

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»
Вариант 19**

Выполнил:
Поляков Никита Александрович
2 курс, группа ИВТ-б-о-24-1,
09.03.01 «Информатика и
вычислительная техника»,
направленность (профиль)
«Программное обеспечение средств
вычислительной техники и
автоматизированных систем», очная
форма обучения

(подпись)

Руководитель практики:
Воронкин Р.А., доцент департамента
цифровых, робототехнических систем и
электроники института перспективной
инженерии

(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 и continue, позволяющих реализовывать разветвляющиеся алгоритмы и алгоритмы циклической структуры.

Практическая часть:

Перед началом работы был создан новый репозиторий для лабораторной работы №3:

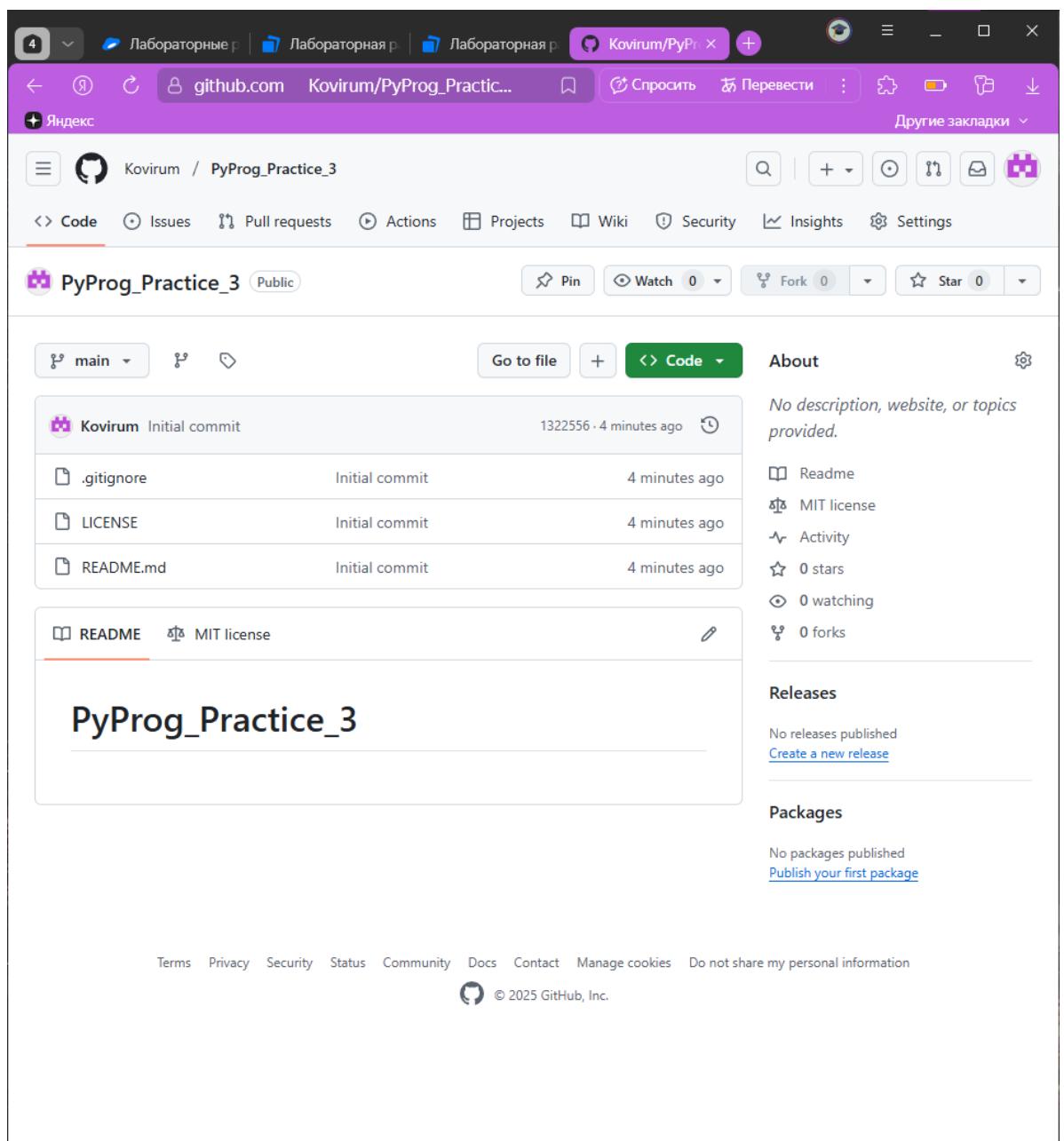


Рисунок 1. Страница созданного репозитория

Ссылка на репозиторий: https://github.com/Kovirum/PyProg_Practice_3

Далее репозиторий был клонирован на компьютер и начата работа над заданиями с соблюдением принципов модели ветвления git-flow, а также правил оформления кода PEP-8:

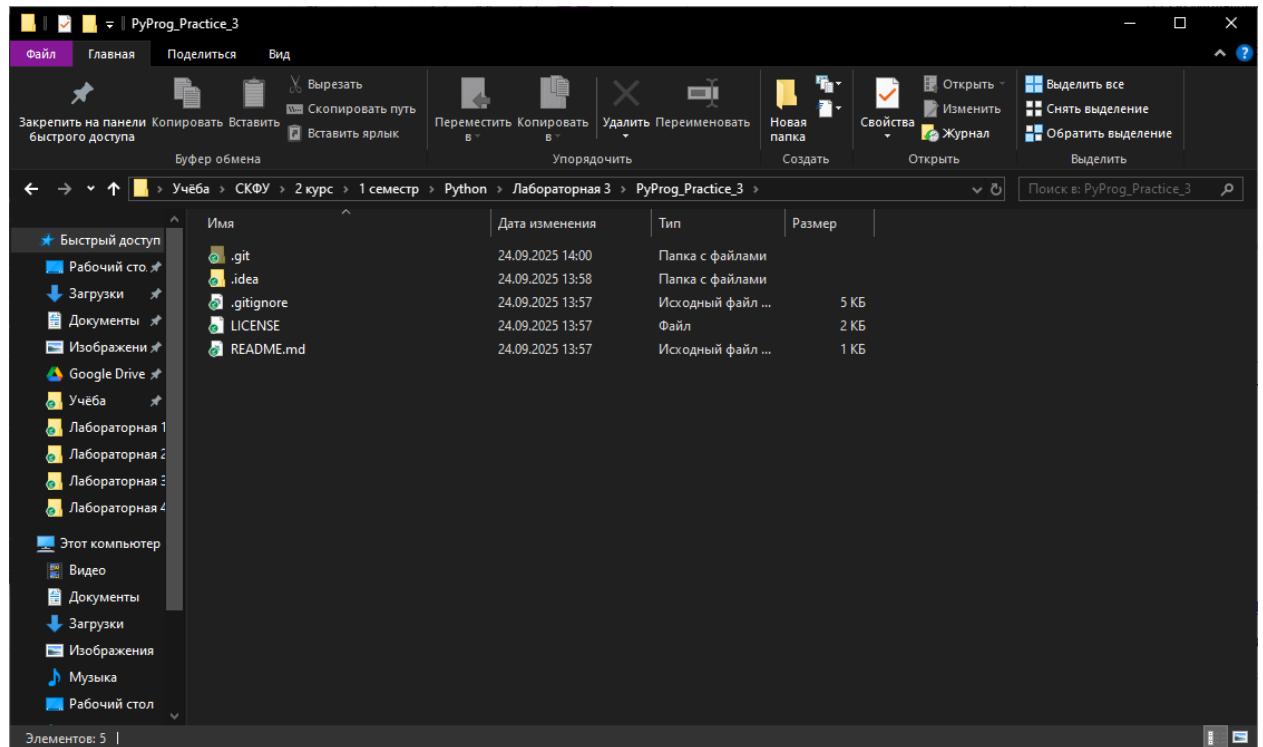


Рисунок 2. Директория локального репозитория

Для выполнения задания проработки всех примеров из методических указаний были организованы отдельные ветки с наименованием «feature/exampleN», где N – номер примера. Каждый отдельный пример реализован в собственном модуле с наименованием «exampleN.py», где N – номер примера.

Пример 1:

The screenshot shows the PyCharm IDE interface. The top part displays the code for `example.py` in a dark-themed editor. The code defines a function that calculates y based on the value of x according to different conditions. The bottom part shows the Git log, which contains three commits:

- feat: add example1.py (Initial commit, Kovirum, A minute ago)
- feat: add example1.py (Kovirum, 35 minutes ago)
- Commit details (for the second commit)

Project navigation and other toolbars are visible at the top and bottom of the interface.

Рисунок 3. Код и git-информация примера 1

Пример 2:

The screenshot shows the PyCharm IDE interface. The top part displays the code for `example2.py` in a dark-themed editor. The code is a script that prints the season corresponding to a given month number. The bottom part shows the Git log, which contains three commits:

- feat: add example2.py (Initial commit, Kovirum, 4 minutes ago)
- feat: add example1.py (Kovirum, 9 minutes ago)
- Commit details (for the second commit)

Project navigation and other toolbars are visible at the top and bottom of the interface.

Рисунок 4. Код и git-информация примера 2

Пример 3:

The screenshot shows the PyCharm IDE interface. The project navigation bar at the top indicates the current file is 'example3.py'. The left sidebar shows the project structure with files like '.gitignore', 'LICENSE', and 'README.md'. The main editor window displays the following Python code:

```
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 import math
5
6
7 if __name__ == '__main__':
8     n = int(input("Value of n? "))
9     x = float(input("Value of x? "))
10
11     S = 0.0
12     for k in range(1, n+1):
13         a = math.log(k * x) / (k * k)
14         S += a
15
16     print(f"S = {S}")
```

The bottom right corner of the editor shows the status: '1:1 CRLF UTF-8 4 spaces Python 3.13'.

The bottom half of the screen shows the Git log. It lists several commits:

- feat: add example3.py (Kovirum, 2 minutes ago)
- feat: add example2.py (Kovirum, 10 minutes ago)
- feat: add example1.py (Kovirum, 15 minutes ago)
- Initial commit (origin & main Kovirum*, 50 minutes ago)

A tooltip in the log area says 'Select commit to view changes'.

Рисунок 5. Код и git-информация примера 3

Пример 4:

The screenshot shows the PyCharm IDE interface. The project navigation bar at the top indicates the current file is 'example4.py'. The left sidebar shows the project structure with files like '.gitignore', 'LICENSE', and 'README.md'. The main editor window displays the following Python code:

```
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 import math
5 import sys
6
7
8 if __name__ == '__main__':
9     a = float(input("Value of a? "))
10    if a < 0:
11        print("Illegal value of a", file=sys.stderr)
12        exit(1)
13
14    x, eps = 1, 1e-10
15    while True:
16        xp = x
17        x = (x + a / x) / 2
18        if math.fabs(x - xp) < eps:
19            break
20
21
22    print(f"x = {x}\nX={math.sqrt(a)}")
```

The bottom right corner of the editor shows the status: 'Analyzing...'.

The bottom half of the screen shows the Git log. It lists several commits:

- feat: add example4.py (Kovirum, A minute ago)
- feat: add example3.py (Kovirum, 8 minutes ago)
- feat: add example2.py (Kovirum, 16 minutes ago)
- feat: add example1.py (Kovirum, 21 minutes ago)
- Initial commit (origin & main Kovirum*, 56 minutes ago)

A tooltip in the log area says 'Select commit to view changes'.

A message box at the bottom right says 'Microsoft Defender configuration' and 'Project paths were successfully added to the Microsoft Defender exclusion list'.

Рисунок 6. Код и git-информация примера 4

Пример 5:

The screenshot shows the PyCharm IDE interface. The top part displays the code for `example5.py` in a dark-themed editor. The code calculates the value of Euler's constant e using a series expansion. The bottom part shows the Git log for the project `PyProg_Practice_3`. The log lists several commits, with the most recent one being `feat: add example5.py` by user `Kovirum` on 24.09.2025 at 14:52.

```
3
4 import math
5 import sys
6
7 # Постоянная Эйлера
8 EULER = 0.5772156649015328606
9 # Точность вычислений
10 EPS = 1e-10
11
12
13
14 if __name__ == '__main__':
15     x = float(input("Value of x? "))
16     if x == 0:
17         print("Illegal value of x", file=sys.stderr)
18         exit(1)
19
20     a = x
21     S, k = a, 1
22
23     # Найти сумму членов ряда.
24     while math.fabs(a) > EPS:
25         a *= x * k / (k + 1) ** 2
26         S += a
27         k += 1
28
29     # Вывести значение функции.
30     print(f"Ei({x}) = {EULER + math.log(math.fabs(x)) + 5}")
```

Commit	Author	Date
feat: add example5.py	Kovirum	24.09.2025 14:52
feat: add example4.py	Kovirum	24.09.2025 14:44
feat: add example3.py	Kovirum	24.09.2025 14:37
feat: add example2.py	Kovirum	24.09.2025 14:29
feat: add example1.py	Kovirum	24.09.2025 14:24
Initial commit	origin & main	24.09.2025 13:50

Рисунок 7. Код и git-информация примера 5

Далее были выполнены индивидуальные задания для варианта 19:

Задание 1. Дано целое число C , такое что $|C| < 9$. Вывести это число в словесной форме, учитывая его знак.

Код программы:

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

import sys

if __name__ == '__main__':
    c = int(input("Введите число C такое, что |C| < 9: "))

    if not abs(c) < 9:
        print("Число C должно соответствовать условию: |C| < 9",
              file=sys.stderr)
        exit(1)

    result_str = ""

    if c < 0:
        result_str += "минус "
```

```
match abs(c):
    case 0:
        result_str += "ноль"
    case 1:
        result_str += "один"
    case 2:
        result_str += "два"
    case 3:
        result_str += "три"
    case 4:
        result_str += "четыре"
    case 5:
        result_str += "пять"
    case 6:
        result_str += "шесть"
    case 7:
        result_str += "семь"
    case 8:
        result_str += "восемь"
    case 9:
        result_str += "девять"

print(result_str)
```

Демонстрация работы программы:

The screenshot shows the PyCharm IDE interface. The top bar displays the project name "PyProg_Practice_3" and the current file "individual_task_1.py". The left sidebar shows the project structure with files like "individual_task_1.py", "LICENSE", and "README.md". The main editor window contains the following Python code:

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

import sys

if __name__ == '__main__':
    c = int(input("Введите число С такое, что |С| < 9:"))

    if not abs(c) < 9:
        print("Число С должно соответствовать условию: |С| < 9", file=sys.stderr)
        exit(1)

    result_str = ""

    if c < 0:
        result_str += "МИНУС "
    match abs(c):
        case 0:
            result_str += "НОЛЬ"
        case 1:
            result_str += "ОДИН"
        case 2:
            result_str += "ДВА"
        case 3:
            result_str += "ТРИ"
        case 4:
            result_str += "ЧЕТЫРЕ"
        case 5:
            result_str += "ПЯТЬ"
```

The bottom run tool window shows the command run: "C:\Users\user\AppData\Local\Programs\Python\Python313\python.exe "C:\Users\user\Desktop\Учёба\СКФУ\2 курс\1 семестр\Python\Лабораторная 3\PyProg_Practice_3\individual_task_1.py"" and the output: "Введите число С такое, что |С| < 9: -6" followed by "МИНУС шесть". The status bar at the bottom indicates the file is saved (green checkmark), the current file is "individual_task_1.py", and the Python version is 3.13.

Рисунок 8. Демонстрация работы программы индивидуального задания 1

UML-диаграмма деятельности:

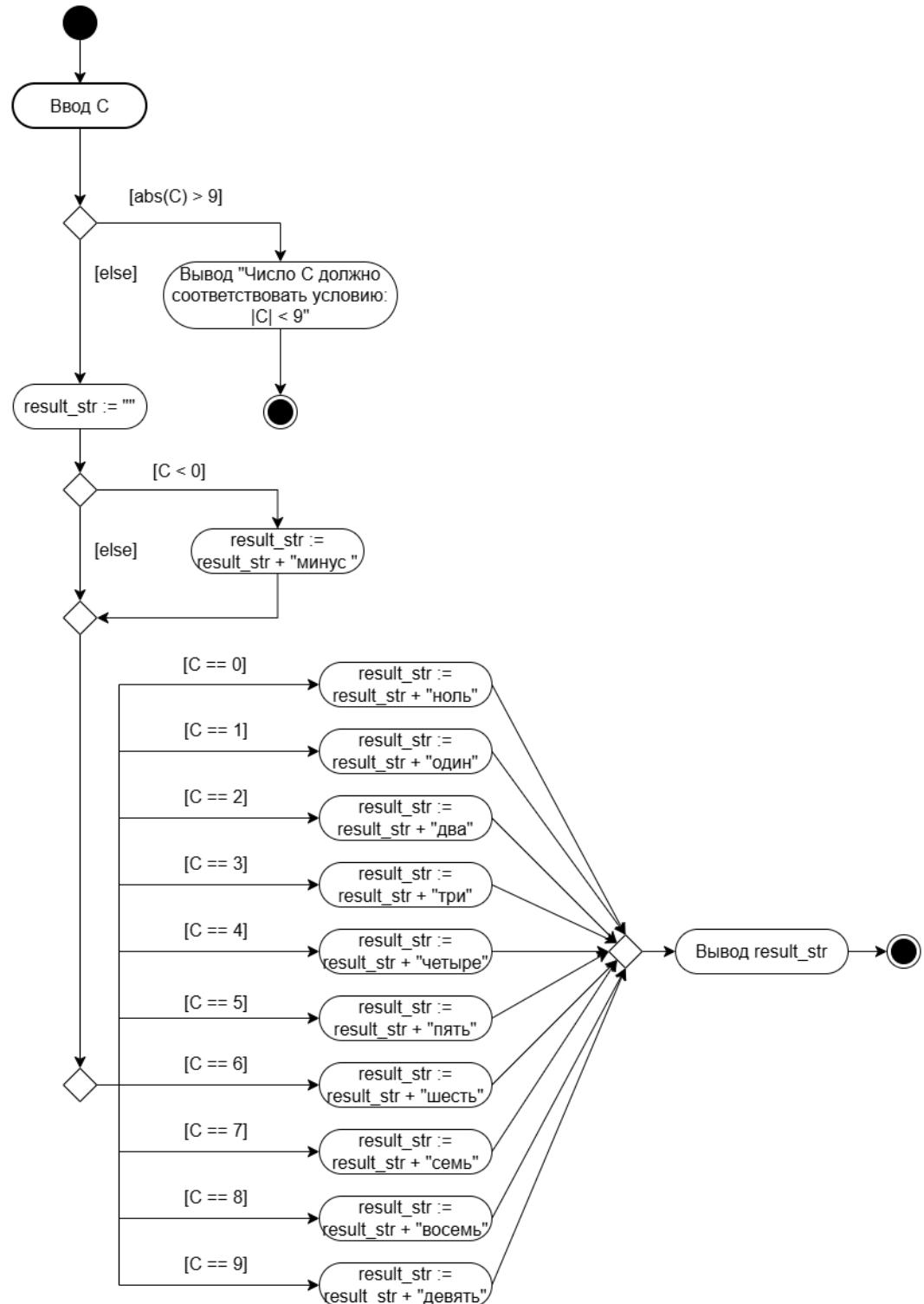


Рисунок 9. UML-диаграмма для индивидуального задания 1

Задание 2. Какая из точек А(а1, а2) или В(б1, б2) находится дальше от центра координат?

Код программы:

```

#!/usr/bin/env python3
# -*- coding: utf-8 -*-
  
```

```

import math

if __name__ == '__main__':
    a1, a2 = map(int, input("Укажите координаты точки А (a1, a2): ").split())
    b1, b2 = map(int, input("Укажите координаты точки В (b1, b2): ").split())

    OA_distance = math.sqrt(math.pow(a1 - 0, 2) + math.pow(a2 - 0, 2))
    OB_distance = math.sqrt(math.pow(b1 - 0, 2) + math.pow(b2 - 0, 2))

    if OA_distance > OB_distance:
        print("Точка А находится дальше от начала координат")
    elif OA_distance == OB_distance:
        print("Точки А и В находятся на одинаковом удалении от начала координат")
    else:
        print("Точка В находится дальше от начала координат")

```

Демонстрация работы программы:

The screenshot shows the PyCharm IDE interface. In the top navigation bar, the project is named 'PyProg_Practice_3' and the current file is 'individual_task_2.py'. The code editor displays the Python script provided above. Below the editor is a 'Run' tool window. The 'Run' tab is selected, showing the command 'C:\Users\user\AppData\Local\Programs\Python\Python313\python.exe "C:\Users\user\Desktop\Учеба\СКФУ\2 курс\1 семестр\Python\Лабораторная 3\PyProg_Practice_3\individual_task_2.py"'. The terminal pane at the bottom shows the program's output: 'Укажите координаты точки А (a1, a2): 1 9', 'Укажите координаты точки В (b1, b2): 3 6', 'Точка А находится дальше от начала координат', and 'Process finished with exit code 0'. The status bar at the bottom right indicates the time as 19:1, encoding as CRLF, character set as UTF-8, and spaces as 4 spaces, with Python 3.13.

Рисунок 10. Демонстрация работы программы индивидуального задания 2

UML-диаграмма:

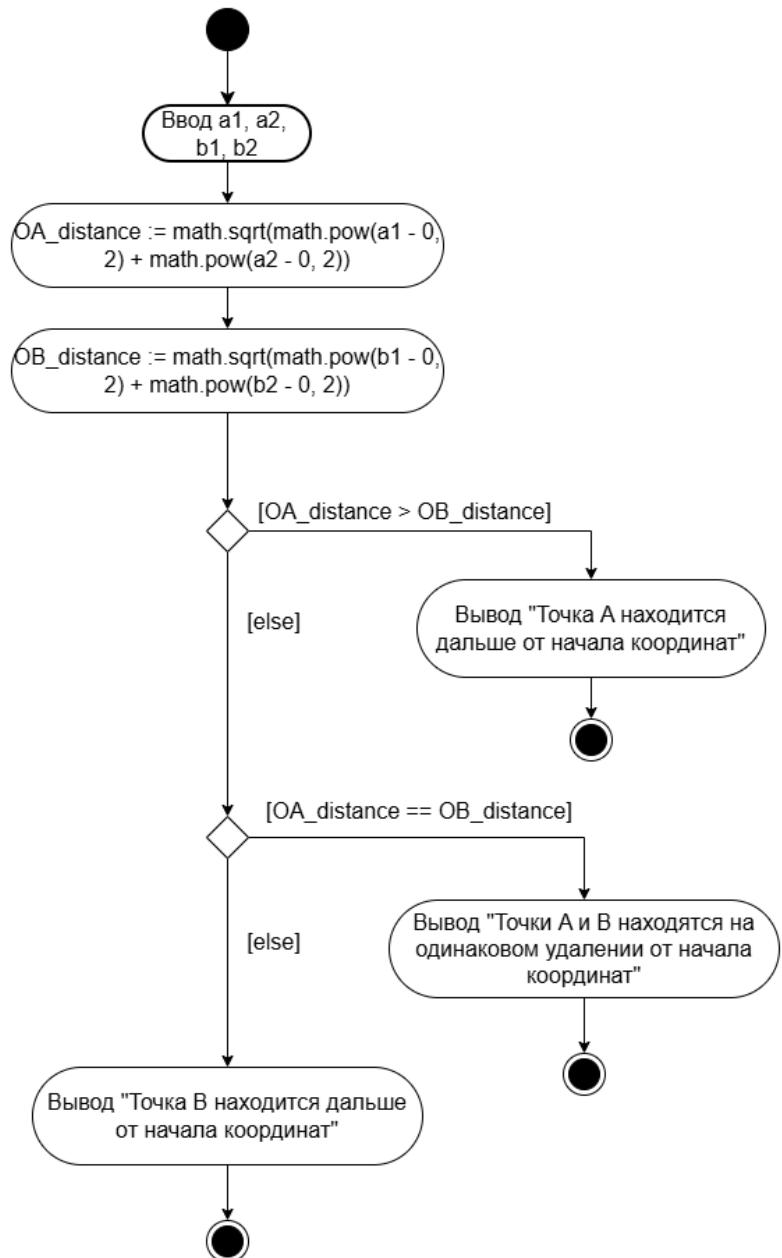


Рисунок 11. UML-диаграмма для индивидуального задания 2

Задание 3. У гусей и кроликов вместе 64 лапы. Сколько могло быть кроликов и гусей (указать все сочетания, которые возможны)

Код программы:

```

#!/usr/bin/env python3
# -*- coding: utf-8 -*-

TOTAL_PAWS = 64

if __name__ == '__main__':
    for c, i in enumerate(range(0, TOTAL_PAWS + 1, 4), 1):
        goose_count = (TOTAL_PAWS - i) // 2
        rabbit_count = (TOTAL_PAWS - goose_count * 2) // 4

```

```

        print(f"[#{c}] Гусей - {goose_count}, Кроликов -
{rabbit_count}. "
              f"Лап: {goose_count * 2} + {rabbit_count * 4} = "
              f"{goose_count * 2 + rabbit_count * 4}")

```

Демонстрация работы программы:

The screenshot shows the PyCharm IDE interface. The top navigation bar includes 'PC', 'PyProg_Practice_3', 'Version control', and file tabs for 'README.md' and 'individual_task_3.py'. The 'individual_task_3.py' tab is active, displaying Python code. The code initializes a total of 64 paws and iterates through possible counts for geese and rabbits, printing the results. The bottom panel shows the 'Run' tab with the output of the program's execution. The output lists 17 rows of results, each showing a combination of geese and rabbits that sum up to 64 paws. The final message indicates the process finished with exit code 0.

```

# !usr/bin/env python3
# -*- coding: utf-8 -*-

TOTAL_PAWS = 64

if __name__ == '__main__':
    for c, i in enumerate(range(0, TOTAL_PAWS + 1, 4), 1):
        goose_count = (TOTAL_PAWS - i) // 2
        rabbit_count = (TOTAL_PAWS - goose_count * 2) // 4
        print(f"[#{c}] Гусей - {goose_count}, Кроликов - {rabbit_count}. "
              f"Лап: {goose_count * 2} + {rabbit_count * 4} = {goose_count * 2 + rabbit_count * 4}")

```

Output:

```

[ #1] Гусей - 32, Кроликов - 0. Лап: 64 + 0 = 64
[ #2] Гусей - 30, Кроликов - 1. Лап: 60 + 4 = 64
[ #3] Гусей - 28, Кроликов - 2. Лап: 56 + 8 = 64
[ #4] Гусей - 26, Кроликов - 3. Лап: 52 + 12 = 64
[ #5] Гусей - 24, Кроликов - 4. Лап: 48 + 16 = 64
[ #6] Гусей - 22, Кроликов - 5. Лап: 44 + 20 = 64
[ #7] Гусей - 20, Кроликов - 6. Лап: 40 + 24 = 64
[ #8] Гусей - 18, Кроликов - 7. Лап: 36 + 28 = 64
[ #9] Гусей - 16, Кроликов - 8. Лап: 32 + 32 = 64
[ #10] Гусей - 14, Кроликов - 9. Лап: 28 + 36 = 64
[ #11] Гусей - 12, Кроликов - 10. Лап: 24 + 40 = 64
[ #12] Гусей - 10, Кроликов - 11. Лап: 20 + 44 = 64
[ #13] Гусей - 8, Кроликов - 12. Лап: 16 + 48 = 64
[ #14] Гусей - 6, Кроликов - 13. Лап: 12 + 52 = 64
[ #15] Гусей - 4, Кроликов - 14. Лап: 8 + 56 = 64
[ #16] Гусей - 2, Кроликов - 15. Лап: 4 + 60 = 64
[ #17] Гусей - 0, Кроликов - 16. Лап: 0 + 64 = 64

```

Process finished with exit code 0

Рисунок 12. Демонстрация работы программы индивидуального задания 3

UML-диаграмма:

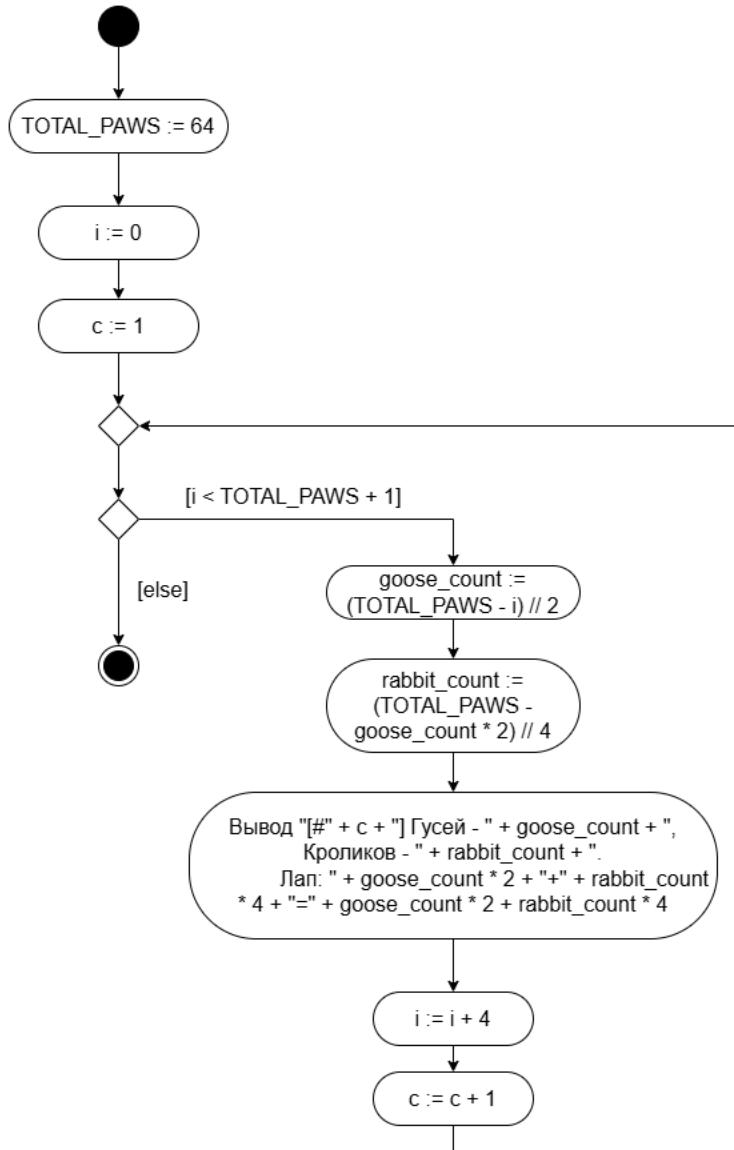


Рисунок 13. UML-диаграмма для индивидуального задания 3

Далее, после выполнения всех примеров и индивидуальных заданий, необходимо внести все изменения в основную ветку согласно модели ветвления git-flow.

Для этого сначала все feature-ветки были слиты в develop:

```
OK Командная строка + ^

C:\Users\polko\Desktop\Учёба\СКФУ\2 курс\1 семестр\Python\Лабораторная 3\PyProg_Practice_3>git merge feature/example4
Already up to date.

C:\Users\polko\Desktop\Учёба\СКФУ\2 курс\1 семестр\Python\Лабораторная 3\PyProg_Practice_3>git merge feature/example5
Auto-merging .idea/workspace.xml
CONFLICT (add/add): Merge conflict in .idea/workspace.xml
Automatic merge failed; fix conflicts and then commit the result.

C:\Users\polko\Desktop\Учёба\СКФУ\2 курс\1 семестр\Python\Лабораторная 3\PyProg_Practice_3>git merge feature/individual1
Merge made by the 'ort' strategy.
individual_task_1.py | 39 ++++++=====
1 file changed, 39 insertions(+)
create mode 100644 individual_task_1.py

C:\Users\polko\Desktop\Учёба\СКФУ\2 курс\1 семестр\Python\Лабораторная 3\PyProg_Practice_3>git merge feature/individual2
Merge made by the 'ort' strategy.
individual_task_2.py | 18 ++++++=====
1 file changed, 18 insertions(+)
create mode 100644 individual_task_2.py

C:\Users\polko\Desktop\Учёба\СКФУ\2 курс\1 семестр\Python\Лабораторная 3\PyProg_Practice_3>git merge feature/individual3
Merge made by the 'ort' strategy.
individual_task_3.py | 12 ++++++=====
1 file changed, 12 insertions(+)
create mode 100644 individual_task_3.py

C:\Users\polko\Desktop\Учёба\СКФУ\2 курс\1 семестр\Python\Лабораторная 3\PyProg_Practice_3>
```

Рисунок 14. Процесс слияния feature-веток в develop

Далее была подготовлена release-ветка: