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

Институт перспективной инженерии
Департамент цифровых робототехнических систем и электроники

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №3
дисциплины «Программирование на Python»

Выполнил:
Щегольков Савва Игоревич
2 курс, группа ИВТ-б-о-24-1,
09.03.01 «Информатика и
вычислительная техника»,
направленность (профиль)
«Автоматизированные системы
обработки информации и
управления», очная форма обучения

_____
(подпись)

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2025 г.

Тема: Условные операторы и циклы в языке Python.

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

Порядок выполнения работы:

Ссылка на репозиторий: https://github.com/xouixao/lab3

В соответствии с заданием были созданы UML-диаграммы к примерам 4 и 5

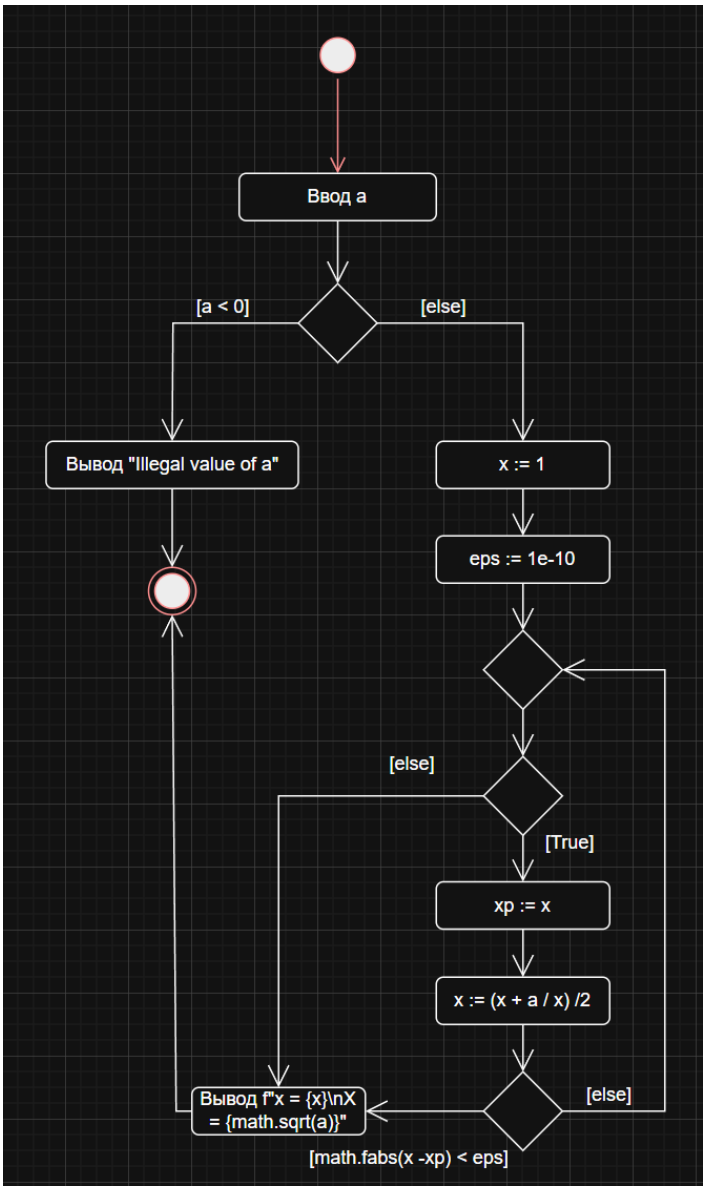

Рисунок 1. UML-диаграмма к примеру 4

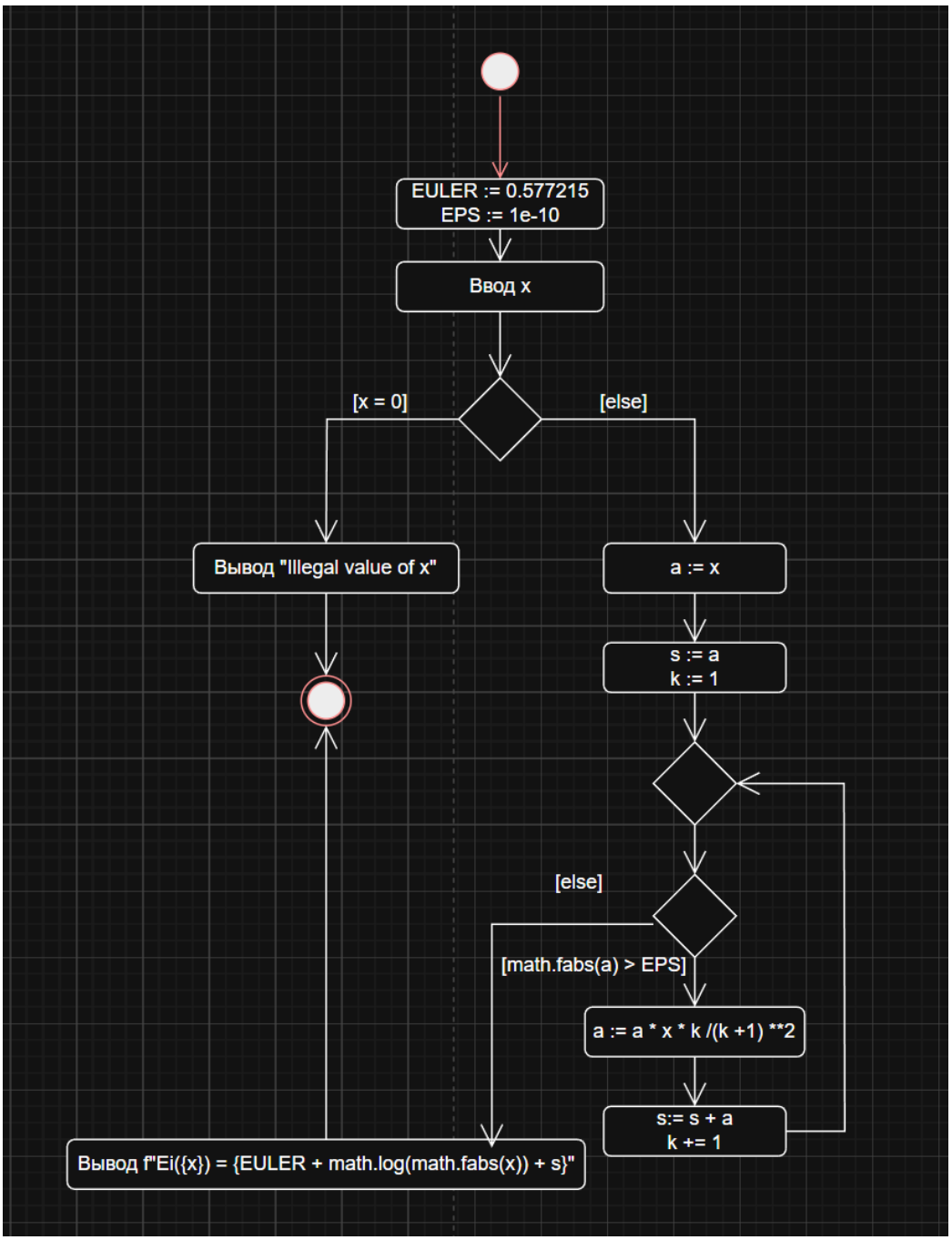

Рисунок 2. UML-диаграмма к примеру 5

Задание 1 заключается в выводе количества дней в месяце с номером вводимым с консоли

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
if __name__ == '__main__':
    x = int(input())
    match x:
        case 1 | 3 | 5 | 7 | 8 | 10 | 12:
            print(31)
        case 2:
            print(28)
        case _:
            print(30)
```

```
Run  1 ×

C:\Users\xumuk\AppData\Local\Programs\Python\Python311\python.exe C:
2
28

Process finished with exit code 0
```

Рисунок 3. Задание 1

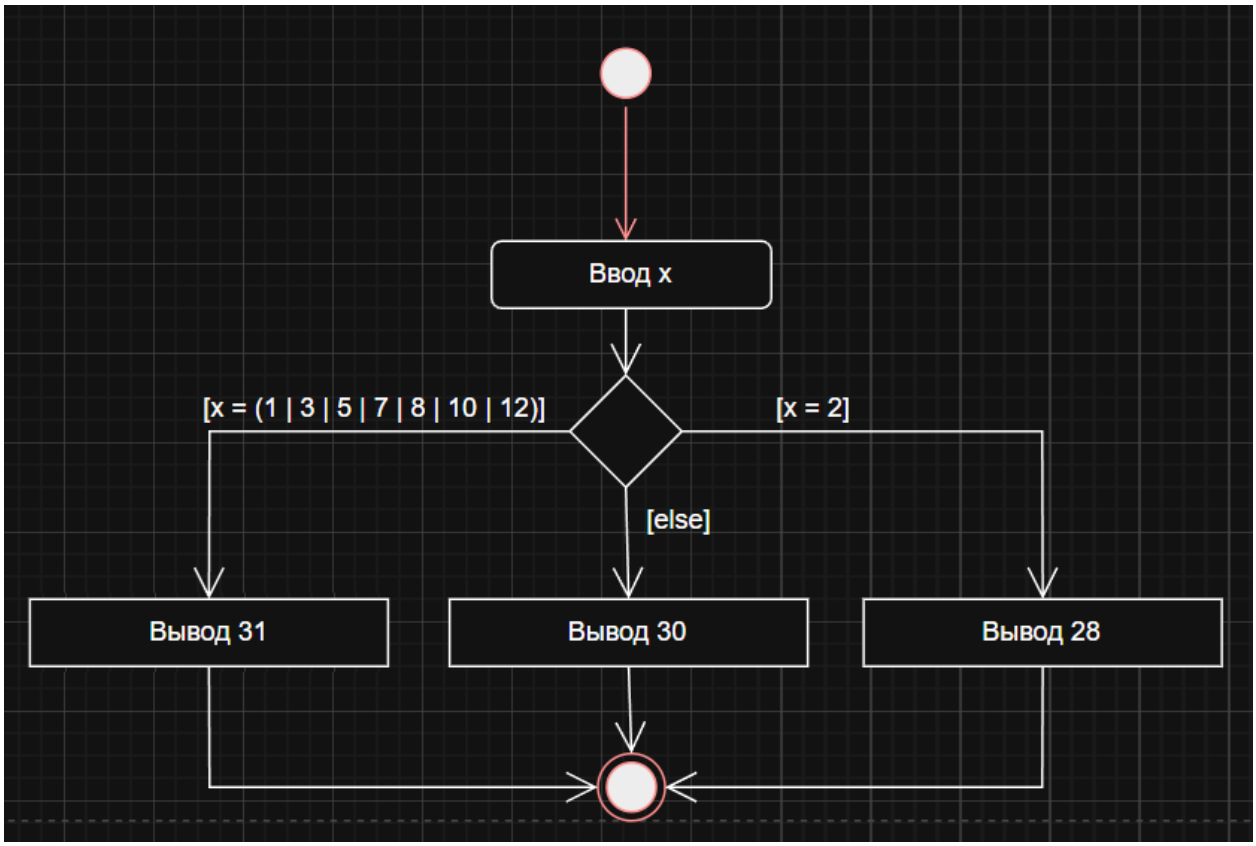

Рисунок 4. UML-диаграмма к заданию 1

2. Даны действительные числа x и y. Найти $U = \max^2(x^2\,y, x\,y^2) + \min^2(x - y, x + 2y)$. Для минимума и максимума использовать условный оператор `if`.

Рисунок 5. Условие задания 2.

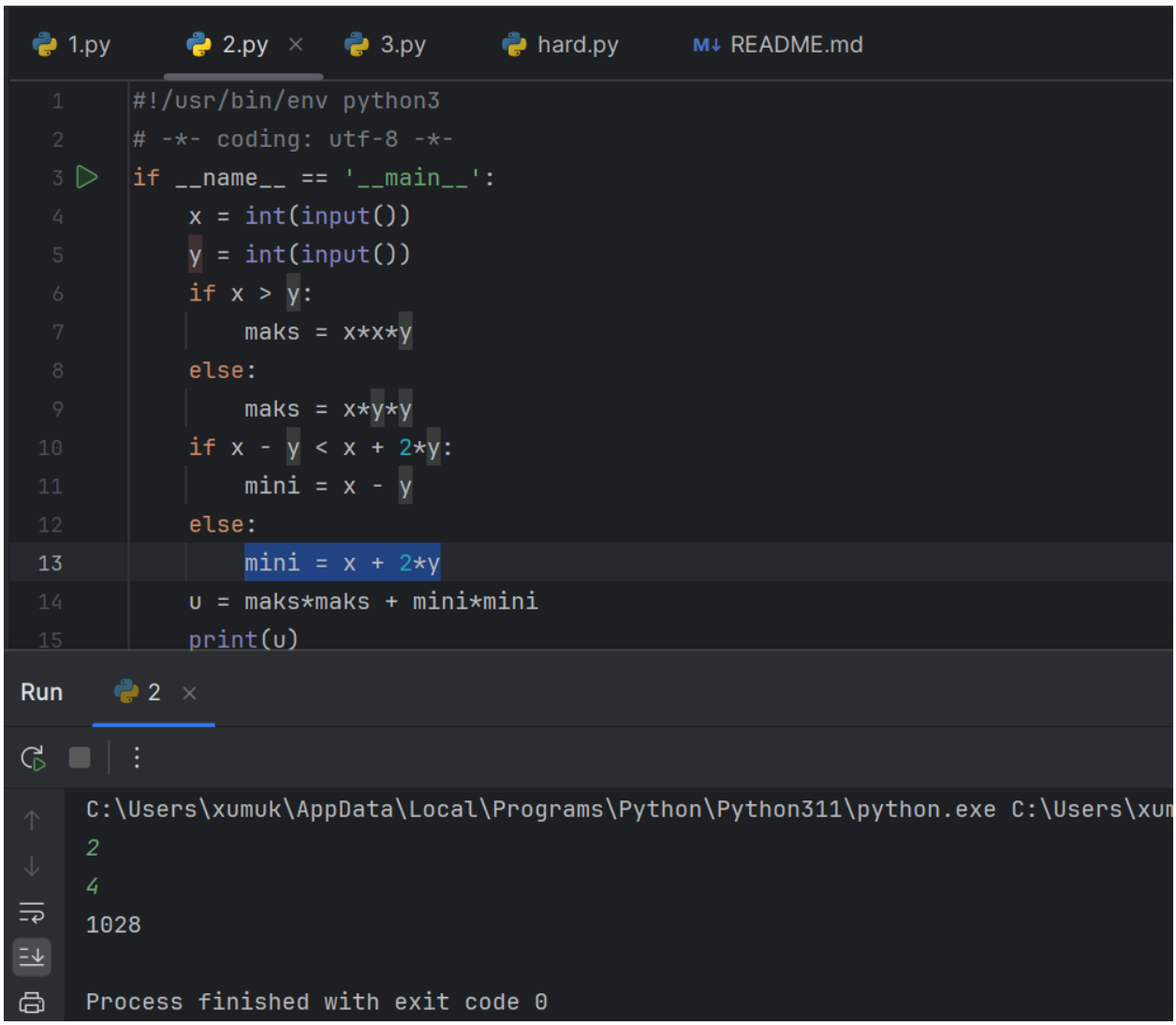

Рисунок 6. Задание 2.

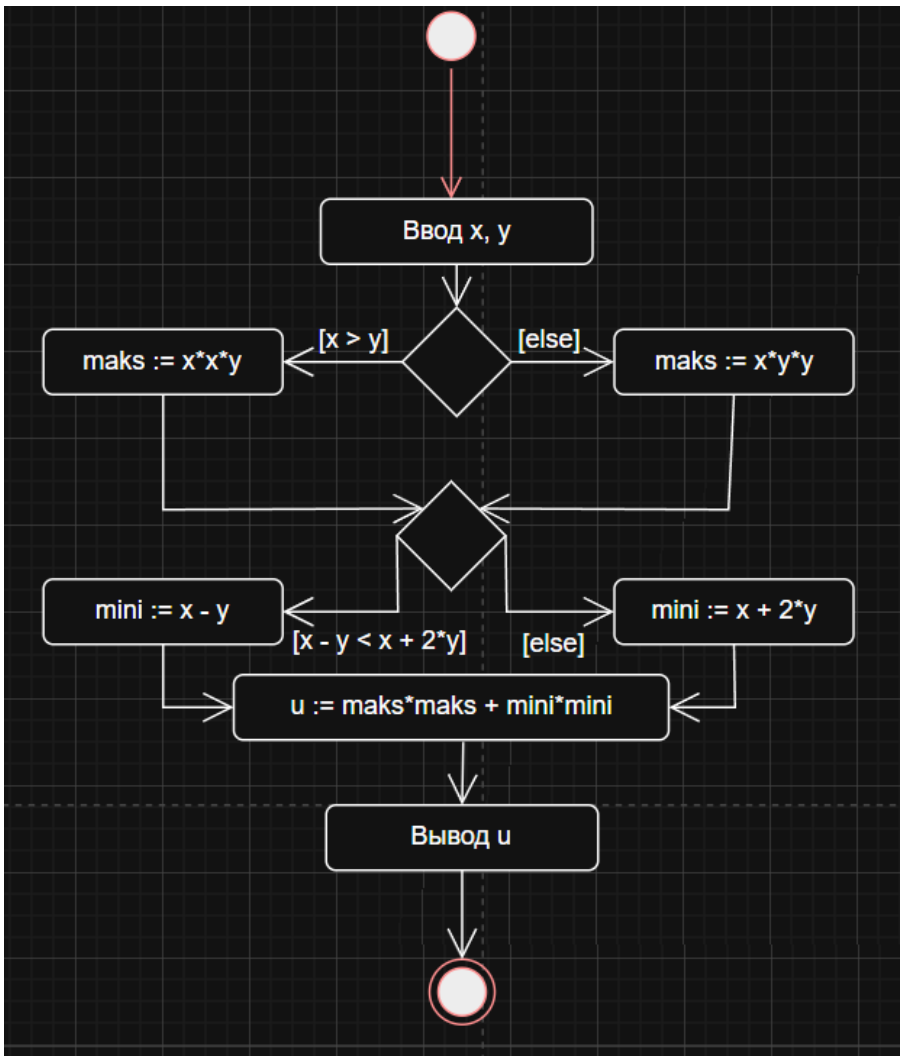

Рисунок 7. UML-диаграмма к заданию 2

Задание 3 заключается в нахождении суммы положительных чисел от 20 до 100, кратных 3.

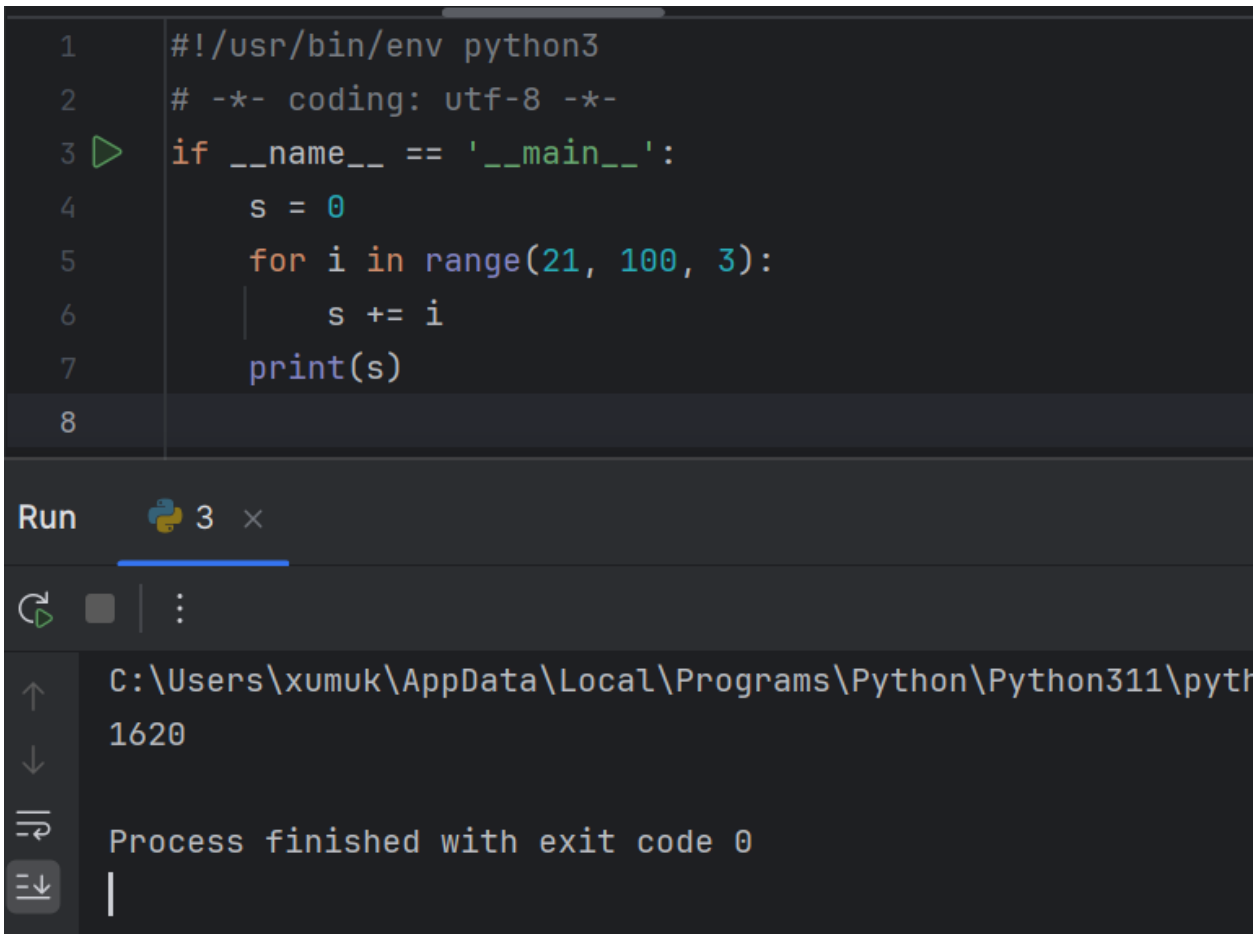

Рисунок 8 – Задание 3.

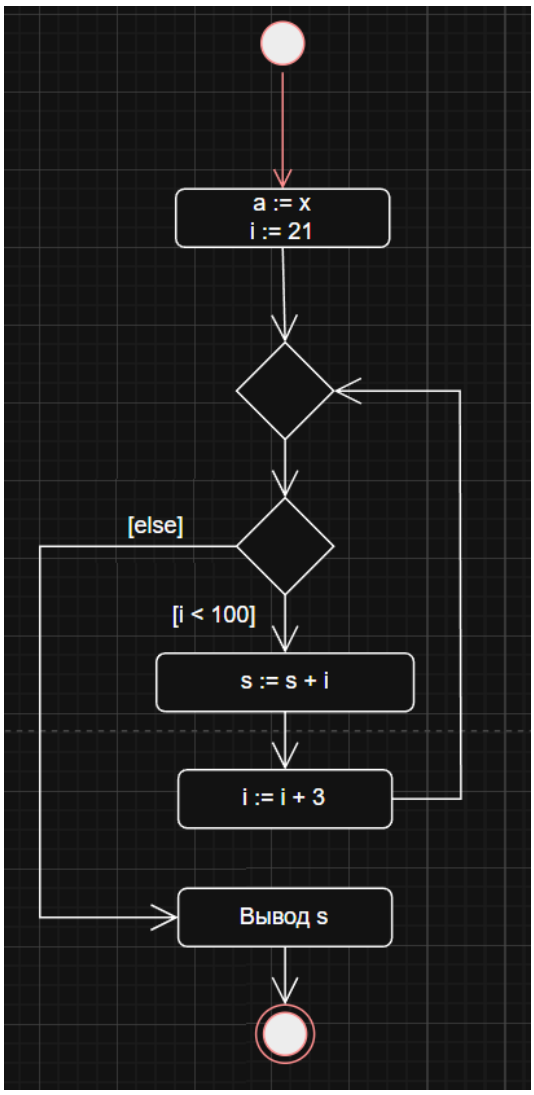

Рисунок 9. UML-диаграмма к заданию 3

Задание повышенной сложности заключается в том, чтобы найти значение функции Бесселя первого рода, где значение n и x вводится с клавиатуры

$$J_n(x) = \left(\frac{x}{2}\right)^n \sum_{k=0}^{\infty} \frac{(-x^2/4)^k}{k!(k+n)!}.$$

Рисунок 10. Формула функции

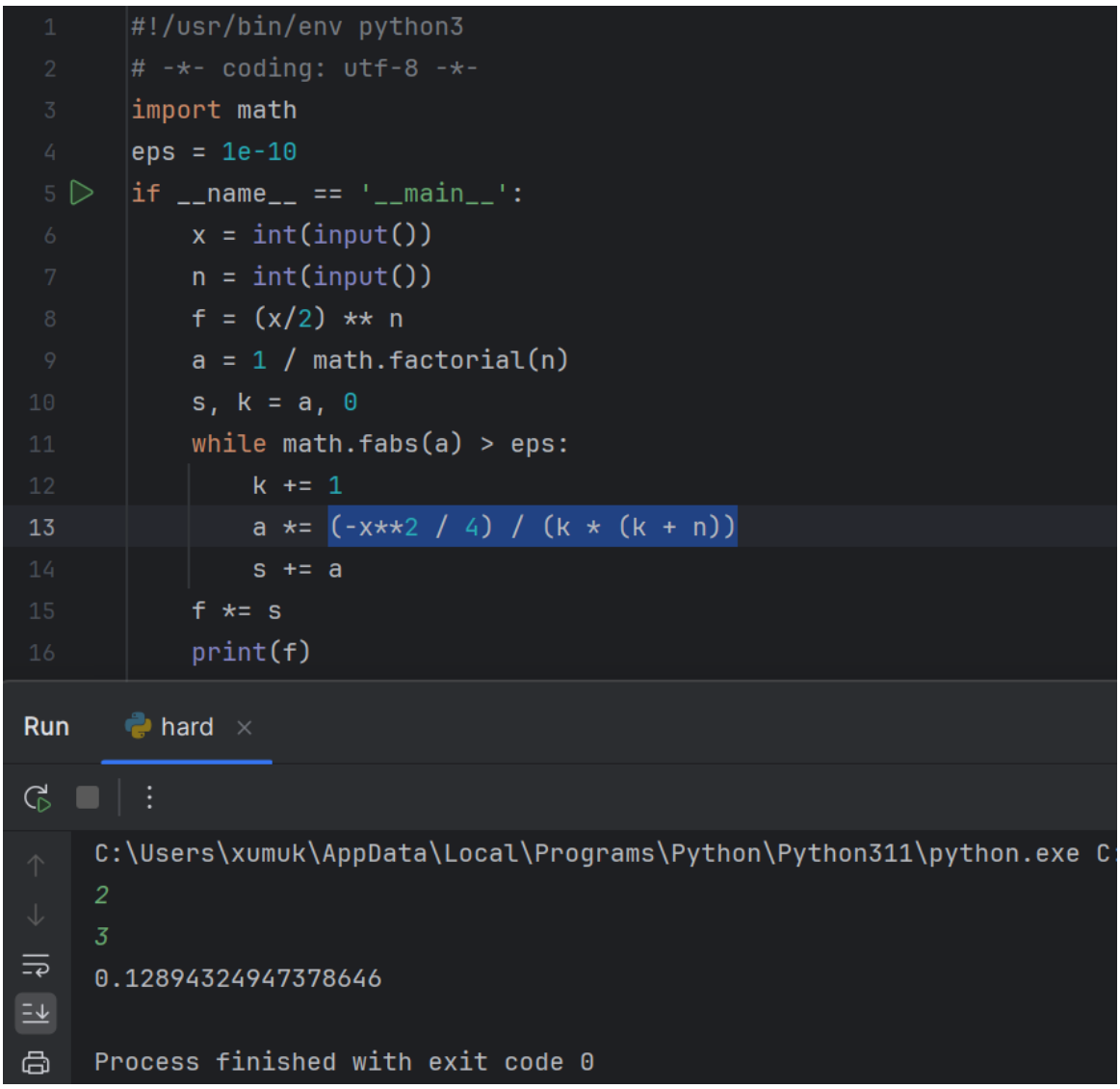

Рисунок 11. Задание повышенной сложности.

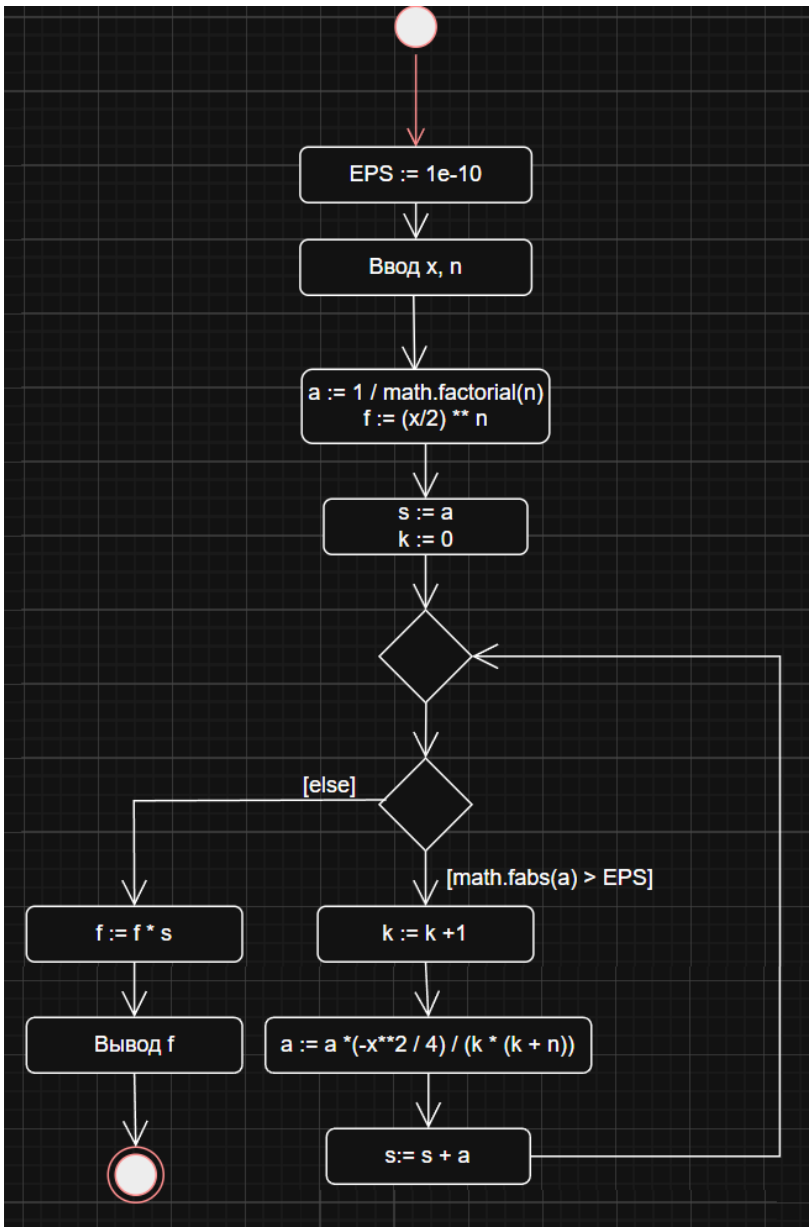

Рисунок 12. UML-диаграмма к заданию повышенной сложности

Контрольные вопросы:

1. Диаграммы деятельности UML нужны для наглядного описания алгоритмов и бизнес-процессов: они показывают последовательность действий, переходы между ними, ветвления и параллельные ветви, что упрощает анализ и проектирование программ и систем.

2. Состояние действия (action) — это атомарный, быстро выполняемый шаг алгоритма, который нельзя логично разложить дальше, а состояние деятельности (activity) — более крупный фрагмент работы, состоящий из нескольких действий и/или внутренних диаграмм деятельности.

3. В диаграммах деятельности переходы изображаются стрелками (дугами потока управления), ветвления и слияния — ромбами (узлы решения/слияния), параллельное разветвление и объединение — толстыми полосками (fork/join), а условия на переходах записываются в квадратных скобках рядом со стрелкой.

4. Алгоритмом разветвляющейся структуры является такой алгоритм, в котором дальнейший ход выполнения выбирается из двух или более вариантов в зависимости от условия, то есть алгоритмы с операторами ветвления (if, if…else, switch и т.п.).

5. Линейный алгоритм выполняет все действия строго по порядку без выбора пути, а разветвляющийся алгоритм содержит проверки условий и может переходить по разным ветвям, поэтому его результат и последовательность шагов зависят от исходных данных.

6. Условный оператор — это оператор, который в зависимости от истинности логического выражения выбирает, какой фрагмент кода выполнить; в Python есть формы if (неполная), if…else (полная), if…elif…else (много вариантов) и тернарное условное выражение A if cond else B.

7. В Python основные операторы сравнения: == (равно), != (не равно), > (больше), < (меньше), >= (больше или равно), <= (меньше или равно); также существуют специальные операторы сравнения идентичности is и is not.

8. Простым условием называют логическое выражение без логических связок, обычно одну проверку сравнения или булеву переменную, например: x > 0, a == b, flag, n != 10.

9. Составное (сложное) условие — это выражение, состоящее из нескольких простых условий, соединённых логическими операторами and, or, not, например: x > 0 and x < 10, (a > b) or (c == 0), not done and (err == 0).

10. При составлении сложных условий в Python используются логические операторы and (логическое И), or (логическое ИЛИ) и not (логическое НЕ), которые позволяют комбинировать простые условия в одно выражение.

11. Да, оператор ветвления может содержать внутри себя другие ветвления: в Python это реализуется через вложенные конструкции if — if внутри if или if внутри ветви else и т.д.

12. Алгоритмом циклической структуры является алгоритм, в котором некоторые действия выполняются многократно (повторяются) пока истинно заданное условие или пока перебираются элементы набора, то есть алгоритмы, реализуемые циклами.

13. В языке Python основной набор типов циклов — это цикл while (повторение, пока истинно условие) и цикл for (перебор элементов итерируемого объекта или диапазона значений).

14. Функция range используется для генерации последовательностей целых чисел, чаще всего для организации циклов for; она может вызываться в формах range(stop), range(start, stop) и range(start, stop, step) и позволяет удобно перебирать индексы, числа в диапазоне или шагать с заданным шагом.

15. Перебор значений от 15 до 0 с шагом 2 можно организовать так: for i in range(15, -1, -2):, где 15 — начало, -1 — значение, до которого не доходят (поэтому фактически до 0 включительно), а -2 — шаг уменьшения.

16. Да, циклы могут быть вложенными, то есть внутри тела одного цикла for или while могут находиться другие циклы, которые выполняются для каждой итерации внешнего цикла.

17. Бесконечный цикл образуется, когда условие выхода никогда не становится ложным или вообще отсутствует, например while True: без break; выйти из такого цикла можно с помощью оператора break, операторов return/raise в функции или прерыванием программы (например, Ctrl+C).

18. Оператор break нужен для немедленного прекращения выполнения ближайшего вложенного цикла (for или while) при наступлении некоторого условия, после чего управление передаётся на первую строку кода после этого цикла.

19. Оператор continue используется внутри циклов for и while, чтобы пропустить оставшуюся часть тела цикла для текущей итерации и сразу перейти к проверке условия и началу следующей итерации.

20. Стандартные потоки stdout и stderr нужны для вывода текстовой информации: stdout — для обычного результата работы программы, а stderr — для сообщений об ошибках и диагностики, причём в операционных системах их можно перенаправлять раздельно.

21. В Python вывод в стандартный поток stderr можно организовать через модуль sys, например: import sys и затем print("сообщение", file=sys.stderr) или использовать sys.stderr.write("сообщение\n").

22. Функция (или вызов) exit/sys.exit() предназначена для завершения работы программы: она прерывает выполнение кода, может вернуть код завершения операционной системе и корректно завершает интерпретатор (закрывает ресурсы, выходит из всех уровней вызовов).

Вывод: в ходе выполнения лабораторной работы были приобретены навыки программирования разветвляющихся алгоритмов и алгоритмов циклической структуры, освоены операторы языка Python версии 3.x if, while, for, break, continue, позволяющие реализовывать разветвляющиеся алгоритмы и алгоритмы циклической структуры.