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 №9
дисциплины «Анализ данных»

Выполнила:
Кубанова Ксения Олеговна
2 курс, группа ИВТ-б-о-22-1,
09.03.01 «Информатика и вычислительная техника», очная форма обучения

_____
(подпись)

Руководитель практики:
Воронкин Р. А.

_____
(подпись)

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2024 г.

Тема: управление потоками

Цель: приобретение навыков написания многопоточных приложений на языке программирования Python версии 3.x.

Порядок выполнения работы

Индивидуальное задание.

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.

Номера варианта:

16. $$S = \sum_{n=0}^{\infty} (-1)^n \frac{x^{2n}}{n!} = 1 - \frac{x^2}{1!} + \frac{x^4}{2!} - \frac{x^6}{3!} + \ldots;\ x = -0,7;\ y = \exp(-x^2).$$

```
th1 = threading.Thread(target=lambda: results.append(calc_chis(x, i)))
th2 = threading.Thread(target=lambda: results.append(calc_znam(i)))

th1.start()
th2.start()

th1.join()
th2.join()
```

Рисунок 1 – реализация многопоточности в коде

Для реализации многопоточности для заданного значения был использован модуль threading. С его помощью были созданы потоки th1, th2, в которые передавались соответственно функции (сначала для вычисления числителя, после для вычисления знаменателя) с соответствующими аргументами. Помимо этого в создании потоков был использован lambda, необходимый для добавления результатов в список, чтобы после их обработать.

```
x = -0.7
Ожидаемое значение y = 0.6126263941844161
Подсчитанное значение = 0.51
```

Рисунок 2 – результат выполнения кода

Полноценный код располагается в ind.py.

Контрольные вопросы

1 Что такое синхронность и асинхронность?

Синхронность означает последовательное выполнение задач: каждая следующая задача начинается только после завершения предыдущей.

Асинхронность подразумевает параллельное выполнение задач: выполнение следующей задачи может начаться до завершения предыдущей.

2 Что такое параллелизм и конкурентность?

Параллелизм означает одновременное выполнение нескольких задач на разных вычислительных ресурсах, таких как ядра процессора или разные машины.

Конкурентность подразумевает одновременное выполнение нескольких задач в одном вычислительном ресурсе, например, на одном ядре процессора.

3 Что такое GIL? Какое ограничение накладывает GIL?

GIL - это механизм внутри интерпретатора Python, который предназначен для обеспечения безопасности потоков в многопоточной среде.

GIL накладывает ограничение на использование только одного потока Python в одно время. Это означает, что в любой момент времени только один поток может выполняться внутри интерпретатора Python, даже если у вас есть несколько ядер процессора.

4 Каково назначение класса Thread ?

Класс Thread в Python используется для создания потоков выполнения.

Он предоставляет удобный способ создания и управления потоками в многопоточном приложении.

5 Как реализовать в одном потоке ожидание завершения другого потока?

Для ожидания завершения другого потока можно использовать метод join(). Вызов thread.join() заставляет текущий поток ждать завершения потока thread.

6 Как проверить факт выполнения потоком некоторой работы?

Для проверки выполнения работы потоком можно использовать флаги или переменные, которые будут изменяться при выполнении определенной работы потоком.

7 Как реализовать приостановку выполнения потока на некоторый промежуток времени?

Для приостановки выполнения потока на некоторый промежуток времени можно использовать функцию time.sleep(secs), где secs - количество секунд, на которое нужно приостановить выполнение потока.

8 Как реализовать принудительное завершение потока?

В стандартной библиотеке Python нет прямой поддержки принудительного завершения потока. Однако можно использовать флаги или переменные для обозначения того, что поток должен завершиться, и проверять эту переменную внутри цикла выполнения потока.

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

Потоки-демоны (daemon threads) - это потоки, которые работают в фоновом режиме и завершаются автоматически, когда завершается основной поток программы.

Для создания потока-демона в Python используется атрибут daemon, который устанавливается в True перед запуском потока.

Вывод: в ходе выполнения работы были приобретены навыки написания многопоточных приложений на языке программирования Python версии 3.x.