

Министерство науки и высшего образования Российской Федерации  
Федеральное государственное автономное образовательное учреждение  
высшего образования  
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития  
Кафедра инфокоммуникаций

**ОТЧЕТ**  
**ПО ЛАБОРАТОРНОЙ РАБОТЕ №2.23**  
**дисциплины «Анализ данных»**  
**Вариант 13**

Выполнил:  
Иващенко Олег Андреевич  
2 курс, группа ИВТ-б-о-22-1,  
09.03.02 «Информационные и  
вычислительные машины»,  
направленность (профиль)  
«Программное обеспечение  
средств вычислительной техники  
и автоматизированных систем»

---

(подпись)

Руководитель практики:  
Воронкин Роман Александрович,  
доцент кафедры  
инфокоммуникаций

---

(подпись)

Отчет защищен с оценкой \_\_\_\_\_ Дата защиты \_\_\_\_\_

Ставрополь, 2024 г.

**Тема:** «Управление потоками в Python»

**Цель:** Приобрести навыки написания многопоточных приложений на языке программирования Python версии 3.x.

### Порядок выполнения работы

Индивидуальное задание: С использованием многопоточности для заданного значения  $x$  найти сумму  $S$  ряда с точностью члена ряда по абсолютному значению  $E = 10^{-7}$  и произвести сравнение полученной суммы с контрольным значением функции для двух бесконечных рядов.

$$13. \quad S = \sum_{n=1}^{\infty} \frac{1}{(2n-1)x^{2n-1}} = \frac{1}{x} + \frac{1}{3x^3} + \frac{1}{5x^5} + \dots; \quad x = 3; \quad y = \frac{1}{2} \ln \frac{x+1}{x-1}.$$

Рисунок 1 – Исходная формула

### Листинг 1 – Код individual.py

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

import math
import threading

"""
Задача:
С использованием многопоточности для заданного значения найти
сумму ряда с точностью члена ряда по абсолютному значению и
произвести сравнение полученной суммы с контрольным значением функции
для двух бесконечных рядов.
"""

e = 10e-7
stepArray = [1]
lock = threading.Lock()

def calculateY(x):
    return 0.5 * math.log((x + 1) / (x - 1))

def calculate_step(step, index, x, n):
    step[index] = 1

    def firstStep():
        step[index] *= (2 * n - 1)

    def secondStep():
        step[index] *= x ** (2 * n - 1)

    def thirdStep():
```

```

step[index] **= -1

firstThread = threading.Thread(target=firstStep)
secondThread = threading.Thread(target=secondStep)
thirdThread = threading.Thread(target=thirdStep)

firstThread.start()
secondThread.start()
thirdThread.start()

firstThread.join()
secondThread.join()
thirdThread.join()

def main():
    x = 3
    index = 0

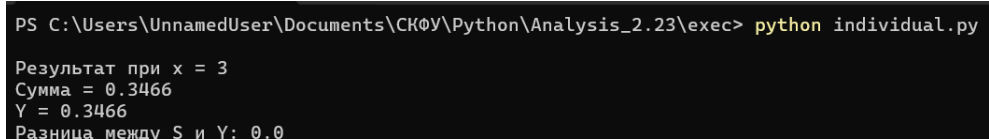
    while abs(stepArray[index]) > e:
        stepArray.append(0)
        calculate_step(stepArray, index + 1, x, index + 1)
        index += 1

    S = sum(stepArray) - 1
    y = calculateY(x)

    print(f"\nРезультат при x = {x}")
    print(f"Сумма = {round(S, 4)}")
    print(f"Y = {round(y, 4)}")
    print(f"Разница между S и Y: {abs(round(S - y, 4))}\n")

if __name__ == "__main__":
    main()

```



```

PS C:\Users\UnnamedUser\Documents\CKФУ\Python\Analysis_2.23\exec> python individual.py

Результат при x = 3
Сумма = 0.3466
Y = 0.3466
Разница между S и Y: 0.0

```

Рисунок 2 – Результат выполнения программы

### Контрольные вопросы

1. Что такое синхронность и асинхронность?

Синхронность - выполнение задач поочередно, в определенном порядке.

Асинхронность - выполнение задач независимо от других, без блокировки.

2. Что такое параллелизм и конкурентность?

Параллелизм - выполнение нескольких задач одновременно.

Конкурентность - планирование выполнения задач в разное время, но они могут выполняться параллельно или чередоваться.

3. Что такое GIL? Какое ограничение накладывает GIL?

GIL (Global Interpreter Lock) - механизм в Python, позволяющий только одному потоку выполнять байт-код Python в любой момент времени, ограничивая параллелизм.

4. Каково назначение класса Thread?

Класс Thread - используется для создания и управления потоками в Python.

5. Как реализовать в одном потоке ожидание завершения другого потока?

Ожидание завершения другого потока - с помощью метода `join()`, который блокирует выполнение текущего потока до завершения указанного потока.

6. Как проверить факт выполнения потоком некоторой работы?

Проверка факта выполнения работы потоком - с использованием флагов, обмена сообщениями или метода `is_alive()` для проверки состояния потока.

7. Как реализовать приостановку выполнения потока на некоторый промежуток времени?

Приостановка выполнения потока - с помощью метода `time.sleep(seconds)` для приостановки выполнения потока на определенное количество секунд.

#### 8. Как реализовать принудительное завершение потока?

Принудительное завершение потока - обычно не рекомендуется, потому что может привести к некорректной работе программы. Можно использовать флаги или другие механизмы для безопасного завершения потока.

#### 9. Что такое потоки-демоны? Как создать поток-демон?

Потоки-демоны - это потоки, которые выполняются в фоновом режиме и завершаются автоматически, когда все другие непосредственные потоки завершают свою работу. Создать поток-демон можно, установив атрибут `daemon` объекта потока в `True` до его запуска.

**Выводы:** В процессе выполнения лабораторной работы были приобретены навыки написания многопоточных приложений на языке программирования Python, а также было выполнено индивидуальное задание.