МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
ИНСТИТУТ ЦИФРОВОГО РАЗВИТИЯ

Отчет по лабораторной работе № 11
«Управление потоками в Python»

по дисциплине «Основы программной инженерии»

Выполнила:
Первых Дарья Александровна,
2 курс, группа ПИЖ-б-о-20-1

Проверил:
Доцент кафедры
инфокоммуникаций, Воронкин Р.А.

Ставрополь, 2022 г

ВЫПОЛНЕНИЕ

```
def func():
    for i in range(5):
        print(f"from child thread: {i}")
        sleep(0.5)


if __name__ == '__main__':
    th = Thread(target=func)
    th.start()
```

```
main
C:\Users\podar\study\anaconda\envs\LR11\pythor
from child thread: 0
from main thread: 0
from main thread: 1
from main thread: 2
from main thread: 3
from main thread: 4
from child thread: 1
from child thread: 2
from child thread: 3
from child thread: 4
```

Рисунок 1 – Создание и ожидание завершения работы потоков

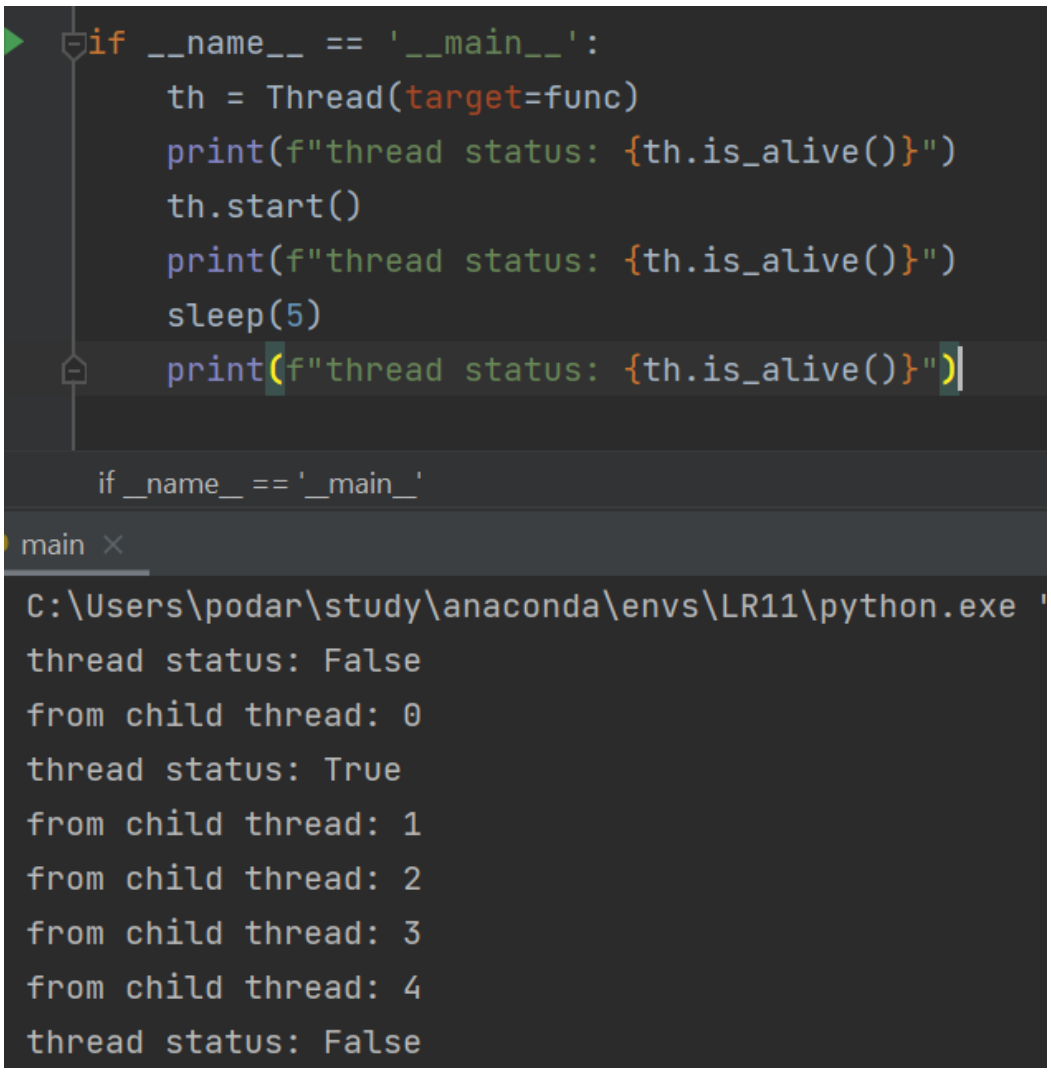

Рисунок 2 – Метод is_alive()

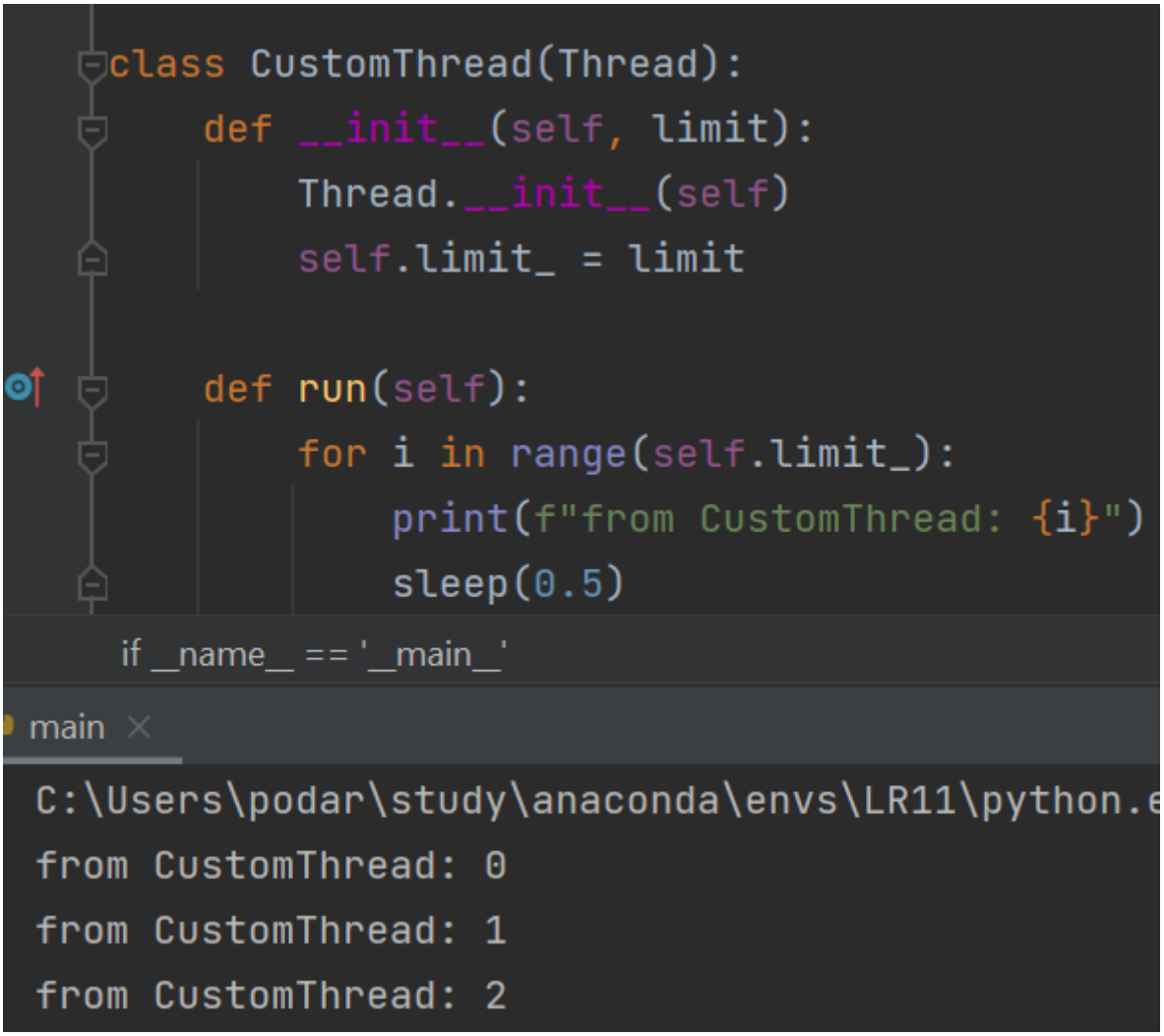

Рисунок 3 – Создание классов наследников от Thread

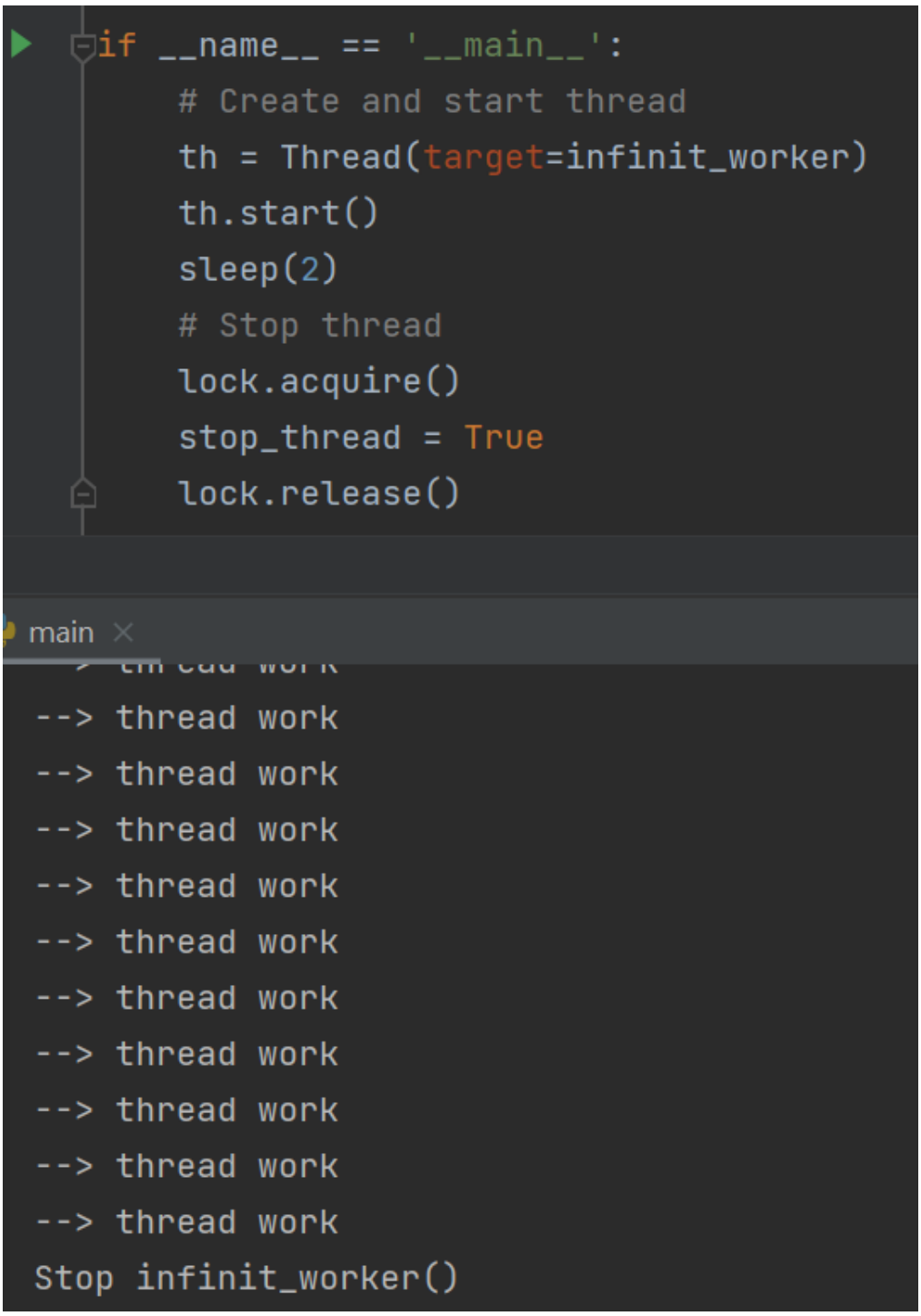

Рисунок 4 – Принудительное завершение работы потока

Рисунок 5 – Потоки-демоны

Индивидуальное задание

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

Вариант 17

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

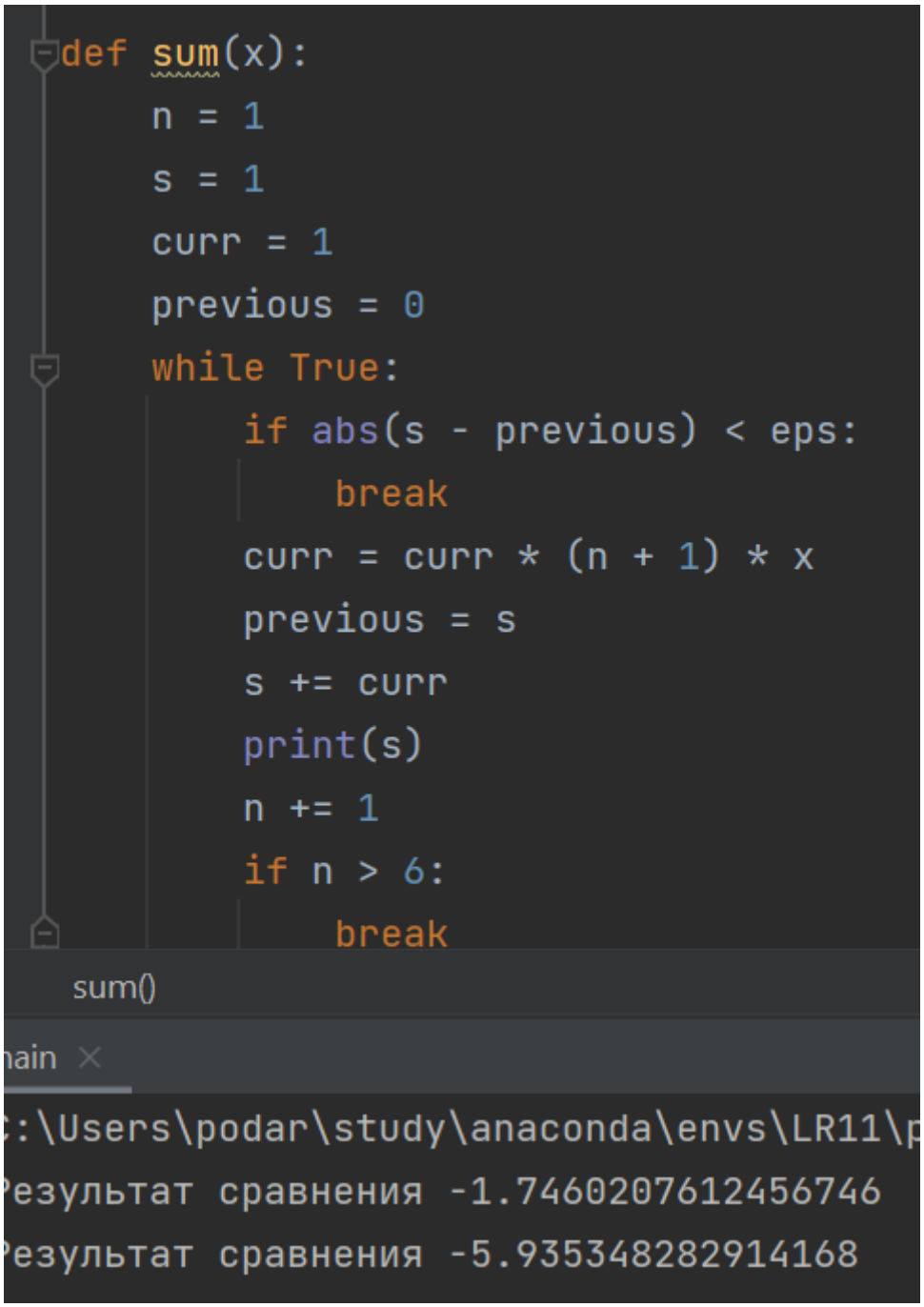

Рисунок 6 – Индивидуальное задание

ВОПРОСЫ

1. Что такое синхронность и асинхронность?

Синхронное выполнение программы подразумевает последовательное выполнение операций. Асинхронное – предполагает возможность независимого выполнения задач.

2. Что такое параллелизм и конкурентность?

Конкурентность предполагает выполнение нескольких задач одним

исполнителем. Параллельность предполагает параллельное выполнение задач разными исполнителями.

3. Что такое GIL? Какое ограничение накладывает GIL?

GIL — это аббревиатура от Global Interpreter Lock – глобальная блокировка интерпретатора. Он является элементом эталонной реализации языка Python, которая носит название CPython. Суть GIL заключается в том, что выполнять байт код может только один поток. Это нужно для того, чтобы упростить работу с памятью (на уровне интерпретатора) и сделать комфортной разработку модулей на языке C. Это приводит к некоторым особенностям, о которых необходимо помнить. Условно, все задачи можно разделить на две большие группы: в первую входят те, что преимущественно используют процессор для своего выполнения, например, математические, их ещё называют CPU-bound, во вторую – задачи работающие с вводом выводом (диск, сеть и т.п.), такие задачи называют IO-bound. Если вы запустили в одном интерпретаторе несколько потоков, которые в основном используют процессор, то скорее всего получите общее замедление работы, а не прирост производительности. Пока выполняется одна задача, остальные простаивают (из-за GIL), переключение происходит через определенные промежутки времени. Таким образом, в каждый конкретный момент времени, будет выполняться только один поток несмотря на то, что у вас может быть многоядерный процессор (или многопроцессорный сервер), плюс ко всему, будет тратиться время на переключение между задачами. Если код в потоках в основном выполняет операции ввода-вывода, то в этом случае ситуация будет в вашу пользу. В CPython все стандартные библиотечные функций, которые выполняют блокирующий ввод-вывод, освобождают GIL, это дает возможность поработать другим потокам, пока ожидается ответ от ОС.

4. Каково назначение класса Thread?

За создание, управление и мониторинг потоков отвечает класс Thread из модуля threading. Поток можно создать на базе функции, либо реализовать свой класс – наследник Thread и переопределить в нем метод run().

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

Если необходимо дождаться завершения работы потока(-ов) перед тем как начать выполнять какую-то другую работу, то воспользуйтесь методом join().У join() есть параметр timeout, через который задается время ожидания завершения работы потоков.

6. Как проверить факт выполнения потоком некоторой работы?

Для того, чтобы определить выполняет ли поток какую-то работу или завершился используется метод is_alive().

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

С помощью метода sleep() из модуля time.

8. Как реализовать принудительное завершение потока?

В Python у объектов класса Thread нет методов для принудительного завершения работы потока. Один из вариантов решения этой задачи – это создать специальный флаг, через который потоку будет передаваться сигнал остановки. Доступ к такому флагу должен управляться объектом синхронизации.

lock.acquire()

if stop_thread is True:

break

lock.release()

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

Для того, чтобы потоки не мешали остановке приложения (т.е. чтобы они останавливались вместе с завершением работы программы) необходимо при создании объекта Thread аргументу daemon присвоить значение True, либо после создания потока, перед его запуском присвоить свойству deamon значение True.

th = Thread(target=func, daemon=True)