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 №7
дисциплины «Основы программной инженерии»

Выполнила:
Панюкова Ксения Юрьевна
2 курс, группа ПИЖ-б-о-22-1,
09.03.04 «Программная инженерия»,
направленность (профиль) «Разработка
и сопровождение программного
обеспечения», очная форма обучения

_____
(подпись)

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

_____
(подпись)

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2023 г.

Ход работы

1. Я изучила теоретический материал работы

Рисунок 1.1 – Изучение материала для лабораторной работы

2. Создала общедоступный репозиторий на GitHub, в котором будет использована лицензия MIT и язык программирования Python

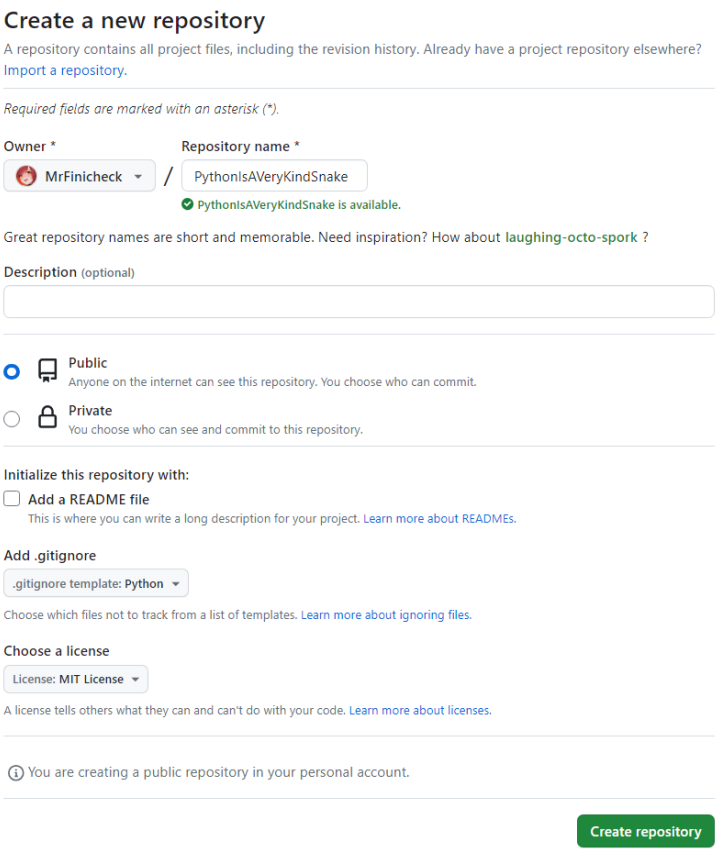

Рисунок 2.1 – Настройка репозитория

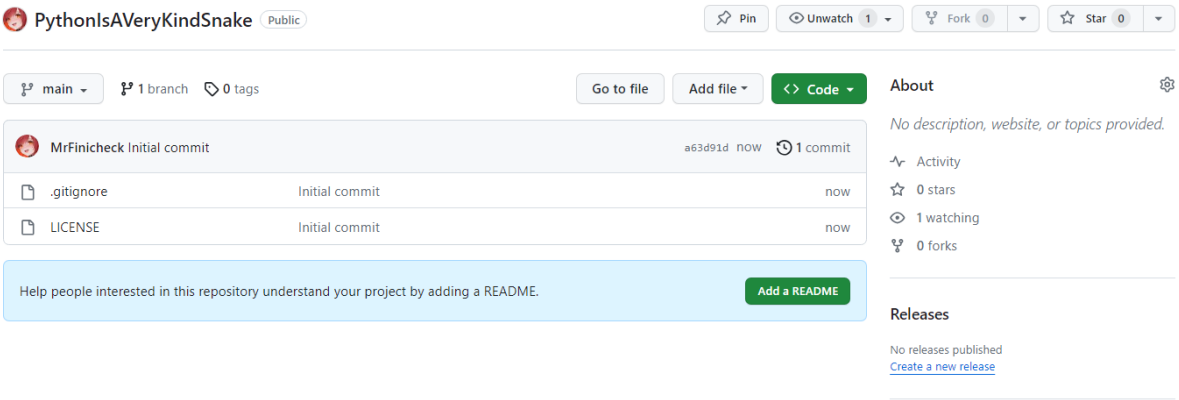

Рисунок 2.2 – Готовый репозиторий

3. Выполняю клонирование созданного репозитория

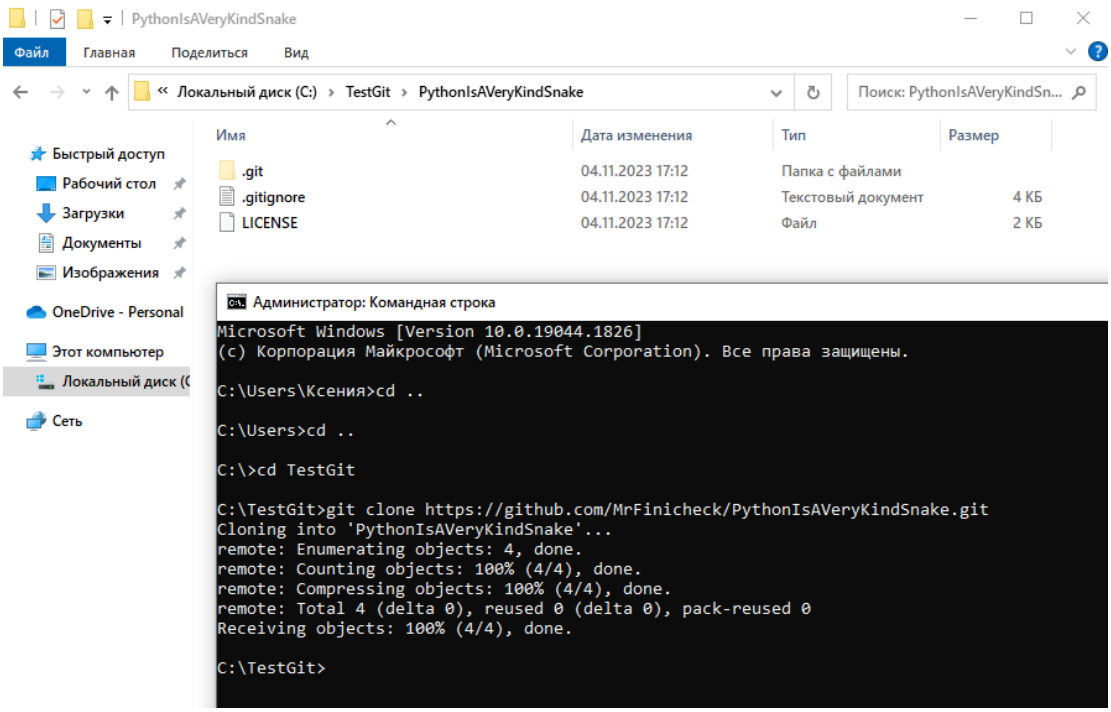

Рисунок 3.1 – Клонирование репозитория на локальный диск

4. Дополнила файл .gitignore необходимыми правилами для работы с IDE PyCharm

python pycharm gitignore

py-gitignore Raw

```
# Created by .ignore support plugin (hsz.mobi)
### Python template
# Byte-compiled / optimized / DLL files
__pycache__/
*.py[cod]
*$py.class

# C extensions
*.so

# Distribution / packaging
.Python
env/
build/
develop-eggs/
dist/
downloads/
eggs/
.eggs/
lib/
lib64/
parts/
sdist/
var/
*.egg-info/
.installed.cfg
*.egg

# PyInstaller
#  Usually these files are written by a python script from a template
#  before PyInstaller builds the exe, so as to inject date/other infos into it.
*.manifest
*.spec

# Installer logs
```

Рисунок 4.1 – .gitignore для IDE PyCharm

5. Организовала свой репозиторий в соответствии с моделью ветвления git-flow

```
C:\TestGit\PythonIsAVeryKindSnake>git checkout develop
Switched to branch 'develop'

C:\TestGit\PythonIsAVeryKindSnake>git push -u origin develop
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'develop' on GitHub by visiting:
remote:      https://github.com/MrFinicheck/PythonIsAVeryKindSnake/pull/new/develop
remote:
To https://github.com/MrFinicheck/PythonIsAVeryKindSnake.git
 * [new branch]      develop -> develop
branch 'develop' set up to track 'origin/develop'.
```

Рисунок 5.1 – Создание ветки develop от ветки main

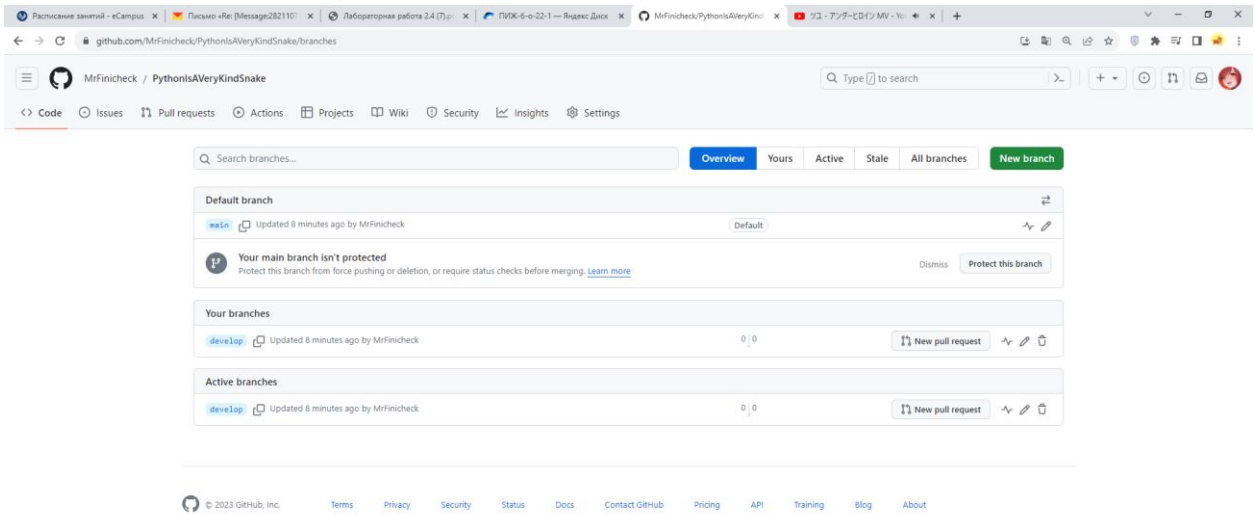

Рисунок 5.2 – Ветка develop на GitHub

6. Создала проект PyCharm в папке репозитория

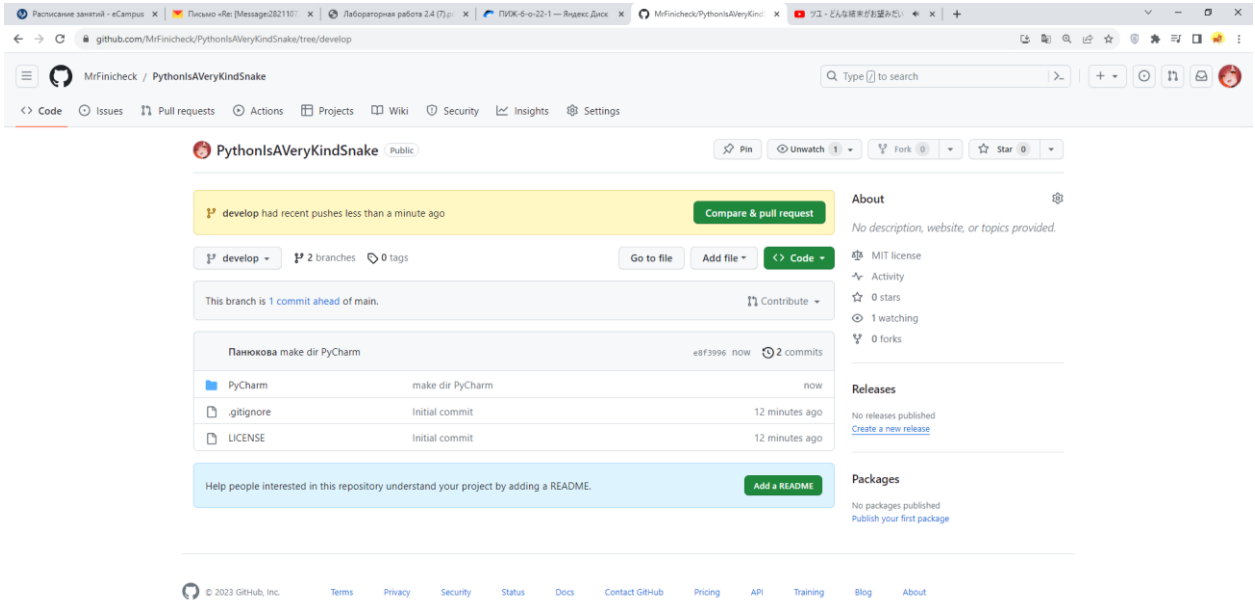

Рисунок 6.1 – Репозиторий с проектом PyCharm

7. Проработала примеры лабораторной работы. Создала для каждого примера отдельный модуль языка Python. Зафиксировала изменения в репозитории.

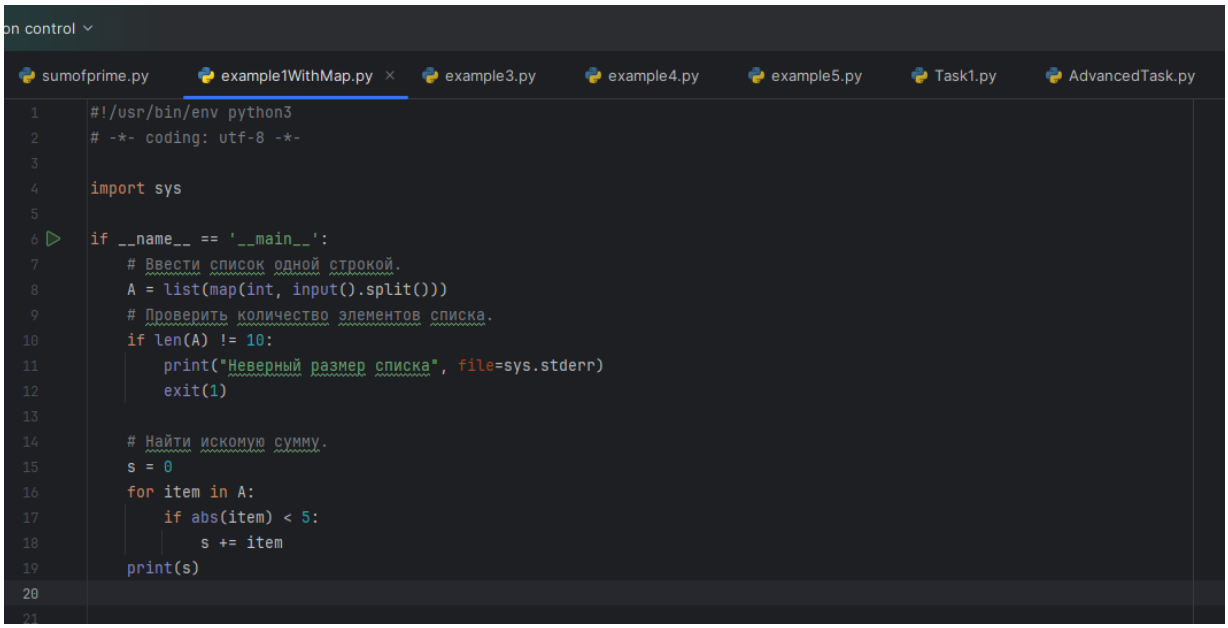

Рисунок 7.1 – Проработка примера 1 с применением map()

Рисунок 7.2 – Проработка примера 1 с применением списковых включений

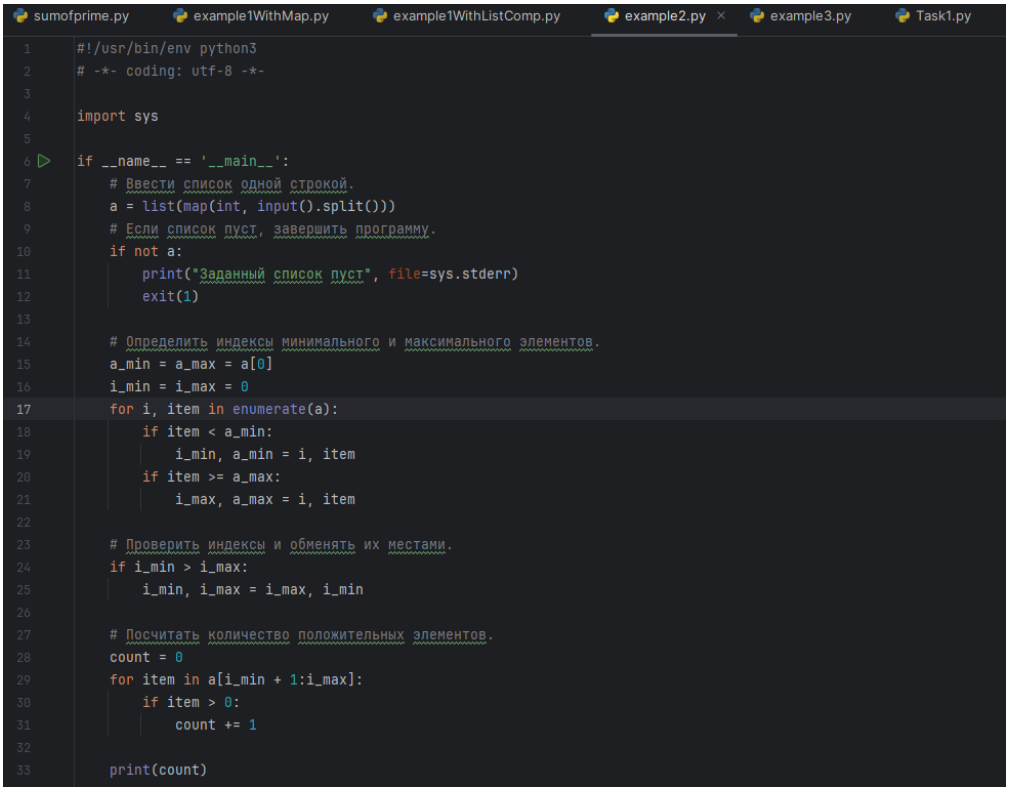

Рисунок 7.3 – Проработка примера 2

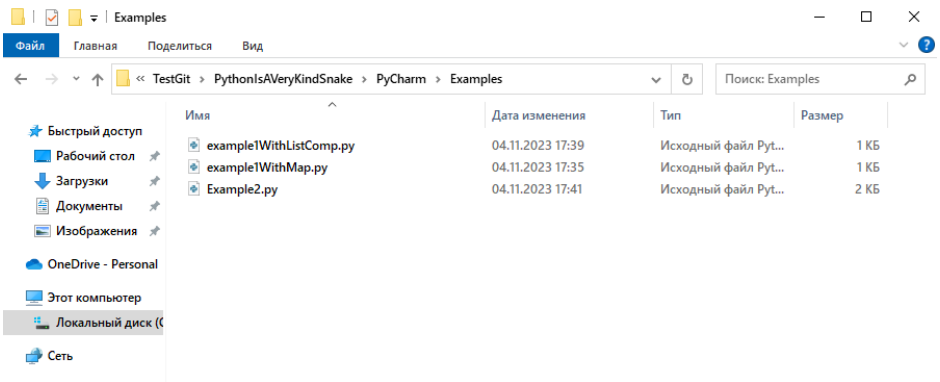

Рисунок 7.4 – Создание отдельных модулей для каждого из примеров

```
C:\TestGit\PythonIsAVeryKindSnake>git add PyCharm

C:\TestGit\PythonIsAVeryKindSnake>git commit -m"adding examples"
[develop 91306e3] adding examples
 4 files changed, 68 insertions(+)
 create mode 100644 PyCharm/Examples/Example2.py
 create mode 100644 PyCharm/Examples/example1WithListComp.py
 create mode 100644 PyCharm/Examples/example1WithMap.py
 delete mode 100644 PyCharm/example1.py

C:\TestGit\PythonIsAVeryKindSnake>
```

Рисунок 7.5 – Фиксирование изменений в репозитории

8. Привела в отчете скриншоты результатов выполнения каждой из программ примеров при различных исходных данных, вводимых с клавиатуры.

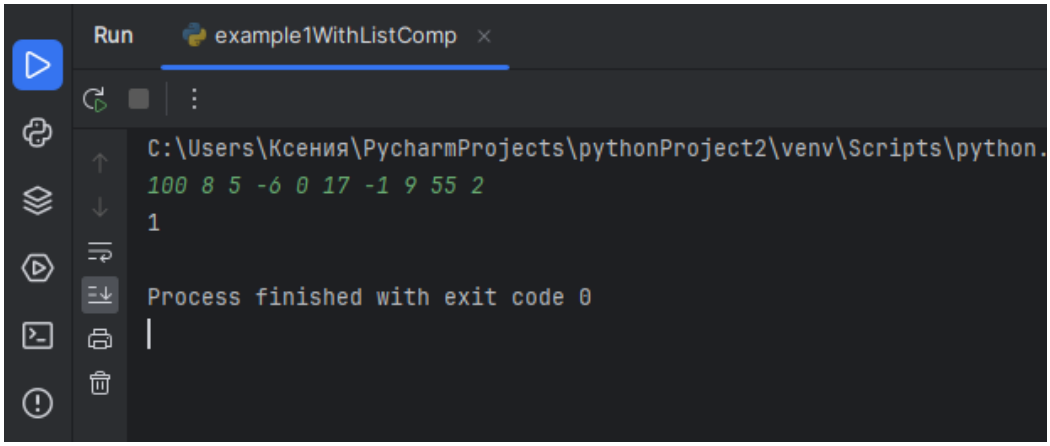

Рисунок 8.1 – Результат примера 1 с применением map()

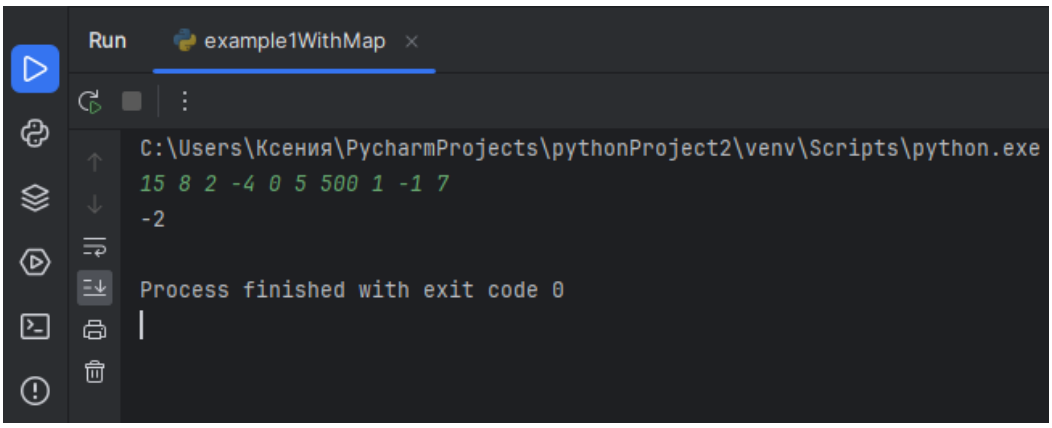

Рисунок 8.2 – Результат примера 2 с применением списковых включений

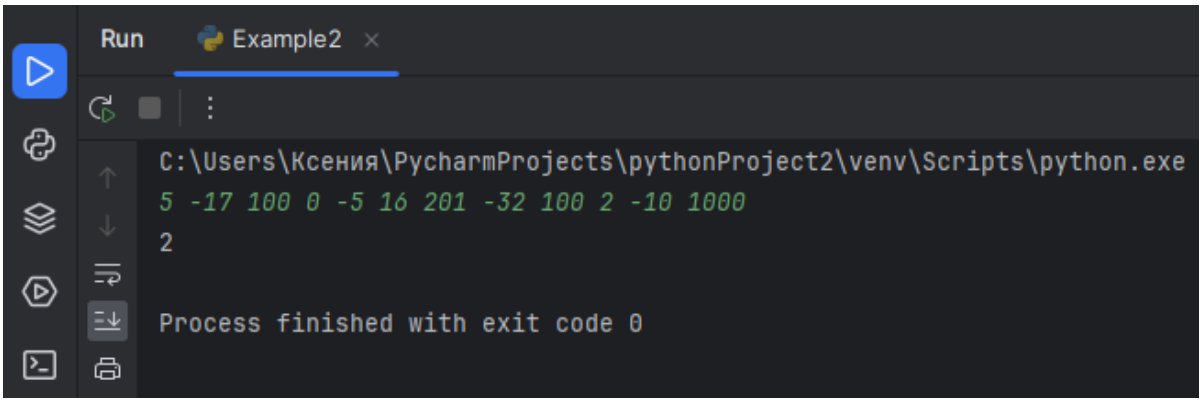

Рисунок 8.3 – Результат примера 2

9. Привела в отчете скриншоты работы программ решения индивидуальных заданий

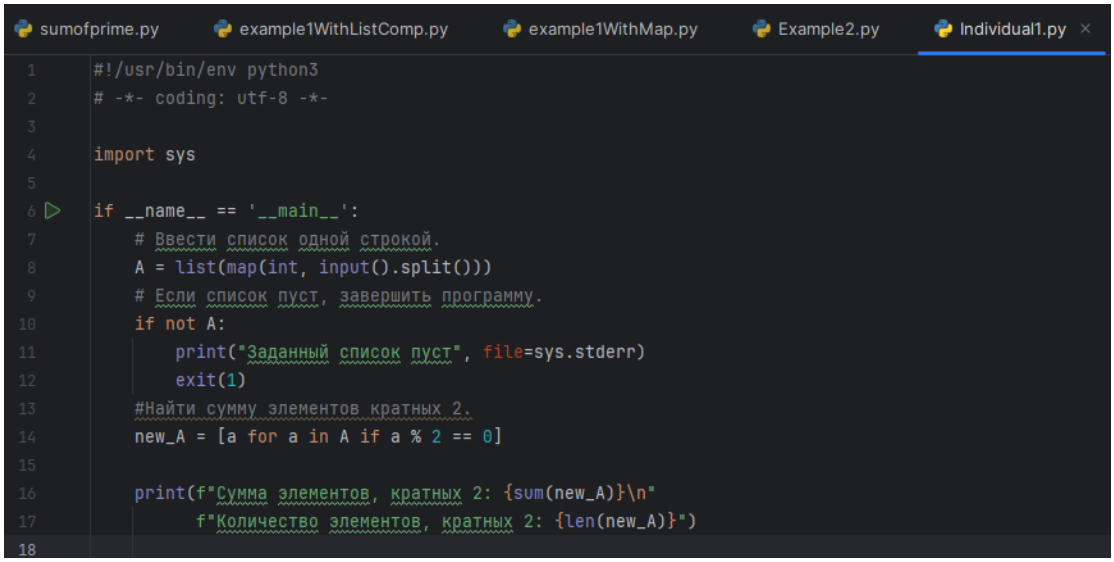

Рисунок 9.1 – Код программы Individual1.py в IDE PyCharm

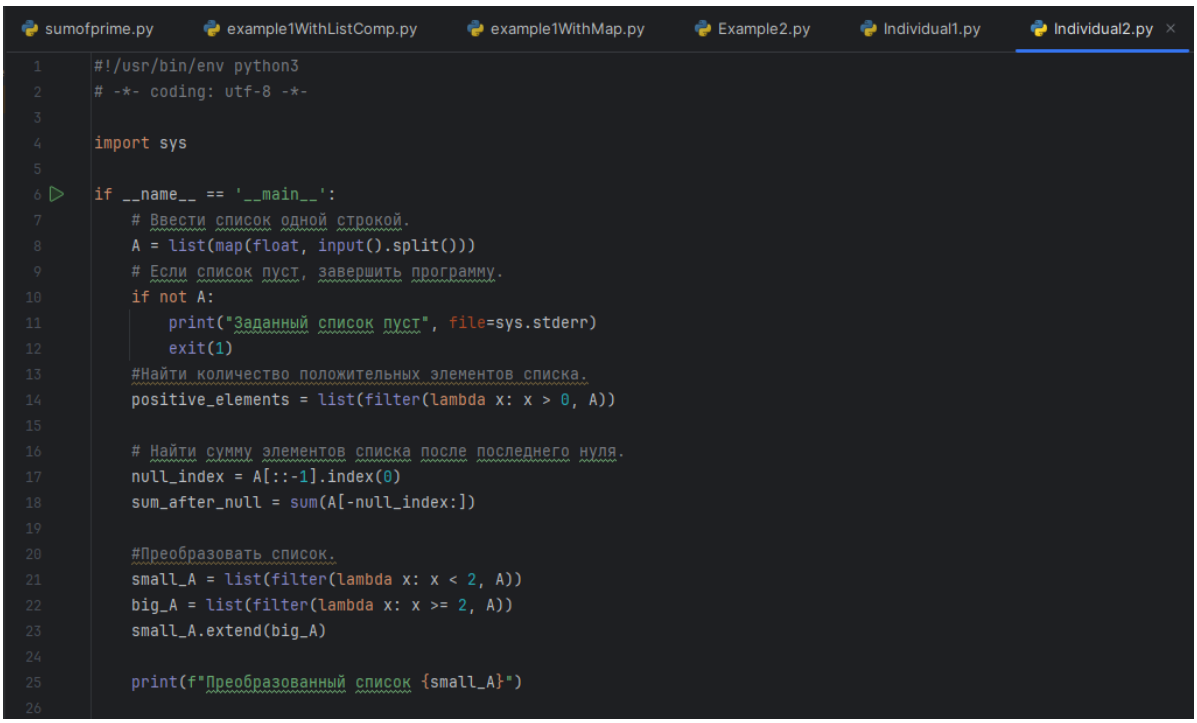

Рисунок 9.2 – Код программы Individual2.py в IDE PyCharm

10. Зафиксировала сделанные изменения в репозитории

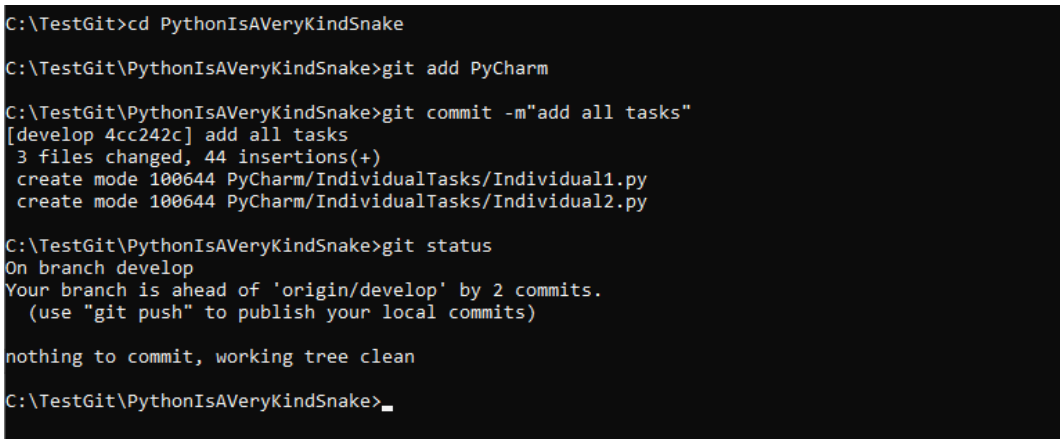

Рисунок 10.1 – Коммит файлов в репозитории git

Контрольные вопросы

1. Что такое списки в языке Python?

Список (list) – это структура данных для хранения объектов различных типов. Если вы использовали другие языки программирования, то вам должно быть знакомо понятие массива. Так вот, список очень похож на массив, только, как было уже сказано выше, в нем можно хранить объекты различных типов.

Размер списка не статичен, его можно изменять. Список по своей природе является изменяемым типом данных. Переменная, определяемая как список, содержит ссылку на структуру в памяти, которая в свою очередь хранит ссылки на какие-либо другие объекты или структуры

2. Как осуществляется создание списка в Python?

Для создания списка нужно заключить элементы в квадратные скобки:

```
my_list = [1, 2, 3, 4, 5]
```

3. Как организовано хранение списков в оперативной памяти?

Список является изменяемым типом данных. При его создании в памяти резервируется область, которую можно условно назвать некоторым «контейнером», в котором хранятся ссылки на другие элементы данных в памяти. В отличии от таких типов данных как число или строка, содержимое «контейнера» списка можно менять.

4. Каким образом можно перебрать все элементы списка?

Читать элементы списка можно с помощью следующего цикла:

```
my_list = ['один', 'два', 'три', 'четыре', 'пять']
for elem in my_list:
    print(elem)
```

5. Какие существуют арифметические операции со списками?

Для объединения списков можно использовать оператор сложения (+). Список можно повторить с помощью оператора умножения (*).

6. Как проверить есть ли элемент в списке?

Для того, чтобы проверить, есть ли заданный элемент в списке Python необходимо использовать оператор in

7. Как определить число вхождений заданного элемента в списке?

Метод count можно использовать для определения числа сколько раз данный элемент встречается в списке.

8. Как осуществляется добавление (вставка) элемента в список?

Метод insert можно использовать, чтобы вставить элемент в список.

9. Как выполнить сортировку списка?

Для сортировки списка нужно использовать метод sort. Для сортировки списка в порядке убывания необходимо вызвать метод sort с аргументом reverse=True.

10. Как удалить один или несколько элементов из списка?

Удалить элемент можно, написав его индекс в методе pop. Элемент можно удалить с помощью метода remove. Оператор del можно использовать для тех же целей. Можно удалить несколько элементов с помощью оператора среза.

11. Что такое списковое включение и как с его помощью осуществлять обработку списков?

List Comprehensions чаще всего на русский язык переводят как абстракция списков или списковое включение, является частью синтаксиса языка, которая предоставляет простой способ построения списков. Проще всего работу list comprehensions показать на примере. Допустим вам необходимо создать список целых чисел от 0 до n, где n предварительно задается. Классический способ решения данной задачи выглядел бы так.

В языке Python есть две очень мощные функции для работы с коллекциями: map и filter. Они позволяют использовать функциональный стиль программирования, не прибегая к помощи циклов, для работы с такими типами как list, tuple, set, dict и т.п. Списковое включение позволяет обойтись без этих функций.

12. Как осуществляется доступ к элементам списков с помощью срезов?

Слайсы (срезы) являются очень мощной составляющей Python, которая позволяет быстро и лаконично решать задачи выборки элементов из списка.

Слайс задается тройкой чисел, разделенных запятой: start:stop:step. Start – позиция, с которой нужно начать выборку, stop – конечная позиция, step – шаг. При этом необходимо помнить, что выборка не включает элемент определяемый stop.

13. Какие существуют функции агрегации для работы со списками?

Для работы со списками Python предоставляет следующие функции:

– len(L) - получить число элементов в списке L;

– min(L) - получить минимальный элемент списка L;

– max(L) - получить максимальный элемент списка L;

– sum(L) - получить сумму элементов списка L, если список L содержит только числовые значения.

14. Как создать копию списка?

Для создания копии списка необходимо использовать либо метод copy, либо использовать оператор среза.

15. Самостоятельно изучите функцию sorted языка Python. В чем ее отличие от метода sort списков?

Функция sort() предназначена для сортировки списка на месте, то есть изменяет сам список, а функция sorted() возвращает новый отсортированный список, не изменяя исходный.