

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ  
ФЕДЕРАЦИИ**  
**Федеральное государственное автономное  
образовательное учреждение высшего образования  
«СЕВЕРОКАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

**Кафедра инфокоммуникаций**

**Институт цифрового развития**

**ОТЧЁТ**

**по лабораторной работе №2.2**

Дисциплина: «Основы кроссплатформенного программирования»

Тема: «Условные операторы и циклы в языке Python»

Выполнил: студент 1 курса

группы ИВТ-б-о-21-1

Кочкаров Умар Ахматович

Ставрополь 2022

## Ход работы

1. Создал репозиторий в GitHub, в который добавил .gitignore, который дополнил правилами для работы с IDE PyCharm с ЯП Python, выбрал лицензию MIT, клонировал его на лок. сервер и организовал в соответствие с моделью ветвления git-flow.

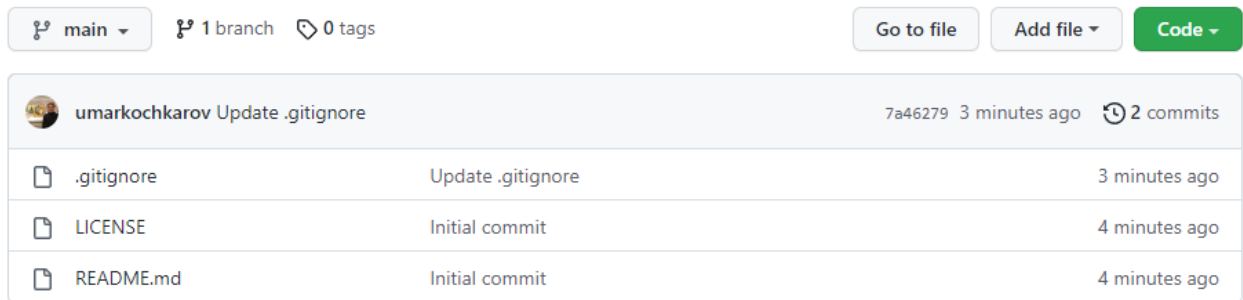


Рисунок 1.1 Репозиторий github

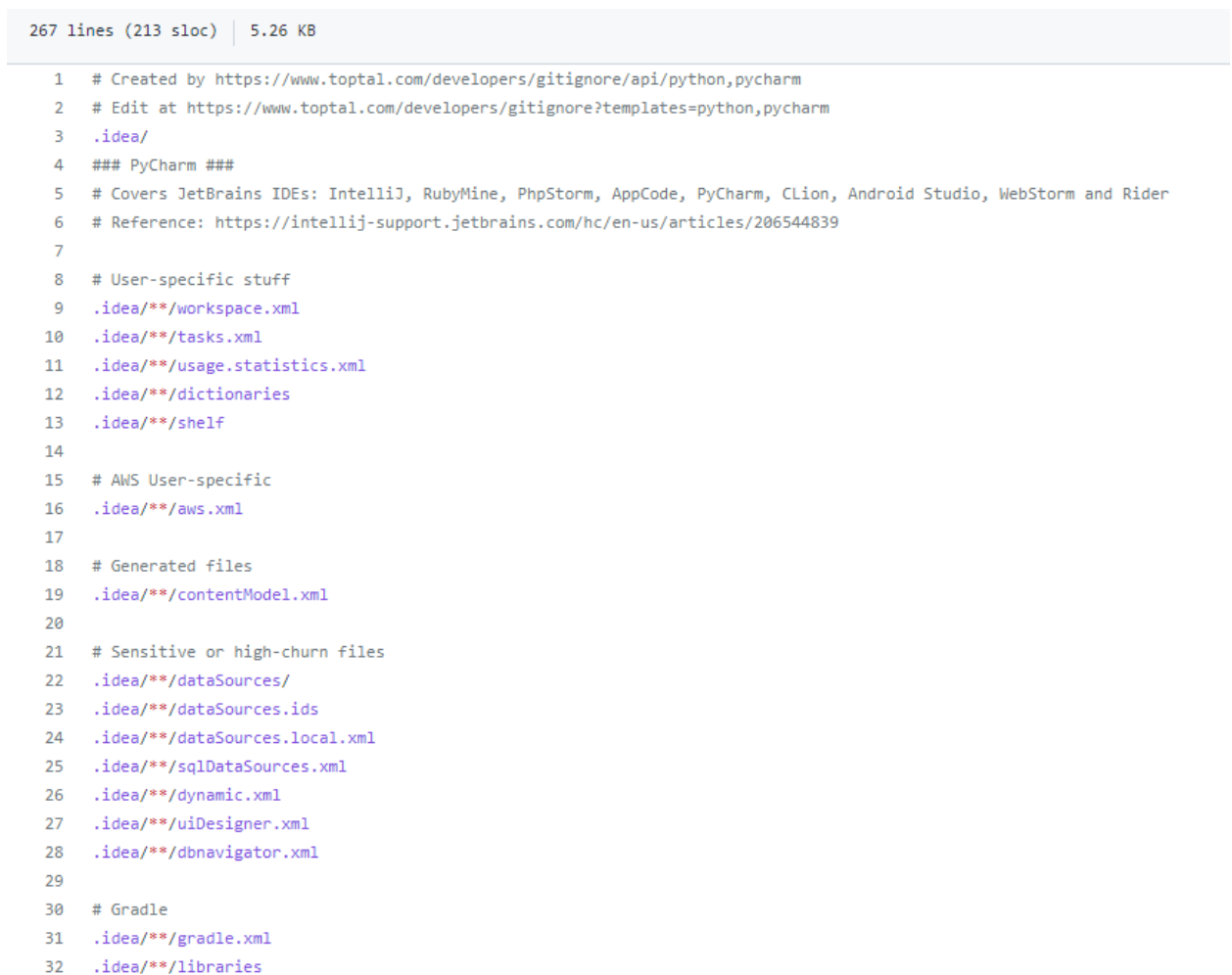


Рисунок 1.2 Добавление правил в .gitignore

```

erken@LAPTOP-ESTC60GF MINGW64 ~/Desktop/Универ/Основы крос/Лаб2.2
$ git clone https://github.com/umarkochkarov/lb2.2.git
Cloning into 'lb2.2'...
remote: Enumerating objects: 8, done.
remote: Counting objects: 100% (8/8), done.
remote: Compressing objects: 100% (7/7), done.
remote: Total 8 (delta 1), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (8/8), 4.35 KiB | 743.00 KiB/s, done.
Resolving deltas: 100% (1/1), done.

```

Рисунок 1.3 Клонирование репозитория

```

erken@LAPTOP-ESTC60GF MINGW64 ~/Desktop/Универ/Основы крос/Лаб2.2/lb2.2 (main)
$ git flow init

Which branch should be used for bringing forth production releases?
- main
Branch name for production releases: [main]
Branch name for "next release" development: [develop]

How to name your supporting branch prefixes?
Feature branches? [feature/]
Bugfix branches? [bugfix/]
Release branches? [release/]
Hotfix branches? [hotfix/]
Support branches? [support/]
Version tag prefix? []
Hooks and filters directory? [C:/Users/erken/Desktop/Универ/Основы крос/Лаб2.2/.git/hooks]

```

Рисунок 1.4 Организация репозитория согласно модели ветвления git – flow

## 2. Проработал примеры из лабораторной работы:

```

1  ▶  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3      import math
4
5  ▶  if __name__ == '__main__':
6      x = float(input("Value of x? "))
7
8      if x <= 0:
9          y = 2 * x * x + math.cos(x)
10     elif x < 5:
11         y = x + 1
12     else:
13         y = math.sin(x) - x * x
14
15     print(f"y = {y}")

```

Run - Пример\_1

Run: main ×

C:\Users\erken\Desktop\Универ\python\Пример\_1\venv\Scripts\python.exe C:/Users/erken/Desktop/Универ/Основы крос/Лаб2.2/lb2.2/lb2.2.py

Value of x? 100

y = -10000.50636564111

Рисунок 2.1 Пример 1

```
1 ▶ #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3 import sys
4
5 ▶ if __name__ == '__main__':
6     n = int(input("Введите номер месяца: "))
7
8     if n == 1 or n == 2 or n == 12:
9         print("Зима")
10    elif n == 3 or n == 4 or n == 5:
11        print("Весна")
12    elif n == 6 or n == 7 or n == 8:
13        print("Лето")
14    elif n == 9 or n == 10 or n == 11:
15        print("Осень")
16    else:
17        print("Ошибка!", file=sys.stderr)
18    exit(1)
```

Run - Пример\_1

Run: Пример\_2 ×

▶ C:\Users\erken\Desktop\Универ\python\Пример\_1\venv\Scripts\python.exe C:/Users/erken/Desktop/Универ/...  
Введите номер месяца: 3  
Весна

Рисунок 2.2 Пример 2

```
1 ▶ #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3 import math
4
5 ▶ if __name__ == '__main__':
6     n = int(input("Value of n? "))
7     x = float(input("Value of x? "))
8
9     S = 0.0
10    for k in range(1, n + 1):
11        a = math.log(k * x) / (k * k)
12        S += a
13
14    print(f"S = {S}")
```

Run - Пример\_1

Run: Пример\_3 ×

▶ C:\Users\erken\Desktop\Универ\python\Пример\_1\venv\Scripts\python.exe C:/Users/erken/Desktop/Универ/...  
Value of n? 6  
Value of x? 5  
S = 2.896444465760977

Рисунок 2.3 Пример 3

```

1  ▶  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3  import math
4  import sys
5
6  ▶  if __name__ == '__main__':
7      a = float(input("Value of a? "))
8      if a < 0:
9          print("Illegal value of a", file=sys.stderr)
10         exit(1)
11
12         x, eps = 1, 1e-10
13         while True:
14             xp = x
15             x = (x + a / x) / 2
16             if math.fabs(x - xp) < eps:
17                 break
18
19         print(f"x = {x}\nX = {math.sqrt(a)}")

```

Run: exercise\_4 x

C:\Users\erken\Desktop\Универ\python\Пример\_1\venv\Scripts\python.exe C:/Users/erken/...

Value of a? 4

x = 2.0

X = 2.0

Рисунок 2.4 Пример 4

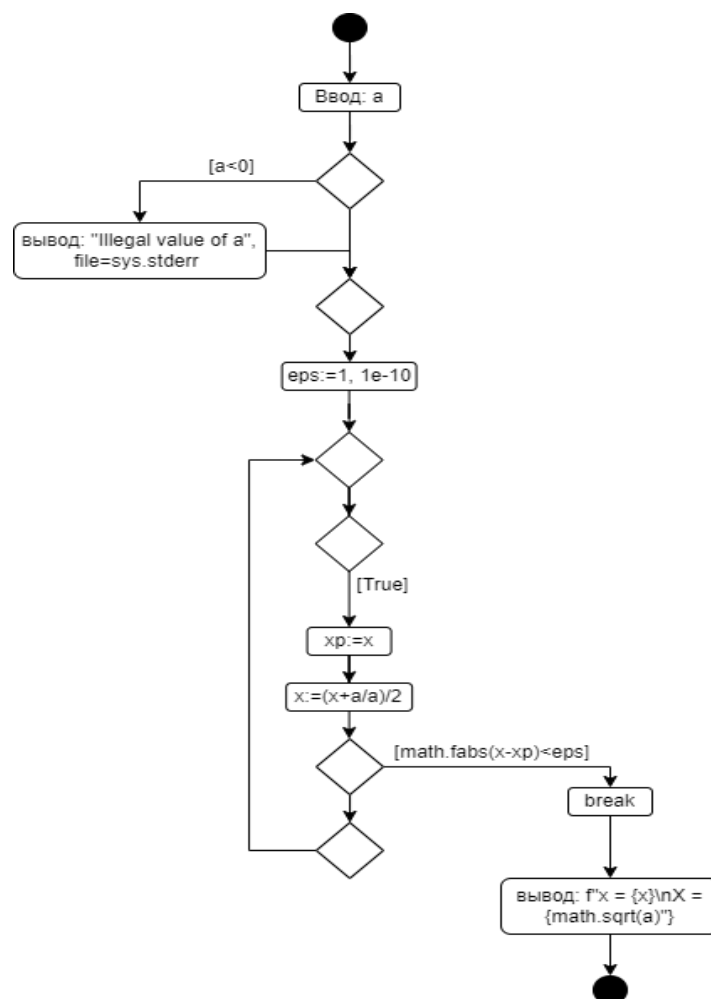


Рисунок 2.5 UML-диаграмма 4 примера

```
1 ▶ #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3 import math
4 import sys
5
6 # Постоянная Эйлера.
7 EULER = 0.5772156649015328606
8 # Точность вычислений.
9 EPS = 1e-10
10
11 ▶ if __name__ == '__main__':
12     x = float(input("Value of x? "))
13     if x == 0:
14         print("Illegal value of x", file=sys.stderr)
15         exit(1)
16
17     a = x
18     S, k = a, 1
19
20     # Найти сумму членов ряда.
21     while math.fabs(a) > EPS:
22         a *= x * k / (k + 1) ** 2
23         S += a
24         k += 1
25
26     # Вывести значение функции.
27     print(f"Ei({x}) = {EULER + math.log(math.fabs(x)) + S}")
```

PC Run - Пример\_1

Run: exercise\_5 ×

▶ C:\Users\erken\Desktop\Универ\python\Пример\_1\venv\Scripts\python.exe C:/Users/erken/Desktop/Vh  
Value of x? 19  
Ei(19.0) = 9950907.251046844

Рисунок 2.6 Пример 5

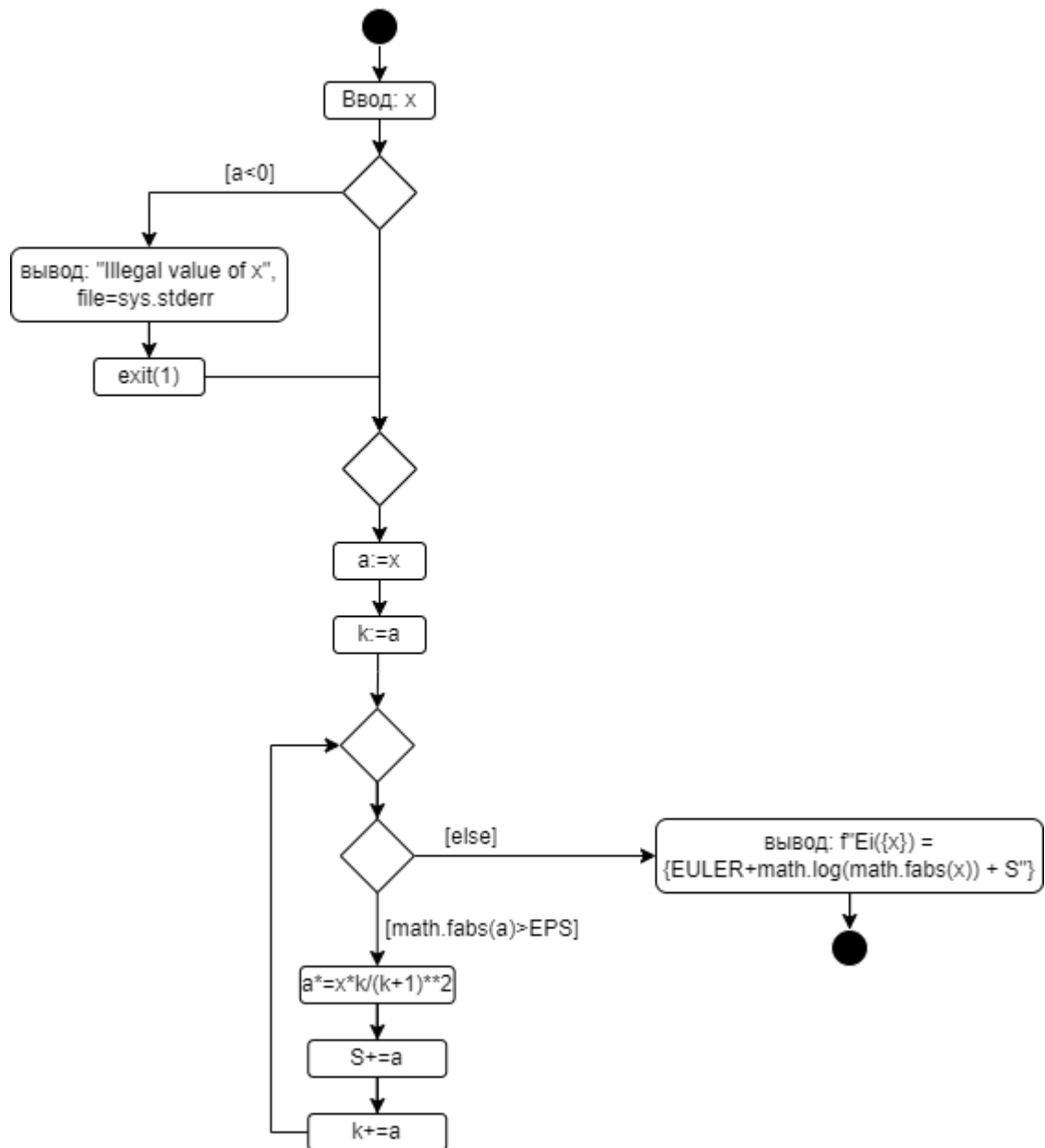


Рисунок 2.7 UML-диаграмма 5 примера

### 3. Индивидуальные задания:

- 1) Дано число  $m(1 \leq m \leq 12)$ . Определить полугодие, на которое приходится месяц с номером и количество дней в том месяце (год не високосный).

```
1 import sys
2 if __name__ == '__main__':
3     m = int(input('Введите номер месяца: '))
4
5     if m == 1:
6         print('Первое полугодие, 31 день')
7     elif m == 2:
8         print("первое полугодие, 28 дней")
9     elif m == 3:
10        print("первое полугодие, 31 день")
11    elif m == 4:
12        print("первое полугодие, 30 дней")
13    elif m == 5:
14        print("первое полугодие, 31 день")
15    elif m == 6:
16        print("первое полугодие, 30 дней")
17    elif m == 7:
18        print("второе полугодие, 31 день")
19    elif m == 8:
20        print("второе полугодие, 31 дней")
21    elif m == 9:
22        print("второе полугодие, 30 день")
23    elif m == 10:
24        print("второе полугодие, 31 дней")
25    elif m == 11:
26        print("второе полугодие, 30 день")
27    elif m == 12:
28        print("второе полугодие, 31 дней")
29    else:
30        print("Ошибка!", file=sys.stderr)
31        exit(1)
32
33 Run - main.py
34 Run: main ×
35 C:\Users\erken\Desktop\Универ\python\индивид\venv\Scripts\python.exe
36 Введите номер месяца: 12
37 второе полугодие, 31 дней
```

Рисунок 3.1 Программа к индивидуальному заданию 1



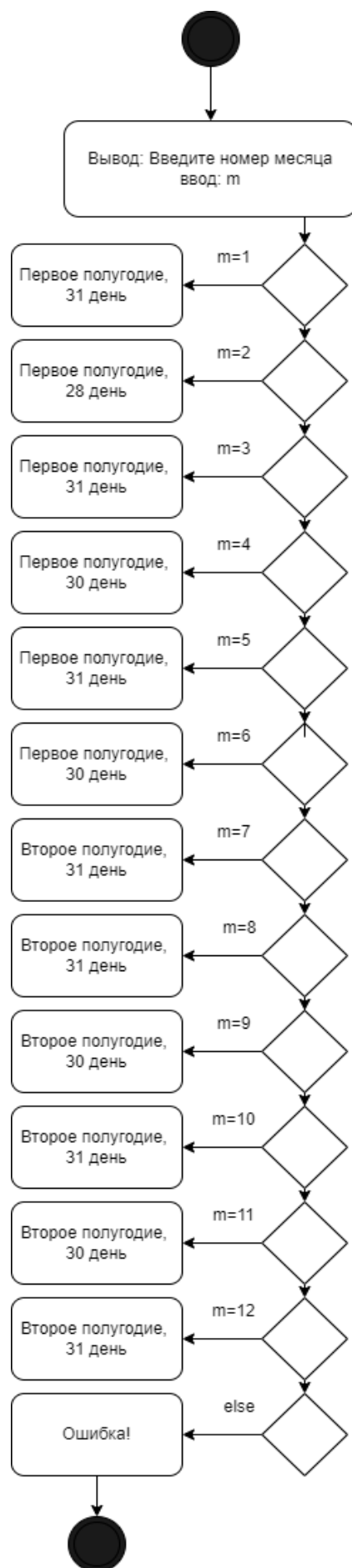


Рисунок 3.2 UML-диаграмма

2) Вывести на экран большее из трёх заданных чисел.

```
1 a = int(input('Введите первое число: '))
2 b = int(input('Введите второе число: '))
3 c = int(input('Введите третье число: '))
4 if a>b and a>c:
5     print("Наибольшее число: ", a)
6 elif b>a and b>c:
7     print("Наибольшее число: ", b)
8 elif c>a and c>b:
9     print("Наибольшее число: ", c)
10
```

Run - main.py

Run: индив\_2 ×

C:\Users\erken\Desktop\Универ\python\индивид\venv\Scripts\py

Введите первое число: 1  
Введите второе число: 2  
Введите третье число: 4  
Наибольшее число: 4

Рисунок 3.3 Код программы

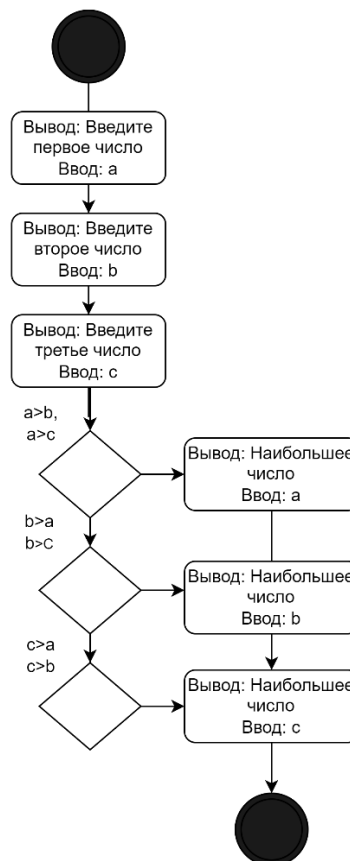
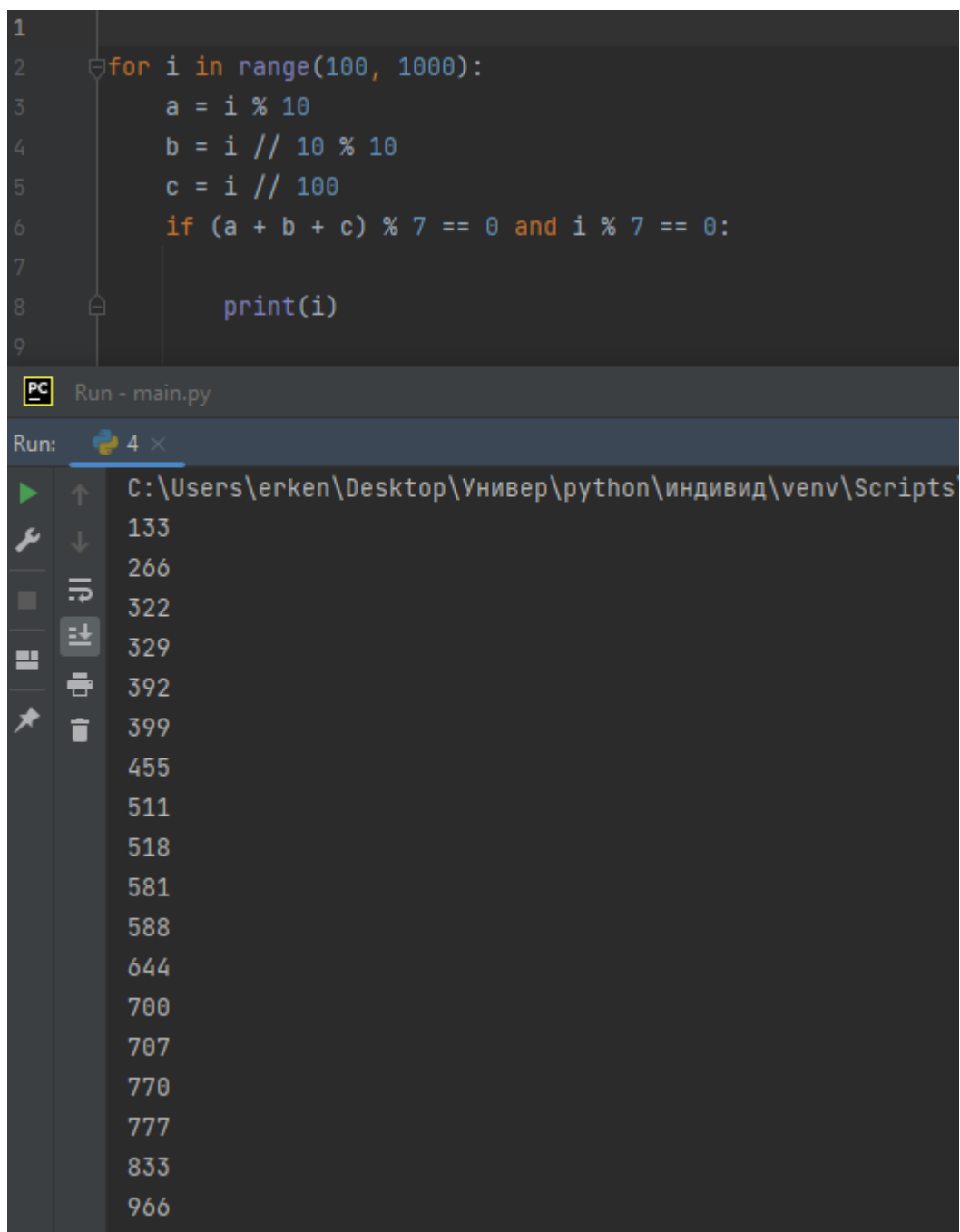


Рисунок 3.4 UML-диаграмма

3) Сумма цифр трехзначного числа кратна 7. Само число также делится на 7. Найти все такие числа.



The image shows a Python IDE with a dark theme. The editor window displays a Python script that iterates through numbers from 100 to 1000, checks if the sum of their digits is divisible by 7, and if the number itself is divisible by 7, then prints the number. Below the editor, the 'Run' console shows the output of the program, listing all such numbers.

```
1
2 for i in range(100, 1000):
3     a = i % 10
4     b = i // 10 % 10
5     c = i // 100
6     if (a + b + c) % 7 == 0 and i % 7 == 0:
7
8         print(i)
9
```

Run - main.py

Run: 4 x

C:\Users\erken\Desktop\Универ\python\индивид\venv\Scripts

133  
266  
322  
329  
392  
399  
455  
511  
518  
581  
588  
644  
700  
707  
770  
777  
833  
966

Рисунок 3.5 Код программы

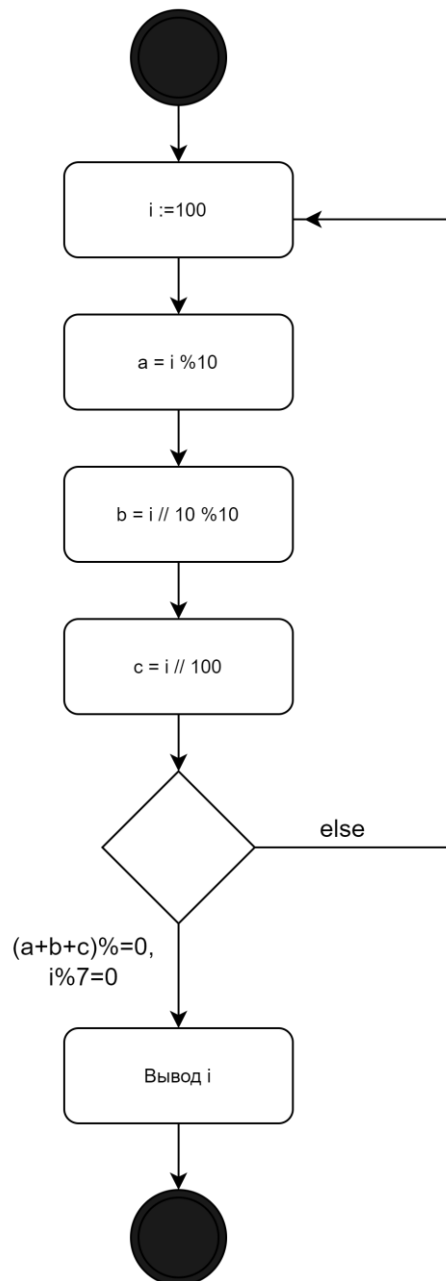


Рисунок 3.6 UML-диаграмма

4. Сделал коммит, выполнил слияние с веткой main, и запустил изменения в уд. репозиторий:

```
erken@LAPTOP-ESTC60GF MINGW64 ~/Desktop/Универ/Основы кросс/лаб2.2/1b2.2 (develop)
$ git commit -m "first com"
[develop 2292812] first com
8 files changed, 145 insertions(+)
create mode 100644 "\320\230\320\275\320\264\320\270\320\262\320\270\320\264\321\203\320\260\320\273\321\214\320\275\321\213\320\265\320\270\320\275\320\264\320\270\320\262_1.py"
create mode 100644 "\320\230\320\275\320\264\320\270\320\262\320\270\320\264\321\203\320\260\320\273\321\214\320\275\321\213\320\265\320\270\320\275\320\264\320\270\320\262_2.py"
create mode 100644 "\320\230\320\275\320\264\320\270\320\262\320\270\320\264\321\203\320\260\320\273\321\214\320\275\321\213\320\265\320\270\320\275\320\264\320\270\320\262_3.py"
create mode 100644 "\320\237\321\200\320\270\320\274\320\265\321\200\321\213\exercise_1.py"
create mode 100644 "\320\237\321\200\320\270\320\274\320\265\321\200\321\213\exercise_2.py"
create mode 100644 "\320\237\321\200\320\270\320\274\320\265\321\200\321\213\exercise_3.py"
create mode 100644 "\320\237\321\200\320\270\320\274\320\265\321\200\321\213\exercise_4.py"
create mode 100644 "\320\237\321\200\320\270\320\274\320\265\321\200\321\213\exercise_5.py"

erken@LAPTOP-ESTC60GF MINGW64 ~/Desktop/Универ/Основы кросс/лаб2.2/1b2.2 (develop)
$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

erken@LAPTOP-ESTC60GF MINGW64 ~/Desktop/Универ/Основы кросс/лаб2.2/1b2.2 (main)
$ git merge develop
Updating 7a46279..2292812
Fast-forward
.../\320\270\320\275\320\264\320\270\320\262_1.py | 33 ++++++
.../\320\270\320\275\320\264\320\270\320\262_2.py | 9 +++++
.../\320\270\320\275\320\264\320\270\320\262_3.py | 10 +++++
.../exercise_1.py | 15 ++++++
.../exercise_2.py | 18 ++++++
.../exercise_3.py | 14 ++++++
.../exercise_4.py | 19 ++++++
.../exercise_5.py | 27 ++++++
8 files changed, 145 insertions(+)
create mode 100644 "\320\230\320\275\320\264\320\270\320\262\320\270\320\264\321\203\320\260\320\273\321\214\320\275\321\213\320\265\320\270\320\275\320\264\320\270\320\262_1.py"
create mode 100644 "\320\230\320\275\320\264\320\270\320\262\320\270\320\264\321\203\320\260\320\273\321\214\320\275\321\213\320\265\320\270\320\275\320\264\320\270\320\262_2.py"
create mode 100644 "\320\230\320\275\320\264\320\270\320\262\320\270\320\264\321\203\320\260\320\273\321\214\320\275\321\213\320\265\320\270\320\275\320\264\320\270\320\262_3.py"
create mode 100644 "\320\237\321\200\320\270\320\274\320\265\321\200\321\213\exercise_1.py"
create mode 100644 "\320\237\321\200\320\270\320\274\320\265\321\200\321\213\exercise_2.py"
create mode 100644 "\320\237\321\200\320\270\320\274\320\265\321\200\321\213\exercise_3.py"
create mode 100644 "\320\237\321\200\320\270\320\274\320\265\321\200\321\213\exercise_4.py"
create mode 100644 "\320\237\321\200\320\270\320\274\320\265\321\200\321\213\exercise_5.py"

erken@LAPTOP-ESTC60GF MINGW64 ~/Desktop/Универ/Основы кросс/лаб2.2/1b2.2 (main)
$ git push
Enumerating objects: 13, done.
Counting objects: 100% (13/13), done.
Delta compression using up to 8 threads
Compressing objects: 100% (12/12), done.
Writing objects: 100% (12/12), 2.58 KiB | 880.00 KiB/s, done.
Total 12 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/umarkochkarov/1b2.2.git
7a46279..2292812 main -> main
```

Рисунок 4.1 Команды в консоли

main

1 branch

0 tags

Go to file

Add file

Code

umarkochkarov first com 2292812 3 minutes ago 3 commits

|                |                   |               |
|----------------|-------------------|---------------|
| Индивидуальные | first com         | 3 minutes ago |
| Примеры        | first com         | 3 minutes ago |
| .gitignore     | Update .gitignore | 5 days ago    |
| LICENSE        | Initial commit    | 5 days ago    |
| README.md      | Initial commit    | 5 days ago    |

Рисунок 4.2 Изменения на удаленном сервере

Ответы на контрольные вопросы

- 1. Для чего нужны диаграммы деятельности UML?  
Позволяет наглядно визуализировать алгоритм программы.
- 2. Что такое состояние действия и состояние деятельности?

Состояние действия - частный вид состояния деятельности, а конкретнее – такое состояние, которое не может быть подвергнуто дальнейшей декомпозиции.

Состояние деятельности можно представлять себе как составное состояние, поток управления которого включает только другие состояния деятельности и действий.

### **3. Какие нотации существуют для обозначения переходов и ветвлений в диаграммах деятельности?**

Переходы, ветвление, алгоритм разветвляющейся структуры, алгоритм циклической структуры.

### **4. Какой алгоритм является алгоритмом разветвляющейся структуры?**

Алгоритм разветвляющейся структуры - это алгоритм, в котором вычислительный процесс осуществляется либо по одной, либо по другой ветви, в зависимости от выполнения некоторого условия.

### **5. Чем отличается разветвляющийся алгоритм от линейного?**

Линейный алгоритм - алгоритм, все этапы которого выполняются однократно и строго последовательно.

Разветвляющийся алгоритм - алгоритм, содержащий хотя бы одно условие, в результате проверки которого ЭВМ обеспечивает переход на один из нескольких возможных шагов.

### **6. Что такое условный оператор? Какие существуют его формы?**

Оператор, конструкция языка программирования, обеспечивающая выполнение определённой команды (набора команд) только при условии истинности некоторого логического выражения, либо выполнение одной из нескольких команд.

Условный оператор имеет полную и краткую формы.

### **7. Какие операторы сравнения используются в Python?**

If, elif, else

### **8. Что называется простым условием? Приведите примеры.**

Простым условием называется выражение, составленное из двух арифметических выражений или двух текстовых величин.

Пример: `a == b`

### **9. Что такое составное условие? Приведите примеры.**

Составное условие – логическое выражение, содержащее несколько простых условий объединённых логическими операциями. Это операции `not`, `and`, `or`.

Пример: `(a == b or a == c)`

### **10. Какие логические операторы допускаются при составлении сложных условий?**

`not`, `and`, `or`.

**11. Может ли оператор ветвления содержать внутри себя другие ветвления?**

Может.

**12. Какой алгоритм является алгоритмом циклической структуры?**

Циклический алгоритм — это вид алгоритма, в процессе выполнения которого одно или несколько действий нужно повторить.

**13. Типы циклов в языке Python.**

В Python есть 2 типа циклов: - цикл while, - цикл for.

**14. Назовите назначение и способы применения функции range.**

Функция range генерирует серию целых чисел, от значения start до stop, указанного пользователем. Мы можем использовать его для цикла for и обходить весь диапазон как список.

**15. Как с помощью функции range организовать перебор значений от 15 до 0 с шагом 2?**

`range(15, 0, 2)`

**16. Могут ли быть циклы вложенными?**

Могут.

**17. Как образуется бесконечный цикл и как выйти из него?**

Бесконечный цикл в программировании — цикл, написанный таким образом, что условие выхода из него никогда не выполняется.

**18. Для чего нужен оператор break?**

Используется для выхода из цикла.

**19. Где употребляется оператор continue и для чего он используется?**

Оператор continue используется только в циклах. В операторах for , while , do while , оператор continue выполняет пропуск оставшейся части кода тела цикла и переходит к следующей итерации цикла.

**20. Для чего нужны стандартные потоки stdout и stderr?**

Ввод и вывод распределяется между тремя стандартными потоками: stdin — стандартный ввод (клавиатура), stdout — стандартный вывод (экран), stderr — стандартная ошибка (вывод ошибок на экран)

**21. Как в Python организовать вывод в стандартный поток stderr?**

Указать в `print(..., file=sys.stderr)`.

**22. Каково назначение функции exit?**

Функция `exit()` модуля `sys` - выход из Python.