

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 №13
дисциплины «Основы программной инженерии»

Выполнил:
Матвеев Александр Иванович
2 курс, группа ПИЖ-б-о-22-1,
09.03.04 «Программная инженерия»,
направленность (профиль) «Разработка и
сопровождение программного
обеспечения», очная форма обучения

(подпись)

Проверил Воронкин Роман Александрович

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

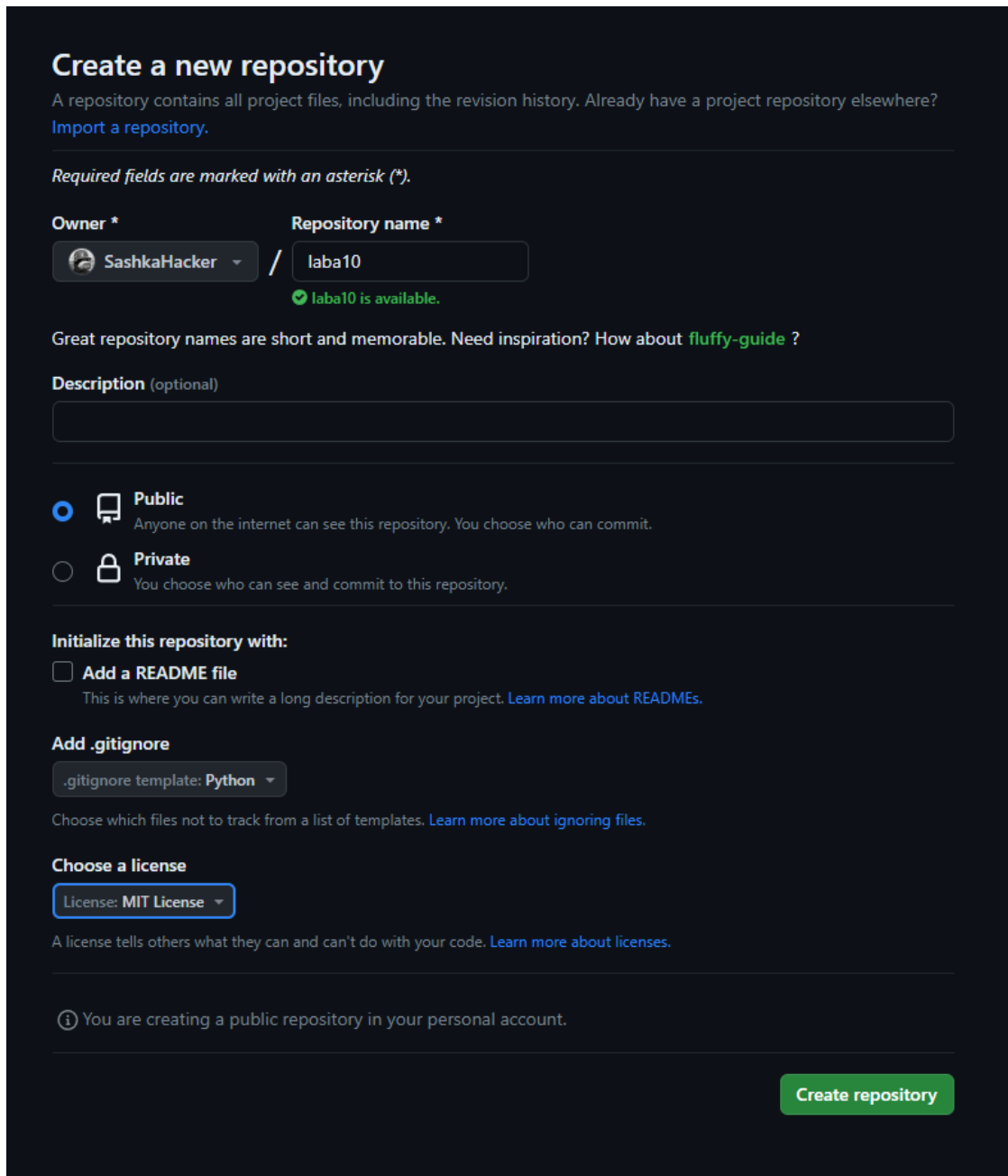
Ставрополь, 2023 г.

Тема: Функции с переменным числом параметров в Python.

Цель работы: приобретение навыков по работе с функциями с переменным числом параметров при написании программ с помощью языка программирования Python версии 3.x.

Ход работы.

1. Создание нового репозитория с лицензией MIT.



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Required fields are marked with an asterisk ().*

Owner * SashkaHacker / **Repository name *** laba10

✔ laba10 is available.

Great repository names are short and memorable. Need inspiration? How about [fluffy-guide](#) ?

Description (optional)

☒ **Public**
Anyone on the internet can see this repository. You choose who can commit.

☐ **Private**
You choose who can see and commit to this repository.

Initialize this repository with:

☐ **Add a README file**
This is where you can write a long description for your project. [Learn more about READMEs.](#)

Add .gitignore

.gitignore template: Python

Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)

Choose a license

License: MIT License

A license tells others what they can and can't do with your code. [Learn more about licenses.](#)

ⓘ You are creating a public repository in your personal account.

Create repository

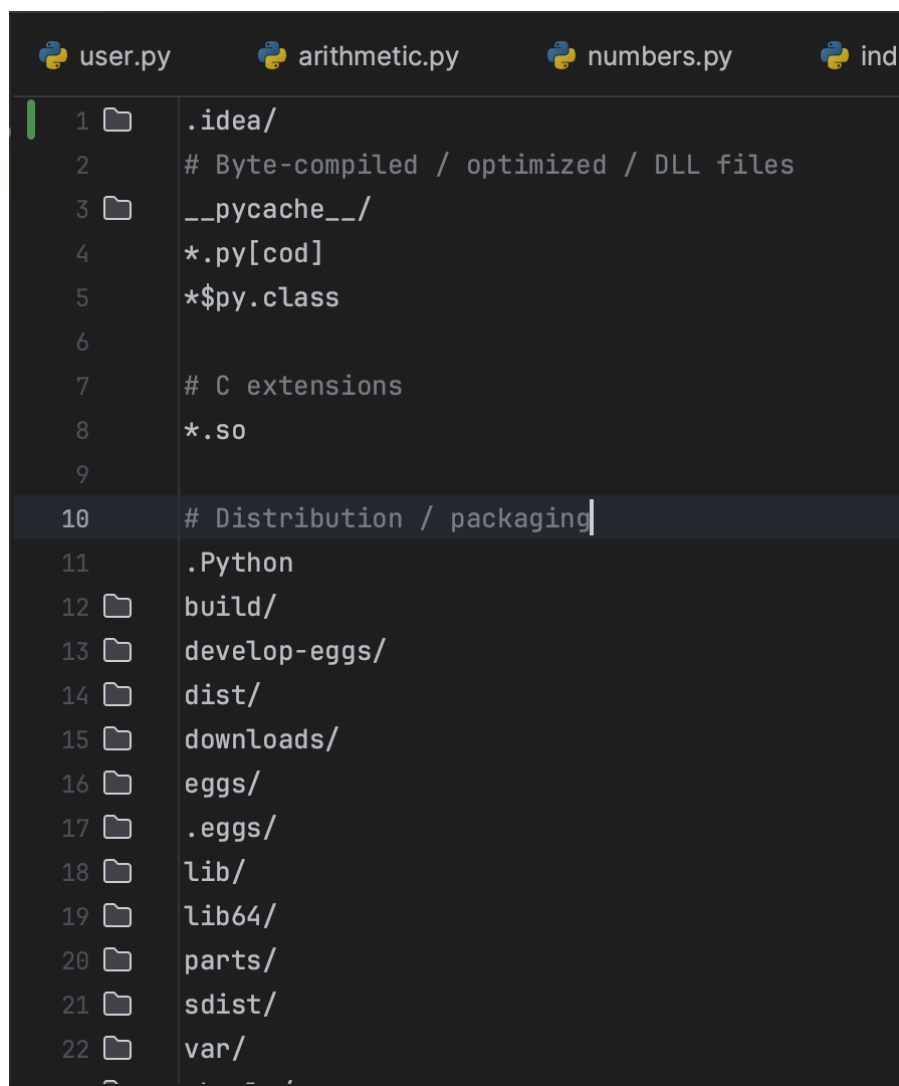
Рисунок 1 – Создание репозитория

2. Клонировал репозиторий на рабочий ПК.

```
Sashka@DESKTOP-U4RPSBI MINGW64 ~/Documents/GitHub
$ git clone https://github.com/SashkaHacker/lab10.git
Cloning into 'lab10'...
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (4/4), done.
```

Рисунок 2 – Клонирование репозитория

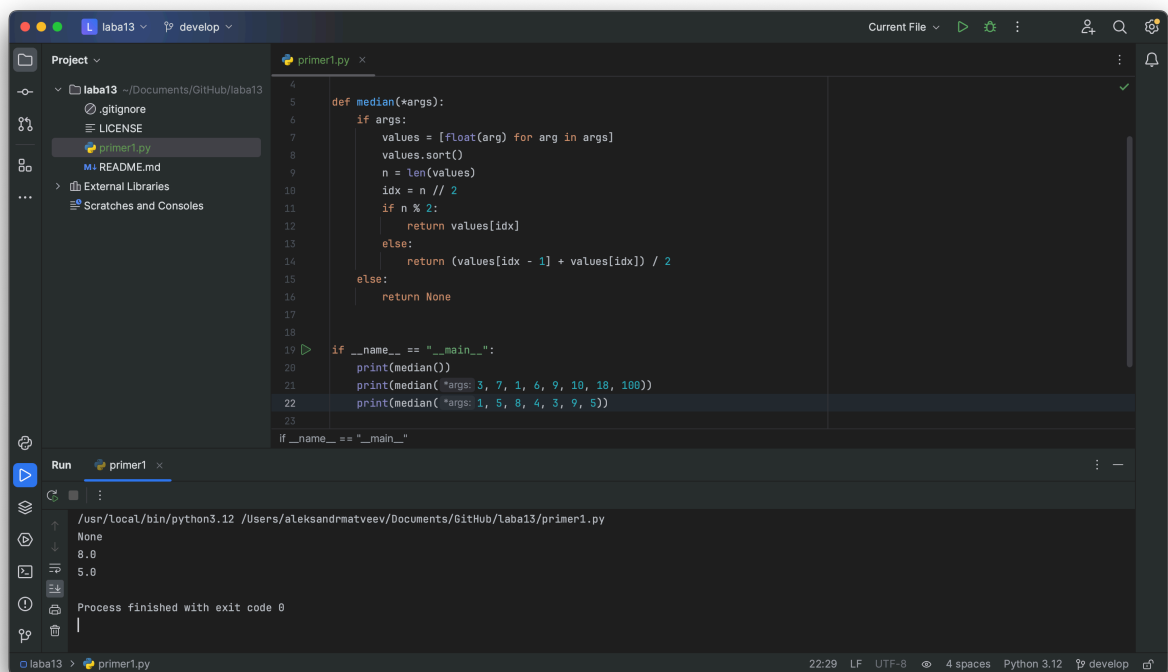
3. Дополнил файл .gitignore необходимыми инструкциями.



```
user.py arithmetic.py numbers.py ind
1  .idea/
2  # Byte-compiled / optimized / DLL files
3  __pycache__/
4  *.py[cod]
5  *$py.class
6
7  # C extensions
8  *.so
9
10 # Distribution / packaging
11 .Python
12 build/
13 develop-eggs/
14 dist/
15 downloads/
16 eggs/
17 .eggs/
18 lib/
19 lib64/
20 parts/
21 sdist/
22 var/
```

Рисунок 3 – Файл .gitignore

4. Проработка примеров из лабораторных задач.



The screenshot shows a code editor with a project named 'laba13'. The file 'primer1.py' is open, containing a Python function 'median' that calculates the median of a list of numbers. The function sorts the list and returns the middle element or the average of the two middle elements. The main block of the script calls 'median()' and 'median()' with two sets of arguments: (3, 7, 1, 6, 9, 10, 18, 100) and (1, 5, 8, 4, 3, 9, 5). The Run console shows the output: 'None', '8.0', and '5.0', indicating that the function returns None for the first call and the median values for the second call.

```
4
5 def median(*args):
6     if args:
7         values = [float(arg) for arg in args]
8         values.sort()
9         n = len(values)
10        idx = n // 2
11        if n % 2:
12            return values[idx]
13        else:
14            return (values[idx - 1] + values[idx]) / 2
15    else:
16        return None
17
18
19 if __name__ == "__main__":
20     print(median())
21     print(median(*args: 3, 7, 1, 6, 9, 10, 18, 100))
22     print(median(*args: 1, 5, 8, 4, 3, 9, 5))
23
if __name__ == "__main__":
```

Run primer1

/usr/local/bin/python3.12 /Users/aleksandrmatveev/Documents/GitHub/laba13/primer1.py
None
8.0
5.0
Process finished with exit code 0

Рисунок 4 – Пример №1

5. Решение задания №1.

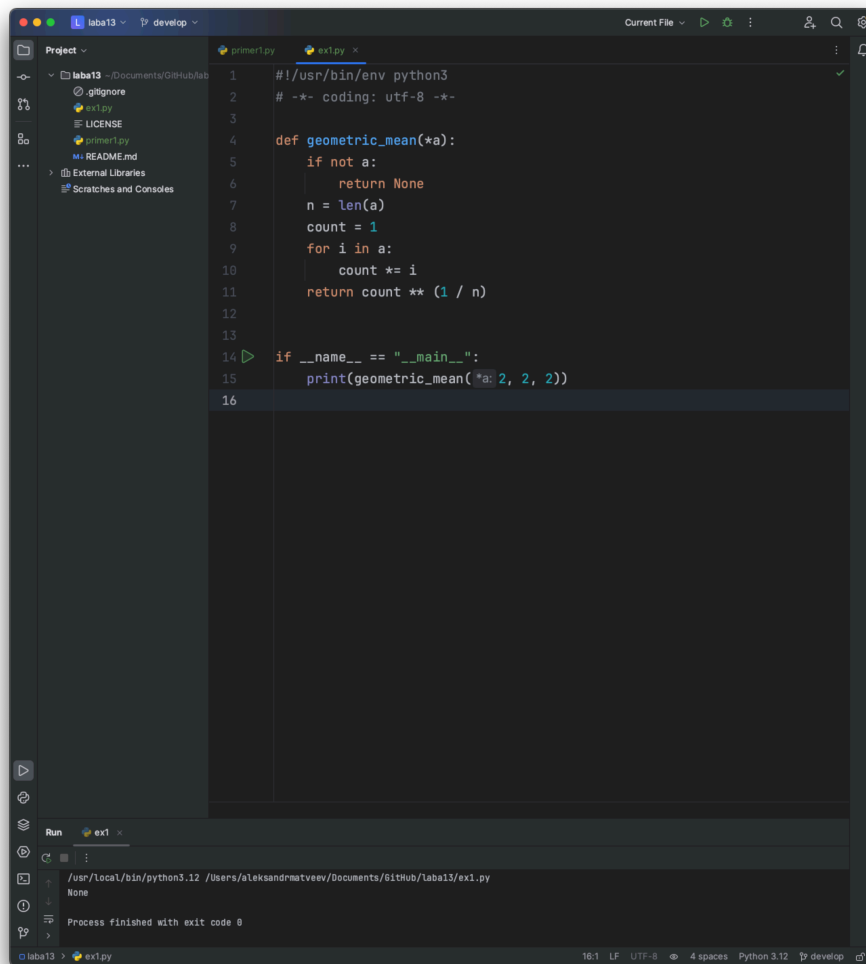


Рисунок 5 – Задание №1

6. Выполнить задание №2.

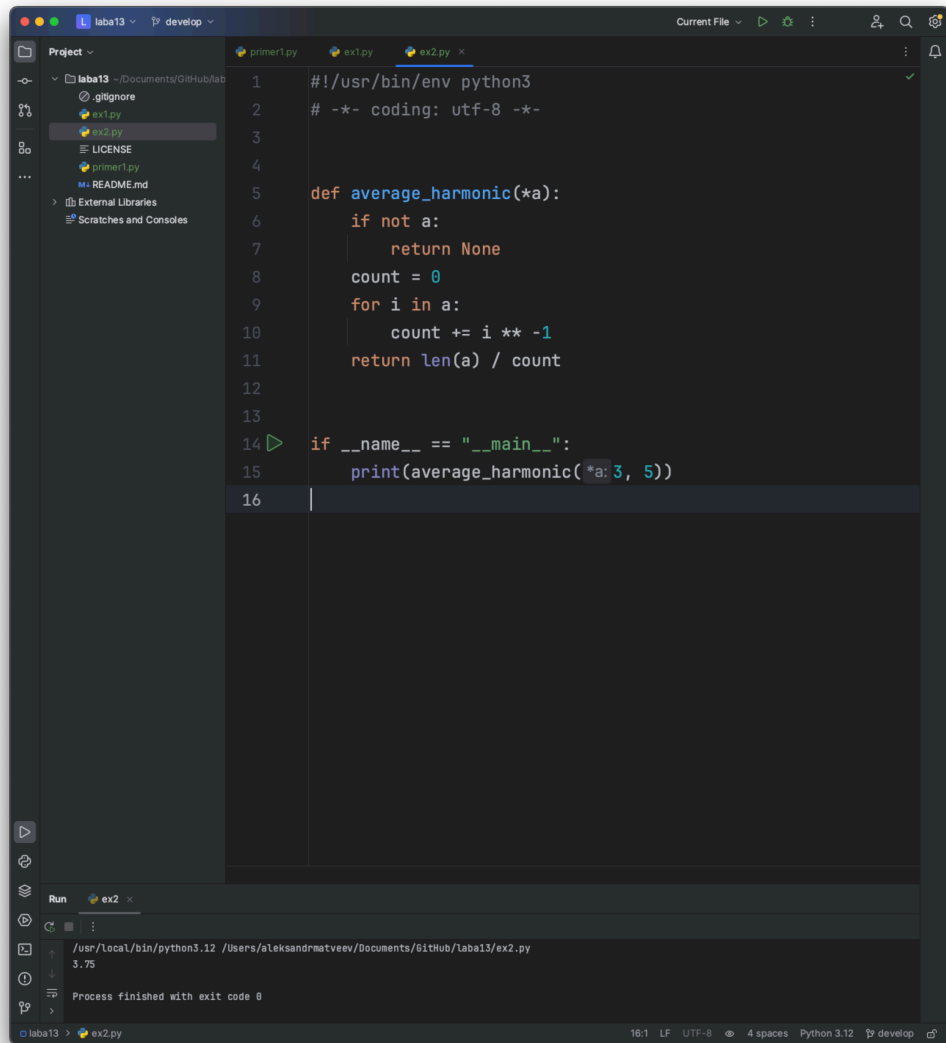


Рисунок 6 – Задание №2

7. Решение индивидуального задания.

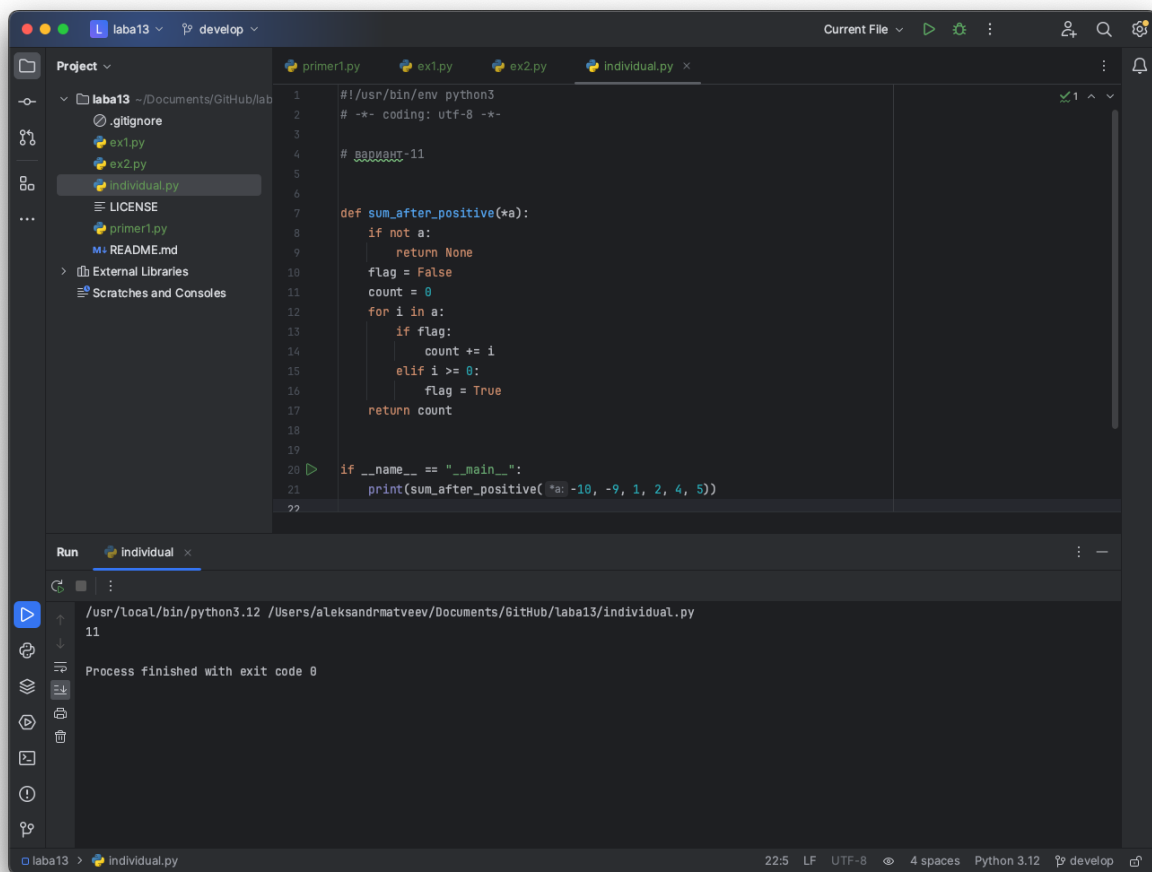


Рисунок 7 – Индивидуальное задание №1

8. Решите задачу: Напишите функцию `calculate_statistics`, которая принимает переменное количество именованных аргументов, представляющих собой числовые данные. Функция должна вычислять и возвращать следующую статистику для каждого набора данных: среднее значение, медиану и стандартное отклонение.

Пример вывода функции:

```
{
    'ages': {
        'mean': 30.4,
        'median': 31,
        'std_dev': 11.07
    },
    'scores': {
        'mean': 86.6,
        'median': 88,
        'std_dev': 5.57
    }
}
```

Рисунок 8 – Необходимый вывод для функции

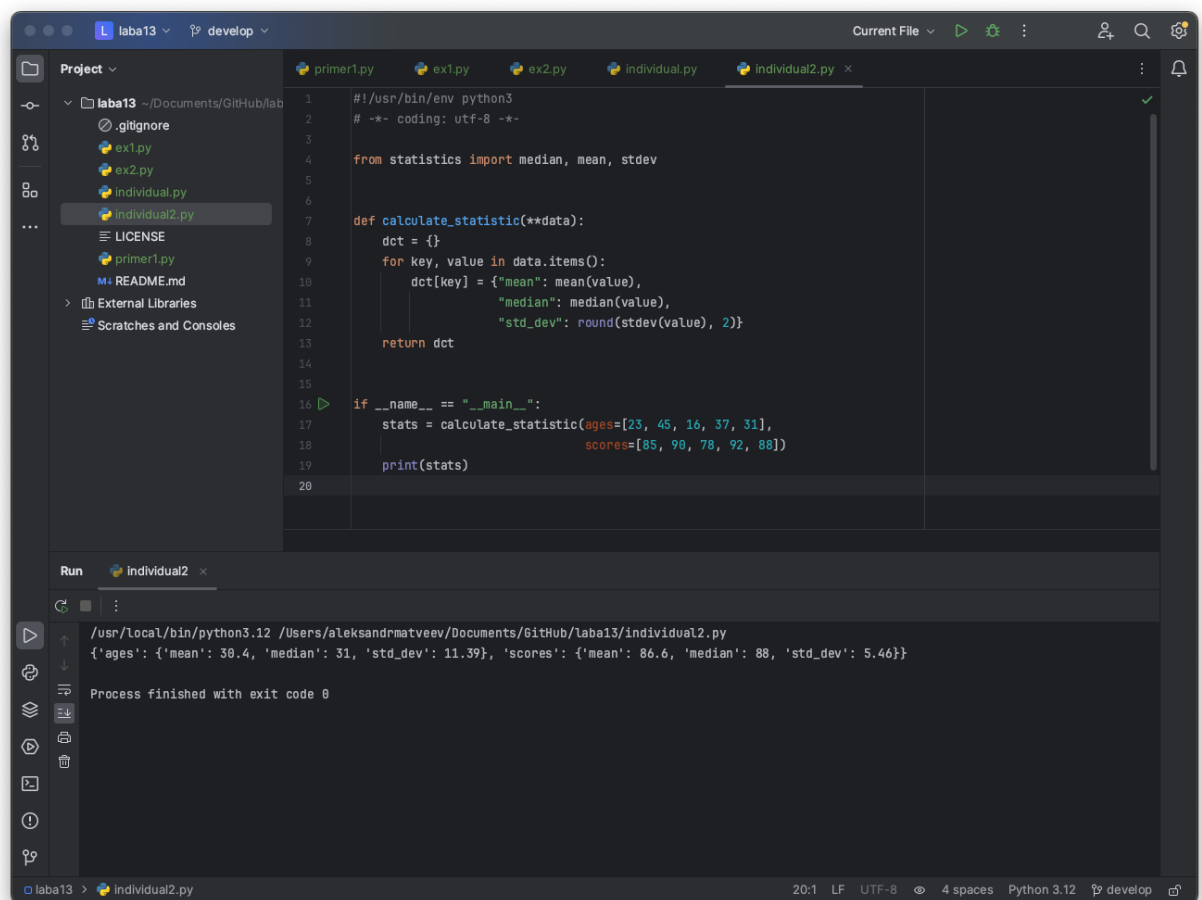


Рисунок 9 – Решение задания

Контрольные вопросы:

1. Позиционные аргументы в Python — это аргументы, которые передаются в функцию в том порядке, в котором они определены. Позиция этих аргументов имеет значение при их передаче.
2. Именованные аргументы в Python, также известные как ключевые аргументы, передаются в функцию с использованием имени параметра. Порядок этих аргументов не имеет значения, поскольку они передаются по имени.
3. Оператор `*` в Python используется для передачи переменного числа аргументов в функцию.
4. Конструкции `*args` и `**kwargs` в Python используются для передачи переменного числа аргументов в функцию. `*args` используется для передачи неименованных аргументов, а `**kwargs` - для передачи именованных аргументов.