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 «Основы кроссплатформенного программирования»

Выполнил:
Евдаков Евгений Владимирович
2 курс, группа ИТС-б-о-22-1,
11.03.02 «Инфокоммуникационные технологии и системы связи»,
направленность (профиль) «Инфокоммуникационные системы и сети», очная форма обучения

_____
(подпись)

Руководитель практики:
Воронкин Р. А., доцент кафедры инфокоммуникаций

_____
(подпись)

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2023 г.

Тема: Управление потоками в Python

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

Ход работы:

Задание 1. Создал общедоступный репозиторий на GitHub, в котором использована лицензий MIT и язык программирования Python, также добавил файл .gitignore с необходимыми правилами. Клонировал свой репозиторий на свой компьютер. Организовал свой репозиторий в соответствие с моделью ветвления git-flow, появилась новая ветка develop в которой буду выполнять дальнейшие задачи.

```
C:\Users\Gaming-PC>git clone https://github.com/EvgenyEvdakov/Laba_2.23.git
Cloning into 'Laba_2.23'...
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (5/5), done.
```

Рисунок 1. Клонирование репозитория

Задание 2. Создал виртуальное окружение conda и активировал его, также установил необходимые пакеты isort, black, flake8.

```
(base) PS C:\Users\Gaming-PC> cd C:\Users\Gaming-PC\Laba_2.23
(base) PS C:\Users\Gaming-PC\Laba_2.23> conda create -n 2.23 python=3.10
Retrieving notices: ...working... done
Collecting package metadata (current_repodata.json): done
Solving environment: done


==> WARNING: A newer version of conda exists. <==
  current version: 23.1.0
  latest version: 23.10.0

Please update conda by running

    $ conda update -n base -c defaults conda

Or to minimize the number of packages updated during conda update use

     conda install conda=23.10.0
```

Рисунок 2. Создание виртуального окружения

Задание 3. Создал проект PyCharm в папке репозитория. Приступил к работе с примером. Добавил новый файл primer1.py.

Условие примера: необходимо импортировать нужные модули. После этого объявить функцию func(), которая выводит пять раз сообщение с числовым маркером с задержкой в 500 мс. Далее создать объект класса Thread, в нем, через параметр target, указать, какую функцию запускать как поток и запустить его. В главном потоке добавить код вывода сообщений с интервалом в 1000 мс.

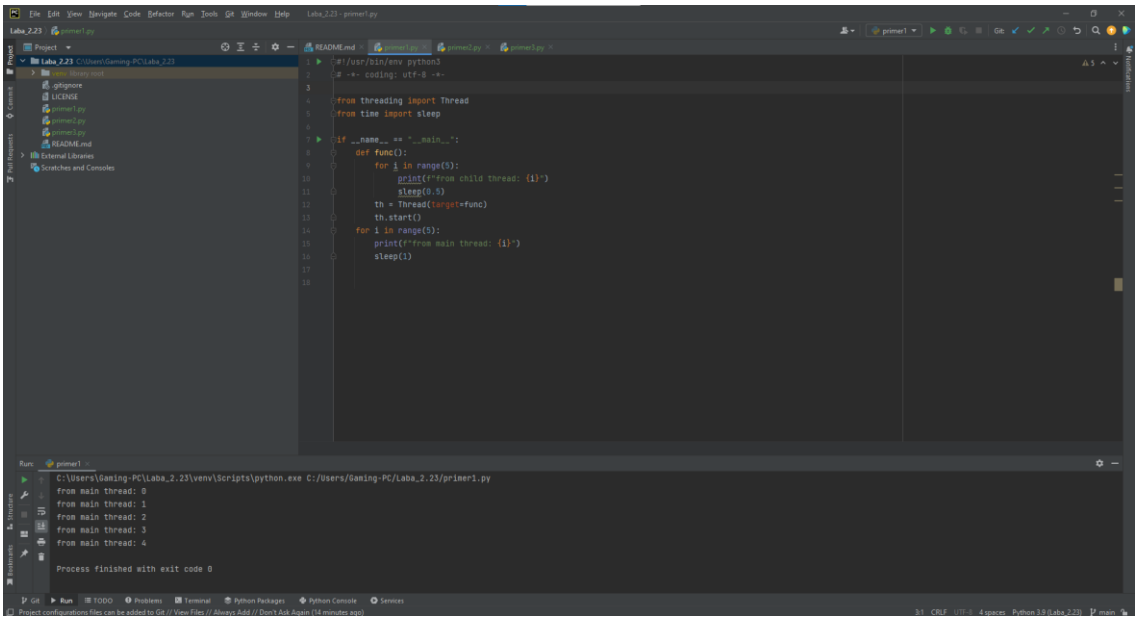

Рисунок 3. Выполнение первого примера

Добавил новый файл primer2.py.

Условие примера: 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.

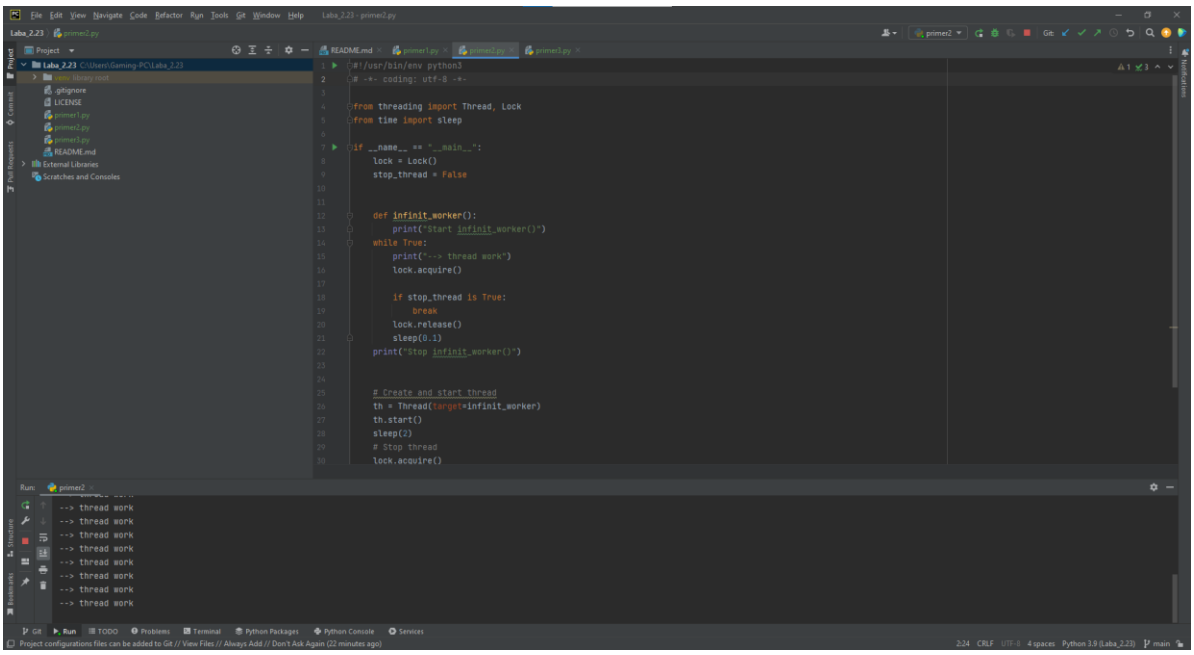

Рисунок 4. Выполнение второго примера

Разберемся с этим кодом более подробно. В строке 4 мы создаем объект класса Lock, он используется для синхронизации доступа к ресурсам из нескольких потоков. В нашем случае, ресурс — это переменная stop_thread, объявленная в строке 6, которая используется как сигнал для остановки потока. После этого, в строке 8, объявляется функция infinit_worker(), ее мы запустим как поток. В ней выполняется бесконечный цикл, каждый проход которого отмечается выводом в терминал сообщения “–> thread work” и проверкой состояния переменной stop_thread. В главном потоке программы создается и запускается дочерний поток (строки 24, 25), выполняется функция задержки и принудительно завершается поток путем установки переменной stop_thread значения True.

Добавил новый файл primer3.py.

Условие примера: есть такая разновидность потоков, которые называются демоны. Python приложение не будет закрыто до тех пор, пока в нем работает хотя бы один недемонический поток. 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.

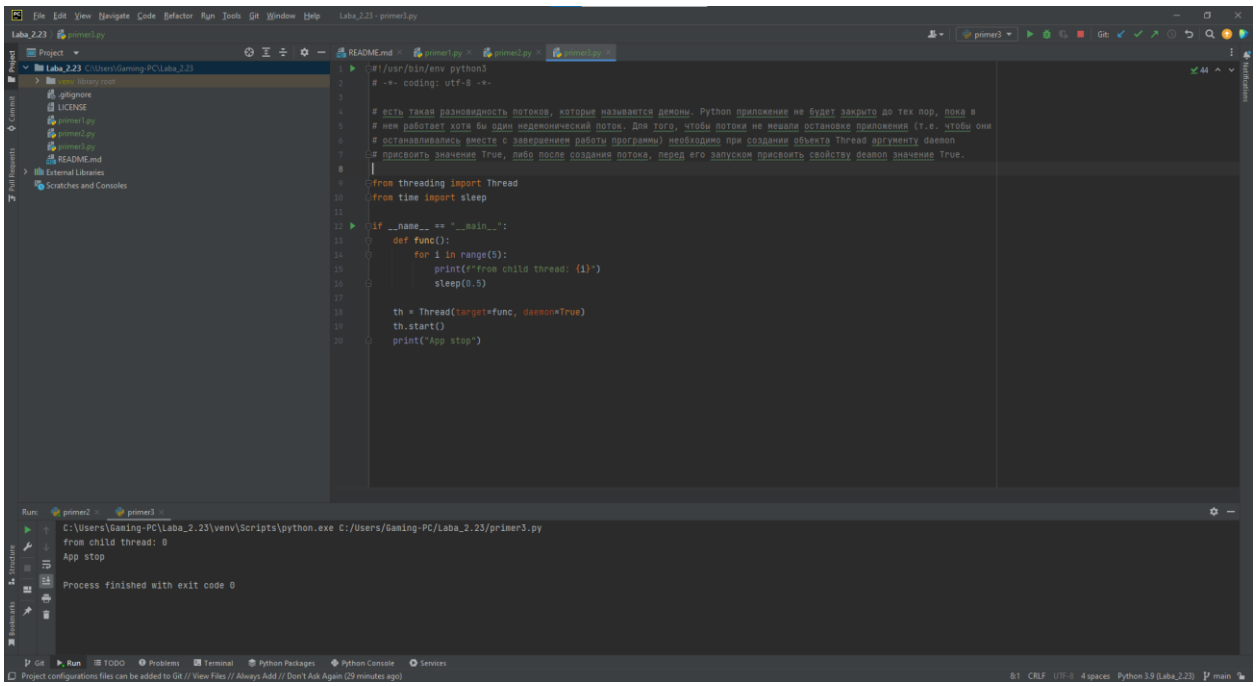

Рисунок 5. Выполнение третьего примера

Задание 4.

Индивидуальное задание

Вариант 10

Создал новый файл под названием idz.py.

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

10. $$S=\sum_{n=0}^{\infty}\frac{x^{2n}}{(2n)!}=1+\frac{x^2}{2!}+\frac{x^4}{4!}+\ldots;\ x=\frac{1}{2};\ y=\frac{e^x+e^{-x}}{2}. \qquad (16)$$

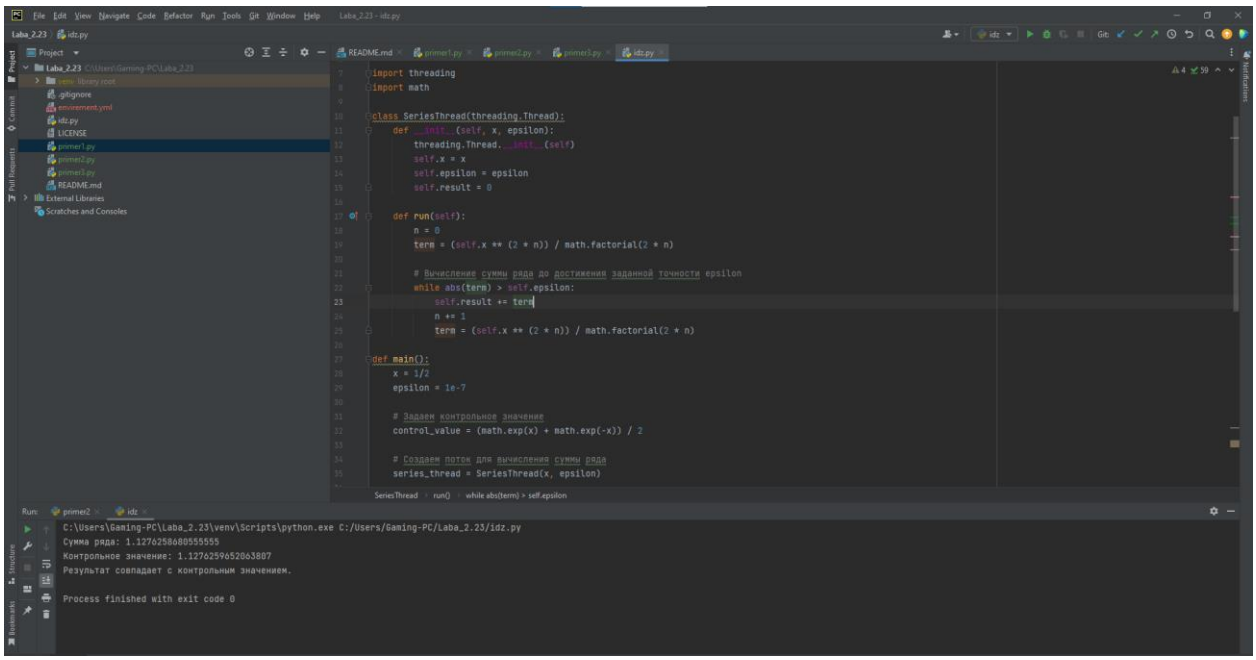

Рисунок 6. Выполнение индивидуального задания

Задание 5.

После выполнения работы на ветке develop, слил ее с веткой main и отправил изменения на удаленный сервер. Создал файл envirement.yml и деактивировал виртуальное окружение.

```
(2.23) PS C:\Users\Gaming-PC\Laba_2.23> conda env export > envirement.yml
(2.23) PS C:\Users\Gaming-PC\Laba_2.23> conda deactivate
```

Рисунок 7. Деактивация ВО

Ссылка: https://github.com/EvgenyEvdakov/Laba_2.23

Ответы на контрольные вопросы:

1. **Что такое синхронность и асинхронность?**

- Синхронность: в синхронном выполнении задачи каждый шаг ожидает завершения предыдущего. То есть код выполняется последовательно, шаг за шагом.

- Асинхронность: в асинхронном выполнении задачи код может продолжать выполнение, не дожидаясь завершения предыдущего шага. Это позволяет эффективнее использовать ресурсы и обрабатывать множество задач одновременно.

2. **Что такое параллелизм и конкурентность?**

- Параллелизм: это выполнение нескольких задач одновременно, фактически в один и тот же момент времени. Каждая задача выполняется независимо от других.
- Конкурентность: это координация выполнения нескольких задач. Задачи могут выполняться в разное время, но между ними существует взаимодействие.

3. **Что такое GIL? Какое ограничение накладывает GIL?**

GIL — это механизм, используемый в некоторых интерпретаторах, например, в CPython (стандартная реализация Python). Он предназначен для обеспечения безопасности в многопоточной среде, но ограничивает возможность использования нескольких ядер процессора для параллельного выполнения Python-кода.

4. **Каково назначение класса Thread?**

Класс Thread в языке программирования Python предоставляет средства для создания и управления потоками выполнения. Потоки представляют собой легковесные процессы, которые выполняются независимо друг от друга.

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

Можно использовать метод join() для ожидания завершения другого потока.

6. **Как проверить факт выполнения потоком некоторой работы?**

Это может зависеть от конкретной реализации, но обычно можно использовать флаги или другие механизмы для сигнализации о выполнении работы.

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

В Python можно использовать time.sleep(seconds) для приостановки выполнения потока на определенное количество секунд.

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

В Python принудительное завершение потока не всегда рекомендуется, но можно использовать флаги или исключения для безопасного завершения.

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

Это потоки, которые выполняются в фоновом режиме и завершаются, когда основной поток завершается. В Python можно создать демонический поток, установив атрибут daemon объекта Thread в True.

Вывод: приобрел навыки написания многопоточных приложений на языке программирования Python версии 3.x.