код програми:

def factorial\_recursive(n):

"""

Обчислює факторіал числа n рекурсивним способом.

Параметри:

n (int): Ціле невід'ємне число, для якого обчислюється факторіал.

Повертає:

int: Факторіал числа n.

Викидає:

ValueError: Якщо n є від'ємним числом.

Приклади:

>>> factorial\_recursive(5)

120

>>> factorial\_recursive(0)

1

"""

if not isinstance(n, int):

raise TypeError("n має бути цілим числом")

if n < 0:

raise ValueError("Факторіал визначений лише для невід'ємних чисел")

if n == 0:

return 1

return n \* factorial\_recursive(n - 1)

Завдання 2

Рекурсивна функція для обчислення n-го числа Фібоначчі:

Код програми:

def fibonacci\_recursive(n):

"""

Обчислює n-те число Фібоначчі рекурсивним способом.

Параметри:

n (int): Ціле невід'ємне число - порядковий номер числа Фібоначчі.

Повертає:

int: n-те число Фібоначчі.

Викидає:

ValueError: Якщо n є від'ємним числом.

Приклади:

>>> fibonacci\_recursive(7)

13

>>> fibonacci\_recursive(0)

0

"""

if not isinstance(n, int):

raise TypeError("n має бути цілим числом")

if n < 0:

raise ValueError("Число Фібоначчі визначене лише для невід'ємних індексів")

if n == 0:

return 0

if n == 1:

return 1

return fibonacci\_recursive(n - 1) + fibonacci\_recursive(n - 2)

Завдання 3

Рекурсивна функція для знаходження суми елементів списку

Код програми:

def sum\_list\_recursive(lst):

"""

Обчислює суму елементів списку рекурсивним способом.

Параметри:

lst (list): Список чисел для обчислення суми.

Повертає:

int або float: Сума елементів списку.

Приклади:

>>> sum\_list\_recursive([1, 2, 3, 4])

10

>>> sum\_list\_recursive([])

0

"""

if not isinstance(lst, list):

raise TypeError("Вхідний аргумент має бути списком")

if not lst:

return 0

return lst[0] + sum\_list\_recursive(lst[1:])

Завдання 4

Рекурсивна функція для перевірки, чи є рядок паліндромом

Код програми:

def is\_palindrome\_recursive(s):

"""

Перевіряє, чи є заданий рядок паліндромом (читається однаково зліва направо та справа наліво).

Параметри:

s (str): Рядок для перевірки.

Повертає:

bool: True, якщо рядок є паліндромом, False - якщо ні.

Приклади:

>>> is\_palindrome\_recursive("А роза упала на лапу Азора")

True

>>> is\_palindrome\_recursive("Python")

False

"""

if not isinstance(s, str):

raise TypeError("Вхідний аргумент має бути рядком")

# Функція для очищення рядка

def clean\_string(s):

s = s.lower()

return ''.join(c for c in s if c.isalnum())

cleaned = clean\_string(s)

# Базові випадки

if len(cleaned) <= 1:

return True

# Рекурсивний крок

if cleaned[0] != cleaned[-1]:

return False

return is\_palindrome\_recursive(cleaned[1:-1])