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 №10
дисциплины «Анализ данных»
Вариант 29

Выполнил:
Саенко Андрей Максимович
2 курс, группа ИВТ-б-о-22-1,
09.03.01 «Информатика и вычислительная техника»,
направленность (профиль) «Программное обеспечение средств вычислительной техники и автоматизированных систем», очная форма обучения

_____
(подпись)

Руководитель практики:
Воронкин Р.А., канд. технических наук, доцент кафедры инфокоммуникаций

_____
(подпись)

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2024 г.

Тема: Синхронизация потоков в языке программирования Python

Цель: приобретение навыков использования примитивов синхронизации в языке программирования Python версии 3.x.

Порядок выполнения работы:

Вариант 29

Задание 1. Для своего индивидуального задания лабораторной работы 2.23 необходимо организовать конвейер, в котором сначала в отдельном потоке вычисляется значение первой функции, после чего результаты вычисления должны передаваться второй функции, вычисляемой в отдельном потоке. Потоки для вычисления значений двух функций должны запускаться одновременно.

Код программы:

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

# С использованием многопоточности для
# заданного значения x найти сумму ряда S с
# точностью члена ряда по абсолютному
# значению e=10^-7 и произвести сравнение
# полученной суммы
# с контрольным значением функции y
# для двух бесконечных рядов.
# Варианты 29 и 30 (4 и 5)

import math
from threading import Lock, Thread, Barrier

lock = Lock()
br = Barrier(3)

# 4 (29) вариант
def sum_row_1(x, eps, s_dict):
    s = 0
    n = 1
    while True:
        a = 1/(2 ** n)
        b = 1/(3 ** n)
        c = math.pow(x, n-1)
        element = (a + b)*c

        if abs(element) < eps:
            break
        else:
```

```
            s += element
            n += 1
    with lock:
        s_dict["row_1"] = s
    br.wait()

# 5 (30) вариант
def sum_row_2(x, eps, sum_dict):
    sum = 0
    n = 0
    f = 1
    i = 1
    while True:
        z = 2*i
        k = x**(2*n)
        f *= (z-1)*z
        element = ((-1) ** n) * k / (f/2)
        i = n+1

        if abs(element) < eps:
            break
        else:
            sum += element
            n += 1
    with lock:
        sum_dict["row_2"] = sum
    br.wait()

def conveyor(sum_dict, y1, y2):
    br.wait()
    sum_1 = sum_dict["row_1"]
    sum_2 = sum_dict["row_2"]
    print("Функция conveyor(): \n")
    print(
        f"Полученное значение (4 вариант): {sum_1}"
        f"\nОжидаемое значение (4 вариант): {y1}"
        f"\nРазница: {abs(sum_1 - y1)}"
    )
    print(
        f"\nПолученное значение (5 вариант): {sum_2}"
        f"\nОжидаемое значение(5 вариант)): {y2}"
        f"\nРазница: {abs(sum_2 - y2)}"
    )

def main():
    sum_dict = {}

    eps = 10**-7
    # 4 Вариант
    x1 = -0.8
    y1 = (5-2*x1) / (6-5*x1+(x1**2))
    # 5 Вариант
```

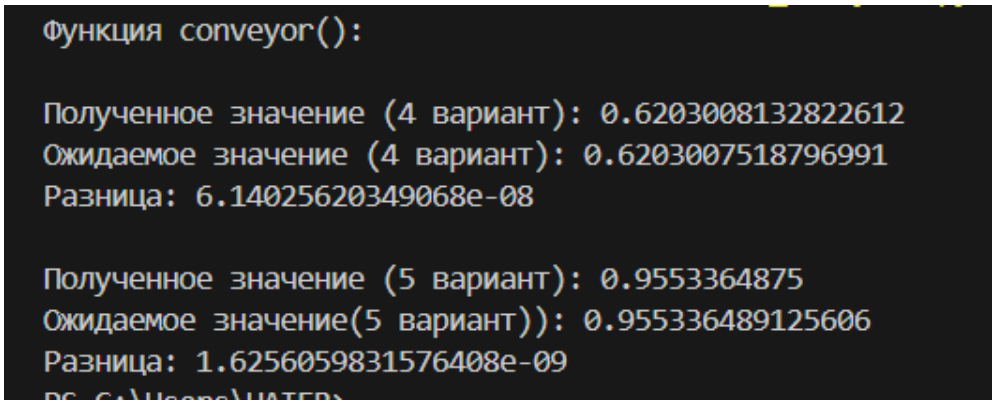
```
    x2 = 0.3
    y2 = math.cos(x2)

    thread1 = Thread(target=sum_row_1, args=(x1, eps, sum_dict))
    thread2 = Thread(target=sum_row_2, args=(x2, eps, sum_dict))
    thread3 = Thread(target=conveyor, args=(sum_dict, y1, y2))

    thread1.start()
    thread2.start()
    thread3.start()

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

Результат работы программы:



Выводы: в ходе выполнения работы получены навыки работы с потоками и средствами для их синхронизации.