

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»

Выполнил:

Гайчук Дарья Дмитриевна
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.-доцент кафедры
инфокоммуникаций

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2024 г.

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

Порядок выполнения работы:

1. Создала новый репозиторий и клонировала его на свой компьютер.

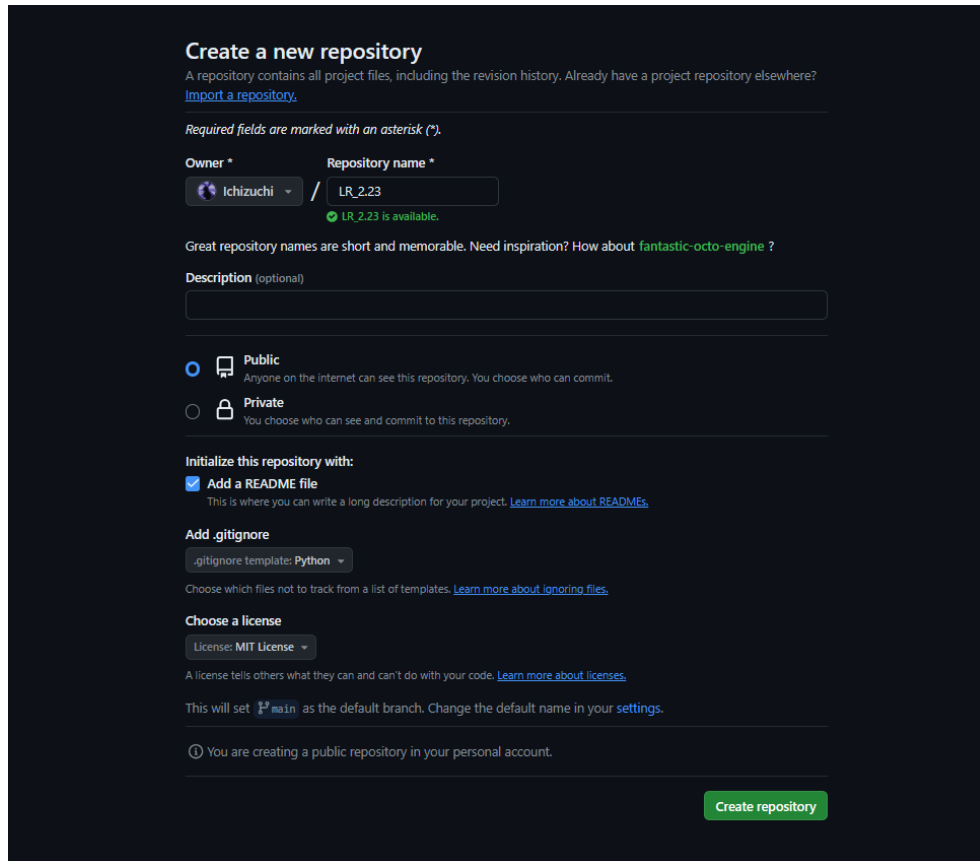


Рисунок 1. Создан новый репозиторий

2. Клонировала репозиторий на свой компьютер.

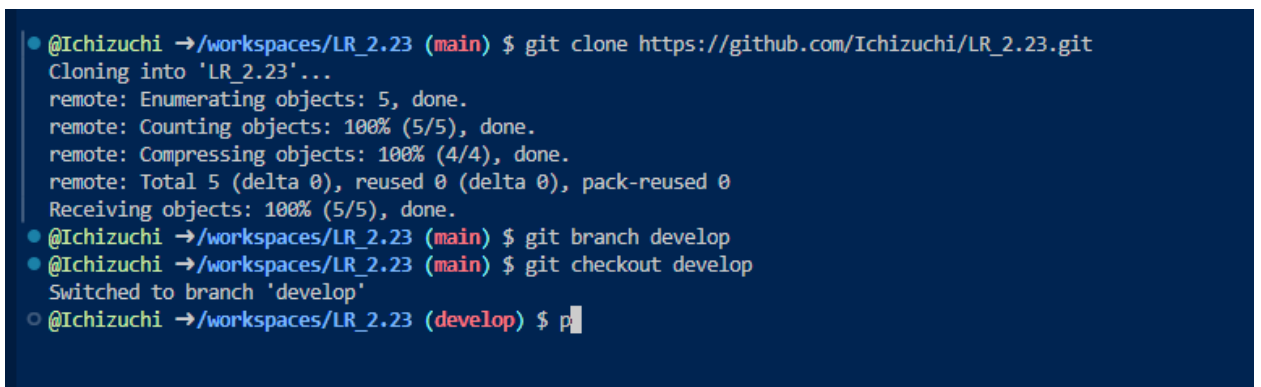


Рисунок 2. Клонирование и модель ветвления git-flow

3. Создала виртуальное окружение Anaconda с именем репозитория.

```
@Ichizuchi →/workspaces/LR_2.23 (develop) $ conda create -n myenv python=3.10
Retrieving notices: ...working... done
Channels:
 - defaults
Platform: linux-64
Collecting package metadata (repodata.json): done
Solving environment: done

## Package Plan ##

environment location: /opt/conda/envs/myenv

added / updated specs:
 - python=3.10

The following packages will be downloaded:
```

package	build	
-----	-----	-----
_libgcc_mutex-0.1	main	3 KB
_openmp_mutex-5.1	1_gnu	21 KB
bzip2-1.0.8	h5eee18b_5	262 KB
ca-certificates-2024.3.11	h06a4308_0	127 KB
ld_impl_linux-64-2.38	h1181459_1	654 KB
libffi-3.4.4	h6a678d5_0	142 KB

Рисунок 3. Создание виртуального окружения

Работа с примерами

```
primer1.py > ...
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  # необходимо импортировать нужные модули. После этого объявить функцию func(), которая выводит пять раз сообщение с
5  # числовым маркером с задержкой в 500 мс. Далее создать объект класса Thread, в нем, через параметр target, указать,
6  # какую функцию запускать как поток и запустить его. В главном потоке добавить код вывода сообщений с интервалом
7  # в 1000 мс.
8
9  from threading import Thread
10 from time import sleep
11
12 if __name__ == "__main__":
13     def func():
14         for i in range(5):
15             print(f"from child thread: {i}")
16             sleep(0.5)
17         th = Thread(target=func)
18         th.start()
19         for i in range(5):
20             print(f"from main thread: {i}")
21             sleep(1)
22
23
```

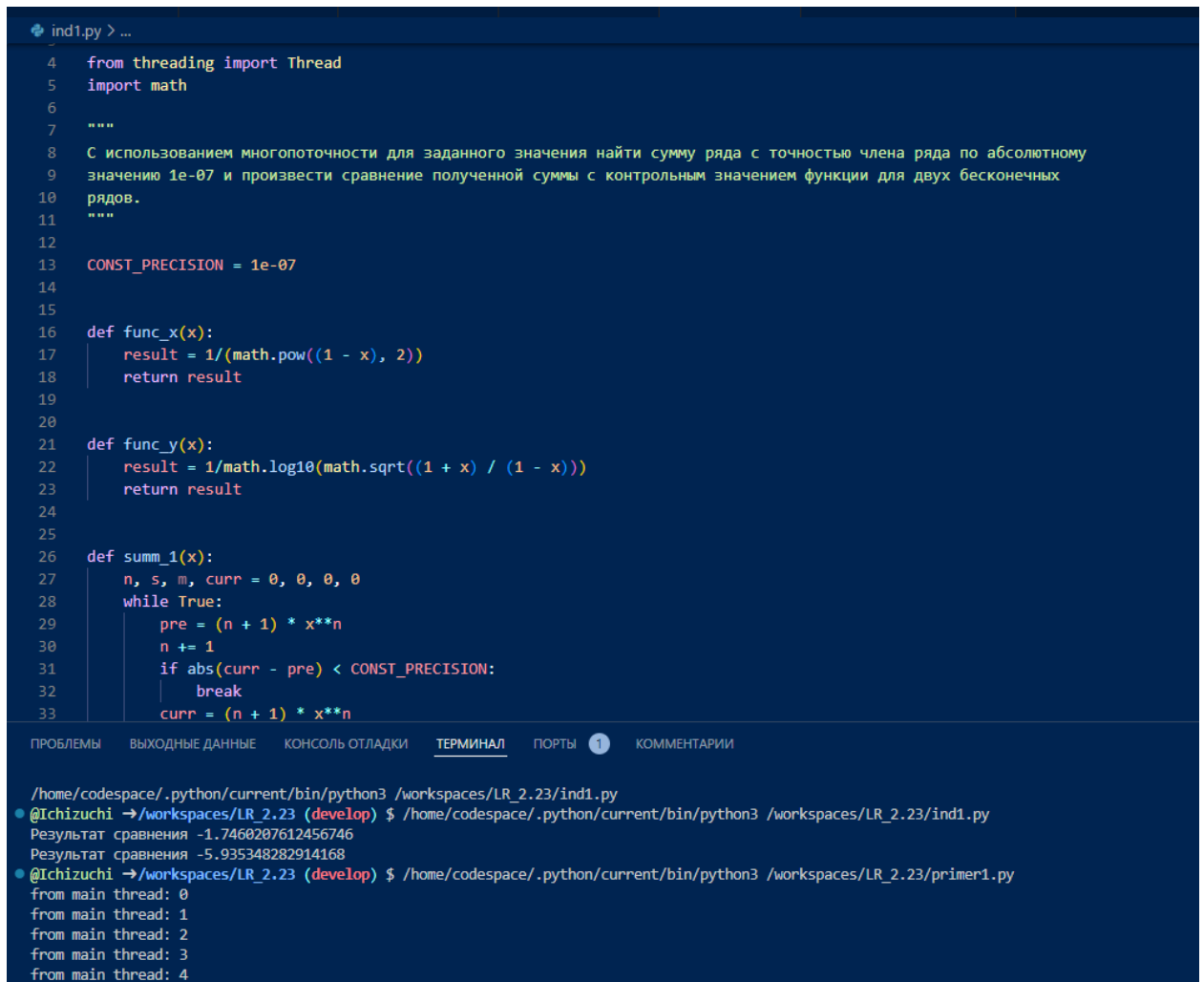
ПРОБЛЕМЫ ВЫХОДНЫЕ ДАННЫЕ КОНСОЛЬ ОТЛАДКИ ТЕРМИНАЛ ПОРТЫ 1 КОММЕНТАРИИ

```
/home/codespace/.python/current/bin/python3 /workspaces/LR_2.23/primer1.py
@Ichizuchi →/workspaces/LR_2.23 (develop) $ /home/codespace/.python/current/bin/python3 /workspaces/LR_2.23/primer1.py
from main thread: 0
from main thread: 1
from main thread: 2
from main thread: 3
from main thread: 4
@Ichizuchi →/workspaces/LR_2.23 (develop) $
```

Рисунок 4. Результат выполнения примера №1

Выполнение индивидуального задания

С использованием многопоточности для заданного значения найти сумму ряда с точностью члена ряда по абсолютному значению $1e-07$ и произвести сравнение полученной суммы с контрольным значением функции для двух бесконечных рядов.



```
ind1.py > ...
4 from threading import Thread
5 import math
6
7 """
8 С использованием многопоточности для заданного значения найти сумму ряда с точностью члена ряда по абсолютному
9 значению 1e-07 и произвести сравнение полученной суммы с контрольным значением функции для двух бесконечных
10 рядов.
11 """
12
13 CONST_PRECISION = 1e-07
14
15
16 def func_x(x):
17     result = 1/(math.pow((1 - x), 2))
18     return result
19
20
21 def func_y(x):
22     result = 1/math.log10(math.sqrt((1 + x) / (1 - x)))
23     return result
24
25
26 def summ_1(x):
27     n, s, m, curr = 0, 0, 0, 0
28     while True:
29         pre = (n + 1) * x**n
30         n += 1
31         if abs(curr - pre) < CONST_PRECISION:
32             break
33         curr = (n + 1) * x**n
34
35
36 # Проблемы
37 # Выходные данные
38 # Консоль отладки
39 # Терминал
40 # Порты
41 # Комментарии
42
43 /home/codespace/.python/current/bin/python3 /workspaces/LR_2.23/ind1.py
44 @Ichizuchi →/workspaces/LR_2.23 (develop) $ /home/codespace/.python/current/bin/python3 /workspaces/LR_2.23/ind1.py
45 Результат сравнения -1.7460207612456746
46 Результат сравнения -5.935348282914168
47 @Ichizuchi →/workspaces/LR_2.23 (develop) $ /home/codespace/.python/current/bin/python3 /workspaces/LR_2.23/primer1.py
48 from main thread: 0
49 from main thread: 1
50 from main thread: 2
51 from main thread: 3
52 from main thread: 4
```

Рисунок 7.Результат выполнения

4. Сформировала файлы environment.yml и requirements.txt

```
• @Ichizuchi →/workspaces/LR_2.23 (develop) $ conda env export > environment.yml
• @Ichizuchi →/workspaces/LR_2.23 (develop) $ pip freeze > requirements.txt
• @Ichizuchi →/workspaces/LR_2.23 (develop) $ conda init
no change      /opt/conda/condabin/conda
no change      /opt/conda/bin/conda
no change      /opt/conda/bin/conda-env
no change      /opt/conda/bin/activate
no change      /opt/conda/bin/deactivate
no change      /opt/conda/etc/profile.d/conda.sh
no change      /opt/conda/etc/fish/conf.d/conda.fish
no change      /opt/conda/shell/condabin/Conda.psm1
no change      /opt/conda/shell/condabin/conda-hook.ps1
no change      /opt/conda/lib/python3.12/site-packages/xontrib/conda_xsh
```

Рисунок 8. Файлы environment.yml и requirements.txt

Ответы на контрольные вопросы

1. Непосредственно модуль sqlite3 – это API к СУБД SQLite. Своего рода адаптер, который переводит команды, написанные на Питоне, в команды, которые понимает SQLite. Как и наоборот, доставляет ответы от SQLite в python-программу.

2. Для взаимодействия с базой данных SQLite3 в Python необходимо создать объект cursor. Вы можете создать его с помощью метода cursor() . Курсор SQLite3 – это метод объекта соединения. Для выполнения инструкций SQLite3 сначала устанавливается соединение, а затем создается объект курсора с использованием объекта соединения

3. При создании соединения с SQLite3 автоматически создается файл базы данных, если он еще не существует. Этот файл базы данных создается на диске, мы также можем создать базу данных в оперативной памяти с помощью функции :memory: with the connect. Такая база данных называется базой данных в памяти.

4. С помощью команды закрытия close().

5. Чтобы вставить данные в таблицу, используется оператор INSERT INTO.

6. Чтобы обновить данные в таблице, просто создайте соединение, затем создайте объект курсора с помощью соединения и, наконец, используйте оператор UPDATE.

7. Оператор SELECT используется для выбора данных из определенной таблицы. Если вы хотите выбрать все столбцы данных из таблицы, вы можете использовать звездочку (*).

8. SQLite3 rowcount используется для возврата количества строк, которые были затронуты или выбраны последним выполненным SQLзапросом.

9. Чтобы перечислить все таблицы в базе данных SQLite3, вы должны запросить данные из таблицы sqlite_master, а затем использовать fetchall() для получения результатов из инструкции SELECT

10. При создании таблицы мы должны убедиться, что она еще не существует. Аналогично, при удалении/удалении таблицы она должна существовать. Чтобы проверить, не существует ли таблица уже, мы используем IF NOT EXISTS с оператором CREATE TABLE следующим образом.

11. Метод executemany можно использовать для вставки нескольких строк одновременно.

12. В базе данных Python SQLite3 мы можем легко хранить дату или время, импортируя модуль datetime . Следующие форматы являются наиболее часто используемыми форматами для datetime:

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