

Министерство науки и высшего образования Российской Федерации  
Федеральное государственное автономное образовательное учреждение  
высшего образования  
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития  
Кафедра инфокоммуникаций

**ОТЧЁТ**  
**ПО ЛАБОРАТОРНОЙ РАБОТЕ №2.2**  
**дисциплины**  
**«Основы кроссплатформенного программирования»**

Выполнил:  
Костукайло Кирилл Николаевич  
1 курс, группа ИВТ-б-о-21-1,  
09.03.01 «Информатика и  
вычислительная техника»,  
направленность (профиль)  
«Программное обеспечение средств  
вычислительной техники и  
автоматизированных систем», очная  
форма обучения

---

(подпись)

Проверил:  
Кафедры инфокоммуникаций, старший  
преподаватель  
Воронкин Р.А.

---

(подпись)

Отчет защищен с оценкой \_\_\_\_\_ Дата защиты \_\_\_\_\_

Ставрополь, 2022 г.

## Тема: Условные операторы и циклы в языке Python

Цель работы: приобретение навыков программирования разветвляющихся алгоритмов и алгоритмов циклической структуры. Освоить операторы языка Python версии 3.x if, while, for, break и continue, позволяющих реализовывать разветвляющиеся алгоритмы и алгоритмы циклической структуры.

Порядок выполнения работы:

1. Создать общедоступный репозиторий на GitHub, в котором будет использована лицензия MIT и язык программирования Python.
2. Выполните клонирование созданного репозитория.
3. Дополните файл .gitignore необходимыми правилами для работы с IDE PyCharm.
4. Организуйте свой репозиторий в соответствие с моделью ветвления git-flow.
5. Самостоятельно изучите рекомендации к оформлению исходного кода на языке Python PEP-8 (<https://pep8.org/>). Выполните оформление исходного примеров лабораторной работы и индивидуальных созданий в соответствие с PEP-8.
6. Создайте проект PyCharm в папке репозитория.
7. Проработайте примеры лабораторной работы. Создайте для каждого примера отдельный модуль языка Python. Зафиксируйте изменения в репозитории.

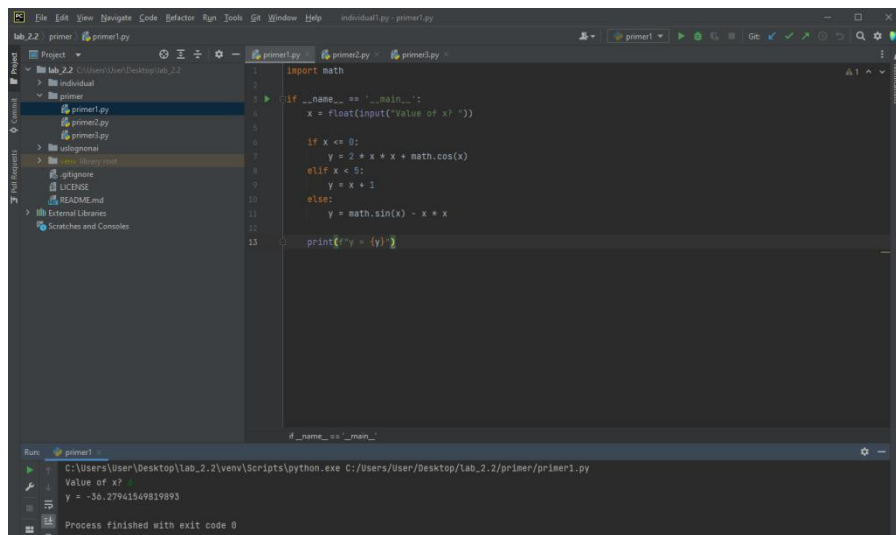


Рисунок 1.1. Пример 1

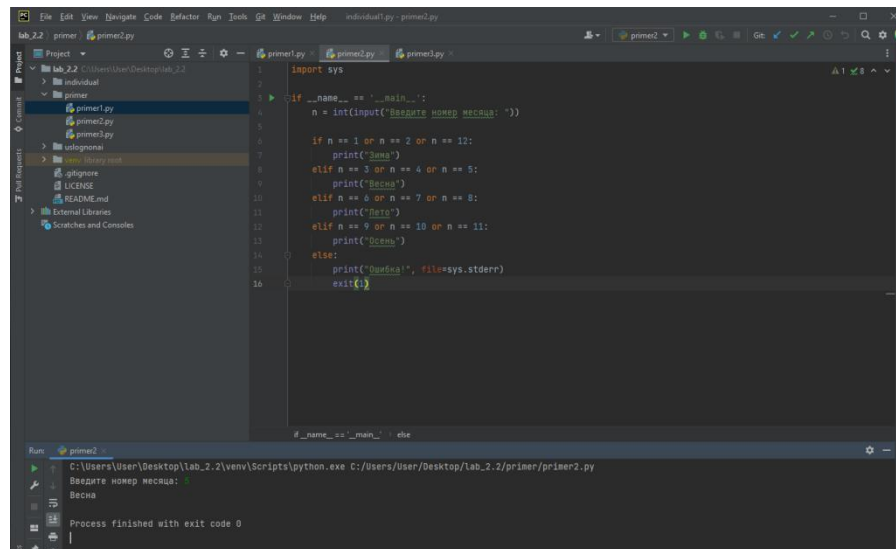


Рисунок 1.2. Пример 2

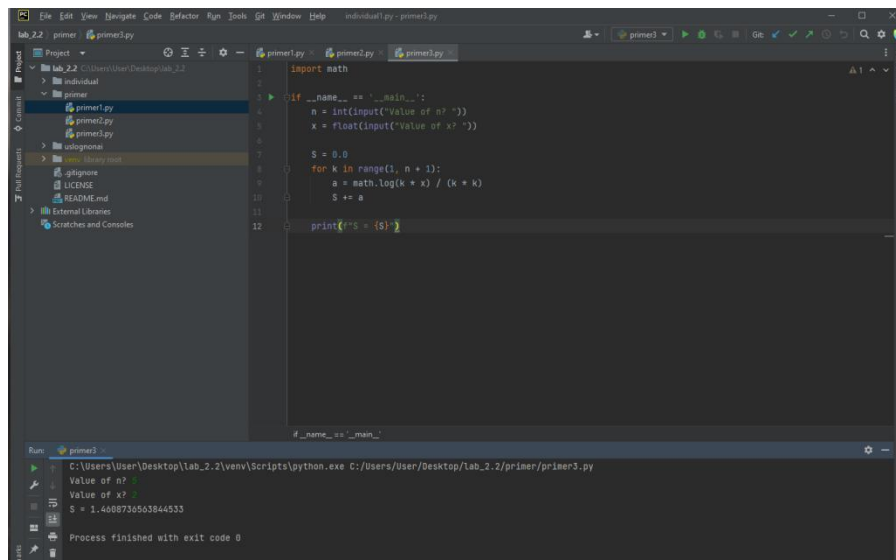
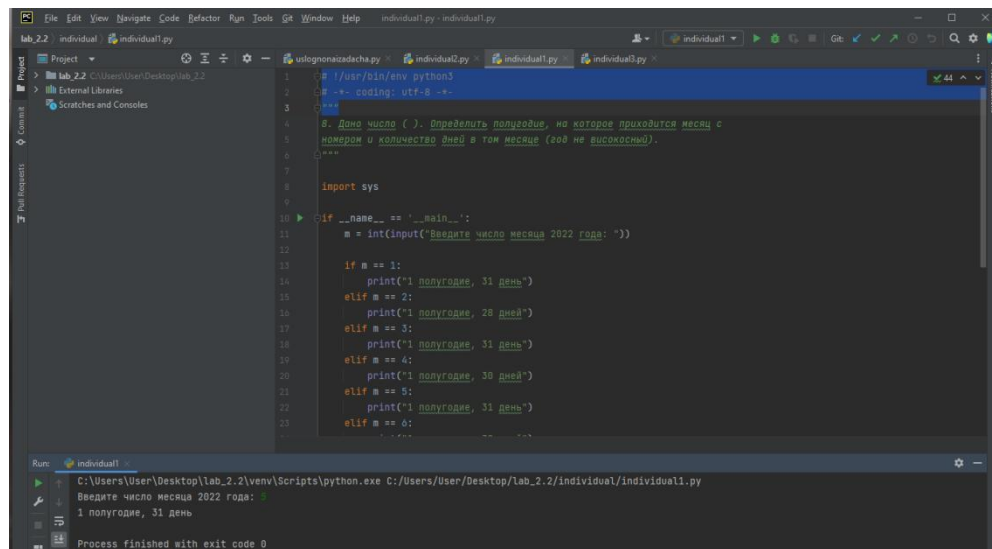


Рисунок 1.3. Пример 3

8. Приведите в отчёте скриншоты результатов выполнения каждой из программ примеров при различных исходных данных вводимых с клавиатуры.
9. Выполните индивидуальные задания, согласно своего варианта. Для заданий повышенной сложности номер варианта должен быть получен у преподавателя.



```
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3 """
4 8. Дано число (.). Определить полугодие, на которое приходится месяц с
5 номером и количеством дней в том месяце (202 не високосный).
6 """
7
8 import sys
9
10 if __name__ == '__main__':
11     m = int(input("Введите число месяца 2022 года: "))
12
13     if m == 1:
14         print("1 полугодие, 31 день")
15     elif m == 2:
16         print("1 полугодие, 28 дней")
17     elif m == 3:
18         print("1 полугодие, 31 день")
19     elif m == 4:
20         print("1 полугодие, 30 дней")
21     elif m == 5:
22         print("1 полугодие, 31 день")
23     elif m == 6:
24         print("1 полугодие, 31 день")
25     else:
26         print("Неверный номер месяца")
27
28 # _name__ == '__main__' : else
```

Run: individual1 -

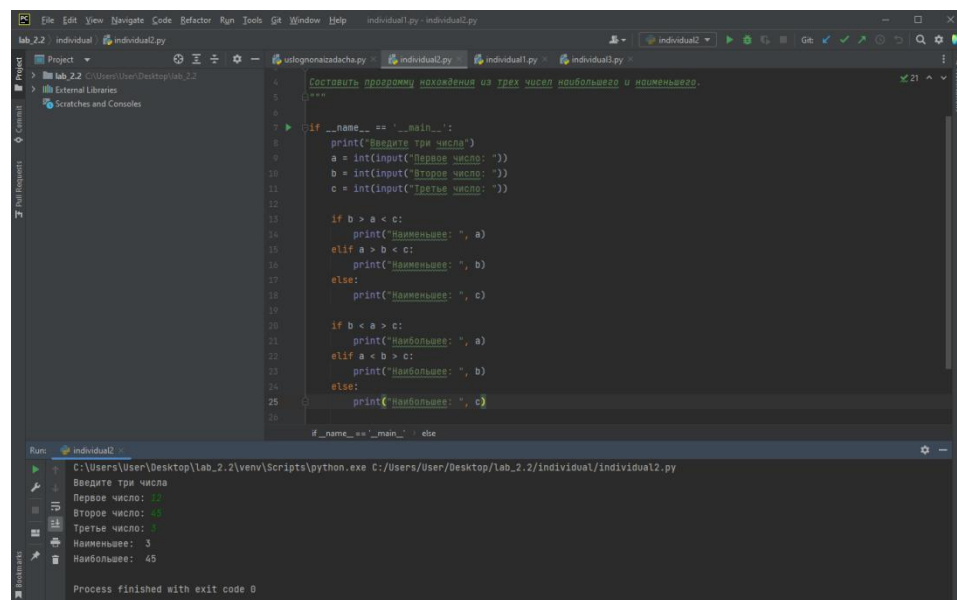
C:\Users\User\Desktop\lab\_2.2\venv\Scripts\python.exe C:/Users/User/Desktop/lab\_2.2/individual/individual1.py

Введите число месяца 2022 года: 1

1 полугодие, 31 день

Process finished with exit code 0

Рисунок 2.1. Индивидуальное задание 1



```
1 """
2 Составить программу нахождения из трех чисел наибольшего и наименьшего.
3 """
4
5 if __name__ == '__main__':
6     print("Введите три числа")
7     a = int(input("Первое число: "))
8     b = int(input("Второе число: "))
9     c = int(input("Третье число: "))
10
11     if b > a < c:
12         print("Наименьшее: ", a)
13     elif a > b < c:
14         print("Наименьшее: ", b)
15     else:
16         print("Наименьшее: ", c)
17
18     if b < a > c:
19         print("Наибольшее: ", a)
20     elif a < b > c:
21         print("Наибольшее: ", b)
22     else:
23         print("Наибольшее: ", c)
24
25 # _name__ == '__main__' : else
```

Run: individual2 -

C:\Users\User\Desktop\lab\_2.2\venv\Scripts\python.exe C:/Users/User/Desktop/lab\_2.2/individual/individual2.py

Введите три числа

Первое число: 12

Второе число: 45

Третье число: 3

Наименьшее: 3

Наибольшее: 45

Process finished with exit code 0

Рисунок 2.2. Индивидуальное задание 2

```
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 #. Сумма цифр трехзначного числа кратна 7. Само число также делится на 7. Найти все такие числа.
5
6
7 if __name__ == '__main__':
8     for i in range(105, 701, 7):
9         d, u = divmod(i, 10)
10        h, d = divmod(d, 10)
11        if h + d + u == 7:
12            print(i)
13
```

Run: C:\Users\User\Desktop\lab\_2.2\venv\Scripts\python.exe C:\Users\User\Desktop\lab\_2.2\Individual\individual3.py

135  
322  
511  
700

Process finished with exit code 0

Рисунок 2.3. Индивидуальное задание 3

```
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3 #. Найти интеграл вероятности.
4
5 import sys
6
7 EPS = 10 ** -10
8
9
10 if __name__ == '__main__':
11     x = float(input("add value for x: "))
12     if x == 0:
13         print("Illegal value of x", file=sys.stderr)
14         exit(1)
15
16     a = x
17     S, n = a, 0
18
19     while math.fabs(a) > EPS:
20         a += ((-1) * x ** 2 * (2 * n + 1)) / ((2 * n + 3) * (n + 1))
21         S += a
22         n += 1
23
24     print(f"erf({x}) = {(2 / math.sqrt(math.pi)) * S}")
25
26 # __name__ == '__main__'
```

Run: C:\Users\User\Desktop\lab\_2.2\venv\Scripts\python.exe C:\Users\User\Desktop\lab\_2.2\uslognonai\uslognonaizadacha.py

add value for x: 3  
erf(3.0) = 0.9999779094967672

Process finished with exit code 0

Рисунок 2.3. Задача повышенной сложности

10. Приведите в отчете скриншоты работы программ и UML-диаграммы деятельности решения индивидуальных заданий.

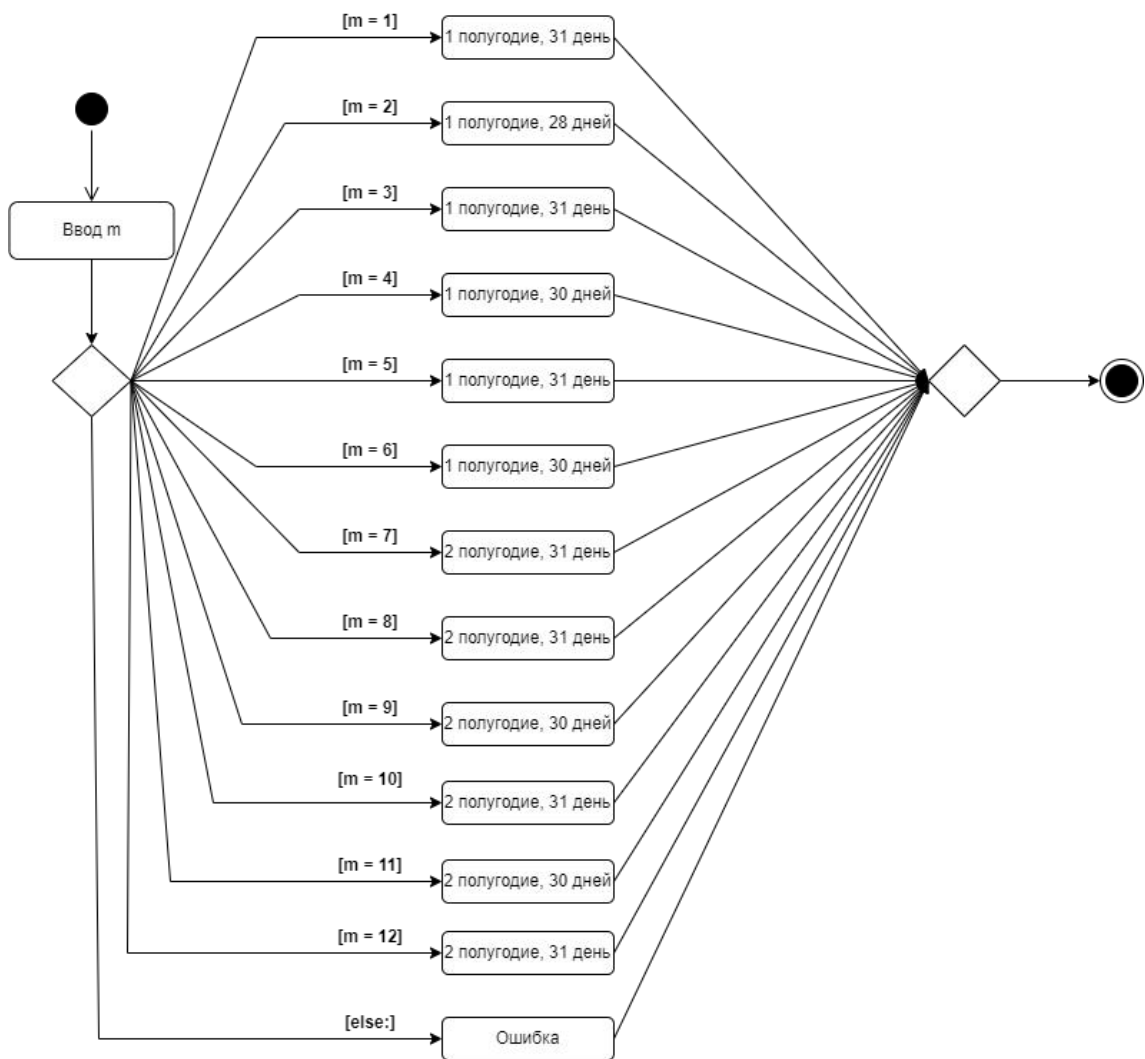


Рисунок 3.1. UML-диаграмма индивидуального задания 1

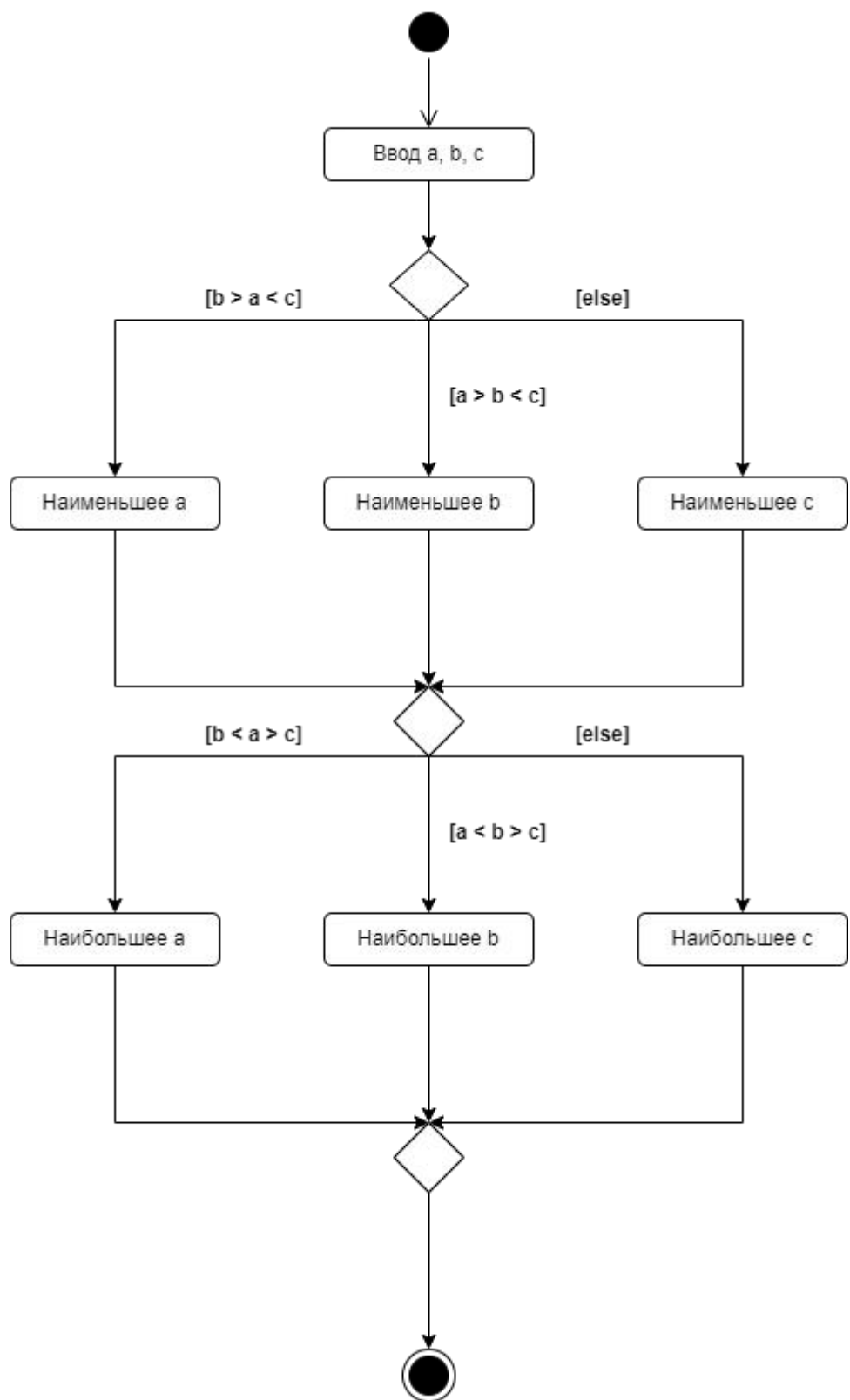


Рисунок 3.2. UML-диаграмма индивидуального задания 2

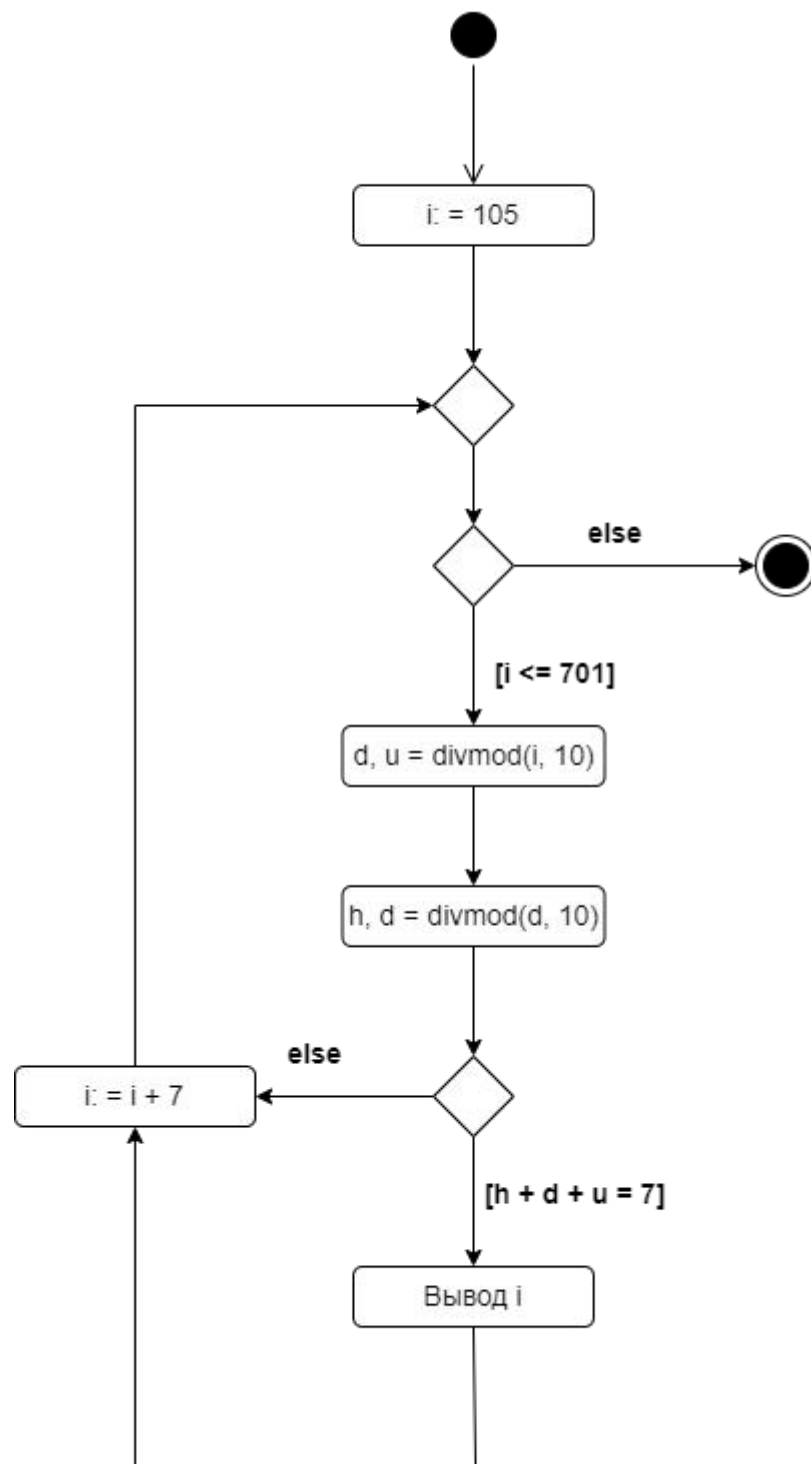


Рисунок 3.3. UML-диаграмма индивидуального задания 3



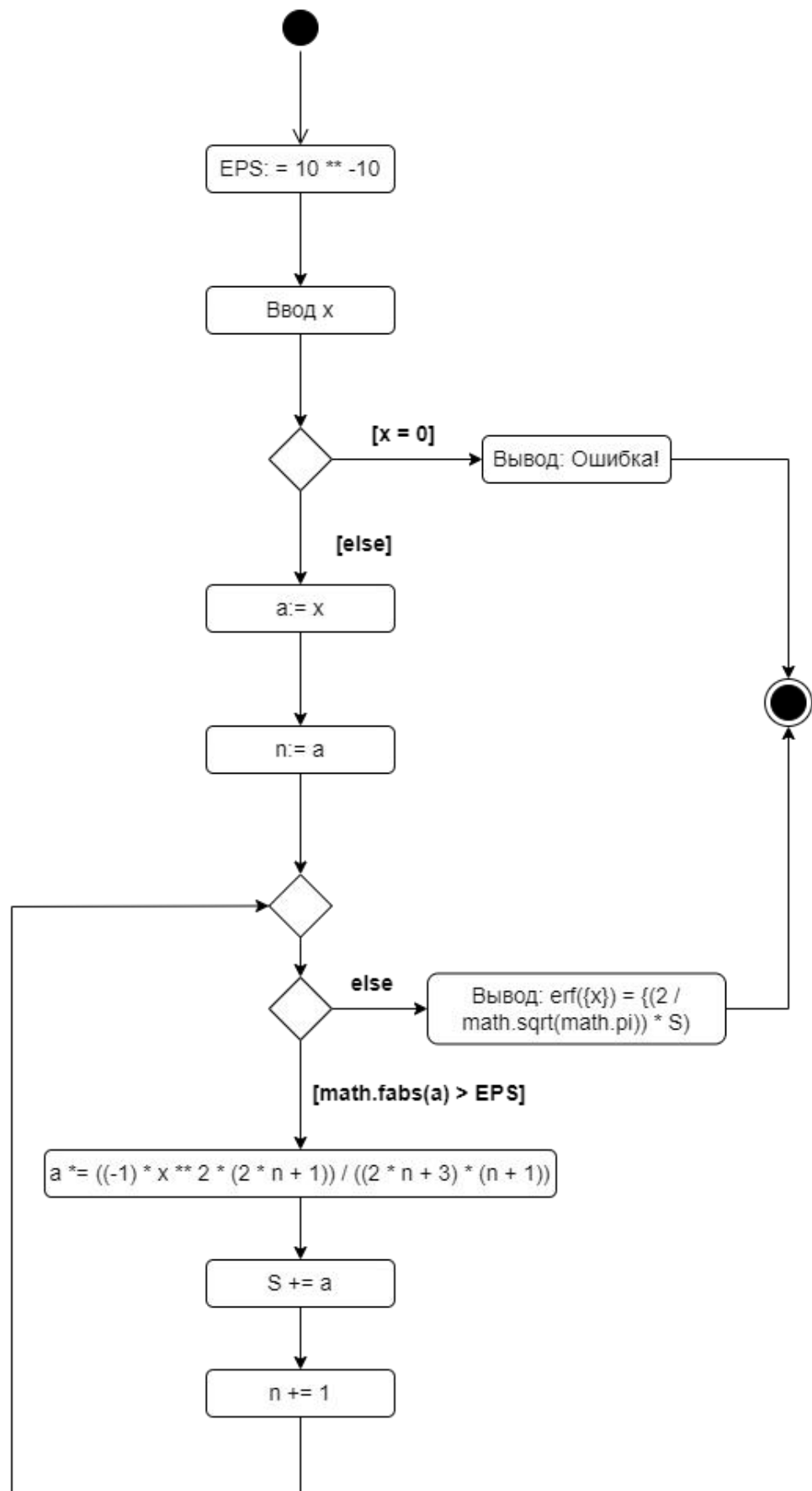


Рисунок 3.4. UML-диаграмма задачи повышенной сложности

11. Зафиксируйте сделанные изменения в репозитории.

## **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 представить 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 объединенных 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.

**Вывод:** приобрёл навыки программирования разветвляющихся алгоритмов и алгоритмов циклической структуры. Освоил операторы языка Python версии 3.x if, while, for, break и continue, позволяющих реализовывать разветвляющиеся алгоритмы и алгоритмы циклической структуры.