Практическая работа № 5

Использование функций языка Python и работа с массивами

Цель работы — мы должны научиться создавать и использовать функции в Python, а также работать с массивами и операциями над ними

**Задание 1: Создание функций**

**1. Написали функцию для вычисления факториала числа**

# Функция для вычисления факториала числа

def factorial(n):

if n == 0 or n == 1:

return 1

else:

return n \* factorial(n - 1)

# Пример использования функции

number = 5

print(f"Факториал числа {number} равен {factorial(number)}")

Решение: Факториал числа 5 равен 120

**2. Создали функцию для нахождения максимального и минимального элемента в списке**

# Функция для нахождения максимального и минимального элемента в списке

def find (numbers):

min\_val = min(numbers)

max\_val = max(numbers)

return min\_val, max\_val

# Пример использования функции

lst = [3, 7, 1, 9, 4]

min\_val, max\_val = find (lst)

print(f"Мин. Знач = {min\_val}, Макс.Знач = {max\_val}")

Решение: Мин. значение: 1, Макс. значение: 9

**Задание 2: Работа с аргументами функций**

**1. Реализовали функцию с переменным количеством аргументов**

# Функция с переменным количеством аргументов

def sum\_all(\*args):

total = 0

for num in args:

total += num

return total

# Пример использования функции

result = sum\_all(1, 2, 3, 4, 5)

print(f"Сумма всех аргументов: {result}")

Решение: Сумма всех аргументов: 15

**2. Написали функцию с аргументами по умолчанию**

# Функция с аргументами по умолчанию

def greet(name, greeting="Hi"):

return f"{greeting}, {name}!"

# Пример использования функции

print(greet("Dredd")) # Используется приветствие по умолчанию

print(greet("Bob", "hi")) # Переопределяем приветствие

**Задание 3: Работа с массивами**

**1. Создали массив и выполните операции добавления, удаления и поиска элементов**

numbers = [1, 2, 3, 4, 5]

# Добавление элемента

numbers.append(6)

print(f"Массив после добавления элемента: {numbers}")

# Удаление элемента

numbers.remove(3)

print(f"Массив после удаления элемента: {numbers}")

# Поиск элемента

search\_val = 4

if search\_val in numbers:

print(f"Элемент {search\_value} найден в массиве.")

else:

print(f"Элемент {search\_value} не найден в массиве.")

Решение:

* Массив после добавления элемента: [1, 2, 3, 4, 5, 6]
* Массив после удаления элемента: [1, 2, 4, 5, 6]
* Элемент 4 найден в массиве.

**2. Реализовали программу для вычисления среднего значения элементов массива**

# Функция для вычисления среднего значения элементов массива

def average(numbers):

total = sum(numbers)

count = len(numbers)

return total / count

# Пример использования функции

lst = [10, 20, 30, 40, 50]

average\_val = average(lst)

print(f"Среднее значение элементов массива: {average\_val}")

Решение: среднее значение элементов массива: 30.0

**Задание 4: Многомерные массивы**

**1. Написали программу для работы с матрицами**

# Создание матрицы

matrix = [

[1, 2, 3],

[4, 5, 6],

[7, 8, 9]

]

# Вывод матрицы

print("Матрица:")

for row in matrix:

print(row)

**2. Реализовали сложение и умножение матриц**

# Функция для сложения двух матриц

def add\_matrices(matrix1, matrix2):

result = []

for i in range(len(matrix1)):

row = []

for j in range(len(matrix1[0])):

row.append(matrix1[i][j] + matrix2[i][j])

result.append(row)

return result

# Функция для умножения двух матриц

def multiply\_matrices(matrix1, matrix2):

result = []

for i in range(len(matrix1)):

row = []

for j in range(len(matrix2[0])):

sum = 0

for k in range(len(matrix2)):

sum += matrix1[i][k] \* matrix2[k][j]

row.append(sum)

result.append(row)

return result

# Пример использования функций

matrix1 = [

[1, 2, 3],

[4, 5, 6],

[7, 8, 9]

]

matrix2 = [

[9, 8, 7],

[6, 5, 4],

[3, 2, 1]

]

# Сложение матриц

sum\_matrix = add\_matrices(matrix1, matrix2)

print("Сумма матриц:")

for row in sum\_matrix:

print(row)

Решение: Сумма матриц:

[10, 10, 10]

[10, 10, 10]

[10, 10, 10]

# Умножение матриц

product\_matrix = multiply\_matrices(matrix1, matrix2)

print("Произведение матриц:")

for row in product\_matrix:

print(row)

Решение: Произведение матриц:

[30, 24, 18]

[84, 69, 54]

[138, 114, 90]

**Задание 5: Рекурсивные функции**

**1. Написали рекурсивную функцию для вычисления чисел Фибоначчи**

# Рекурсивная функция для вычисления чисел Фибоначчи

def fibonacci(n):

if n <= 1:

return n

else:

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

# Пример использования функции

n = 10

print(f"Число Фибоначчи для n = {n} равно {fibonacci(n)}")

Решение: Число для n = 10 равно 55

**2. Реализовали функцию для вычисления суммы цифр числа рекурсивно**

# Рекурсивная функция для вычисления суммы цифр числа

def sum\_of\_digits(n):

if n == 0:

return 0

else:

return (n % 10) + sum\_of\_digits(n // 10)

# Пример использования функции

number = 12345

print(f"Сумма цифр числа {number} равна {sum\_of\_digits(number)}")

Решение: Сумма цифр числа 12345 равна 15