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 программной инженерии»

Тема: «Условные операторы и циклы в языке Python»

Выполнила:
студентка 2 курса
группы Пиж-б-о-21-1
Джолдошова Мээрим
Бекболотовна

Ставрополь 2022

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.

1. Был создан общедоступного репозиторий в GitHub в котором были добавлены gitignore, правила для работы с IDE PyCharm с ЯП Python и лицензия MIT, репозиторий был клонировал на локальный сервер и организован в соответствие с моделью ветвления git-flow.

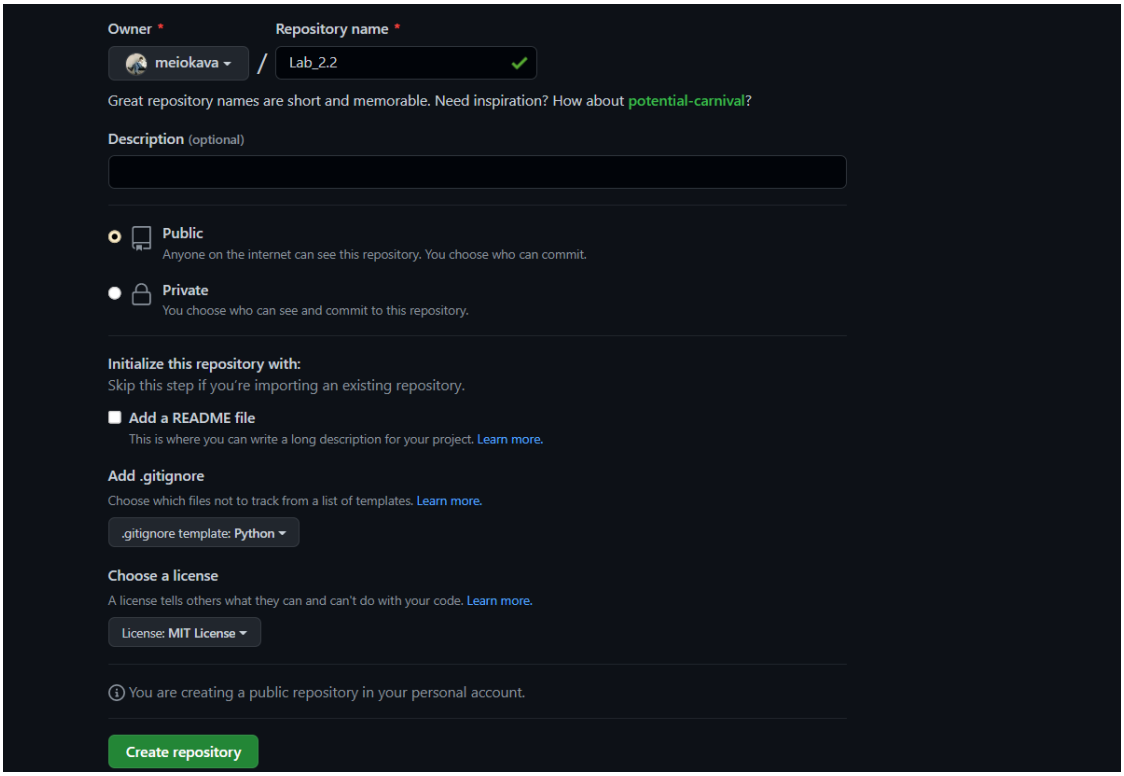

Рисунок 1.1 – Создание общедоступного репозитория

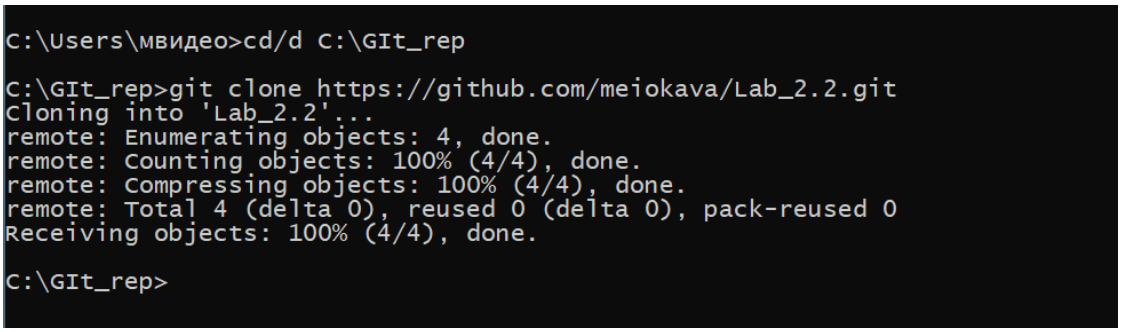

Рисунок 1.2 – Клонирование созданного репозитория

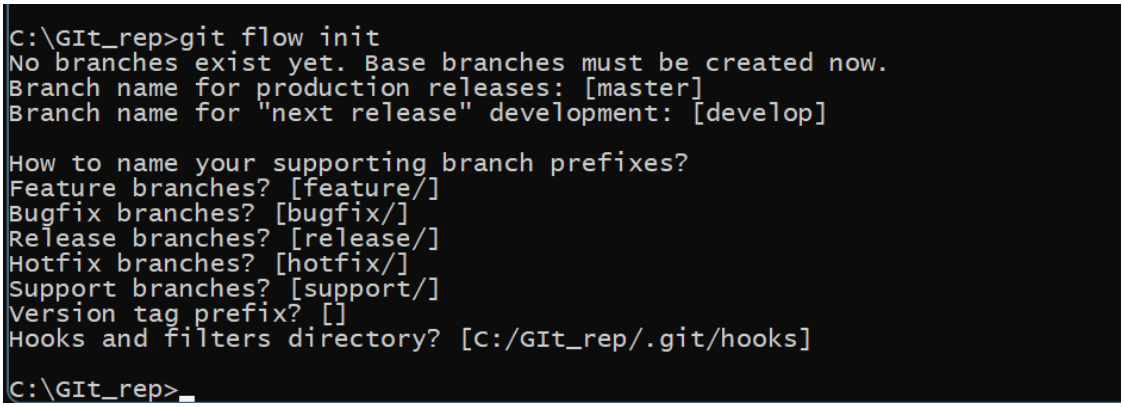

Рисунок 1.3 – Организование репозитория по модели ветвления git glow

2. Была создана папка (common_files), содержащая примеры из лабораторной работы

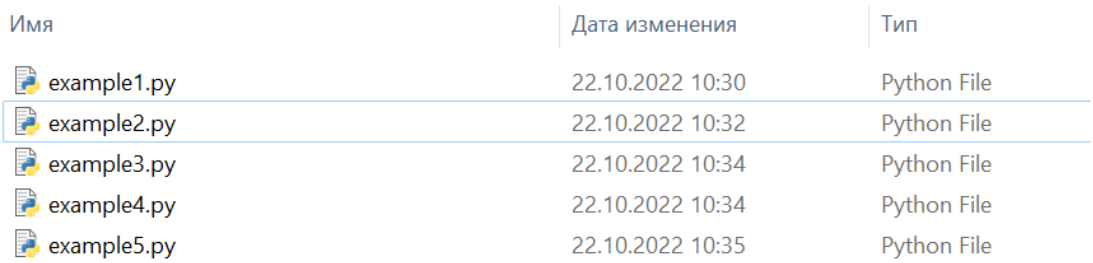

Рисунок 2.1 – Содержание папки common_files

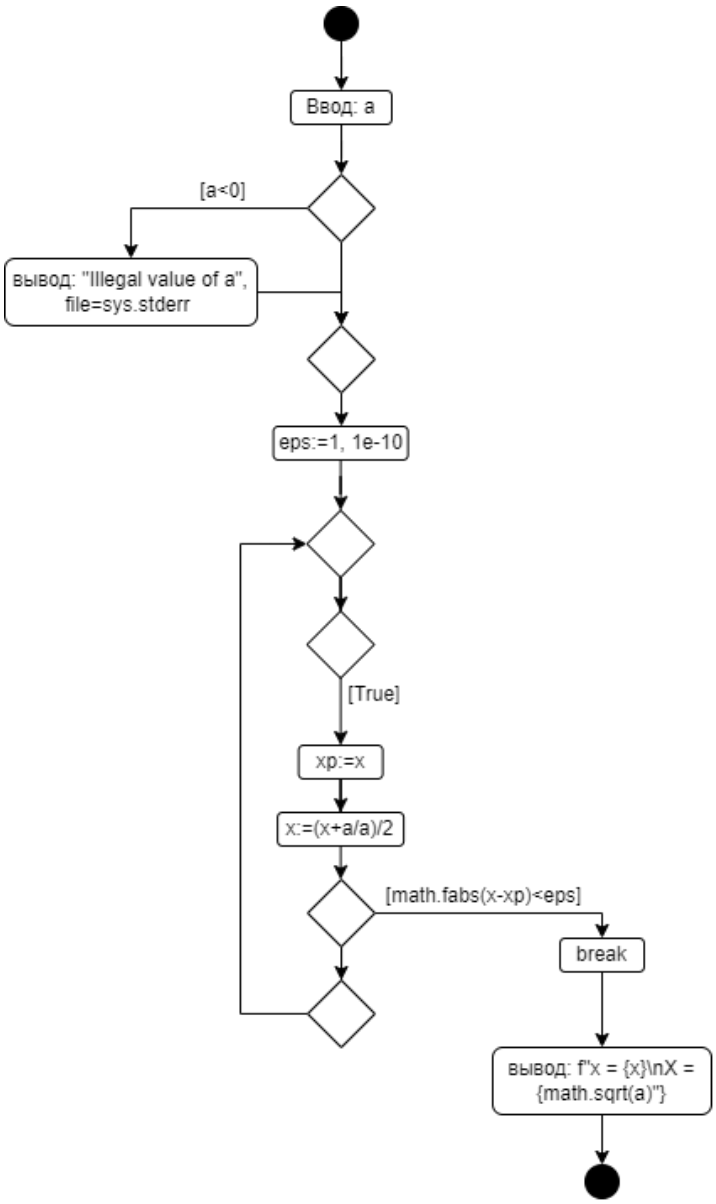

Рисунок 2.2 – UML-диаграмма программы для 4 примеры

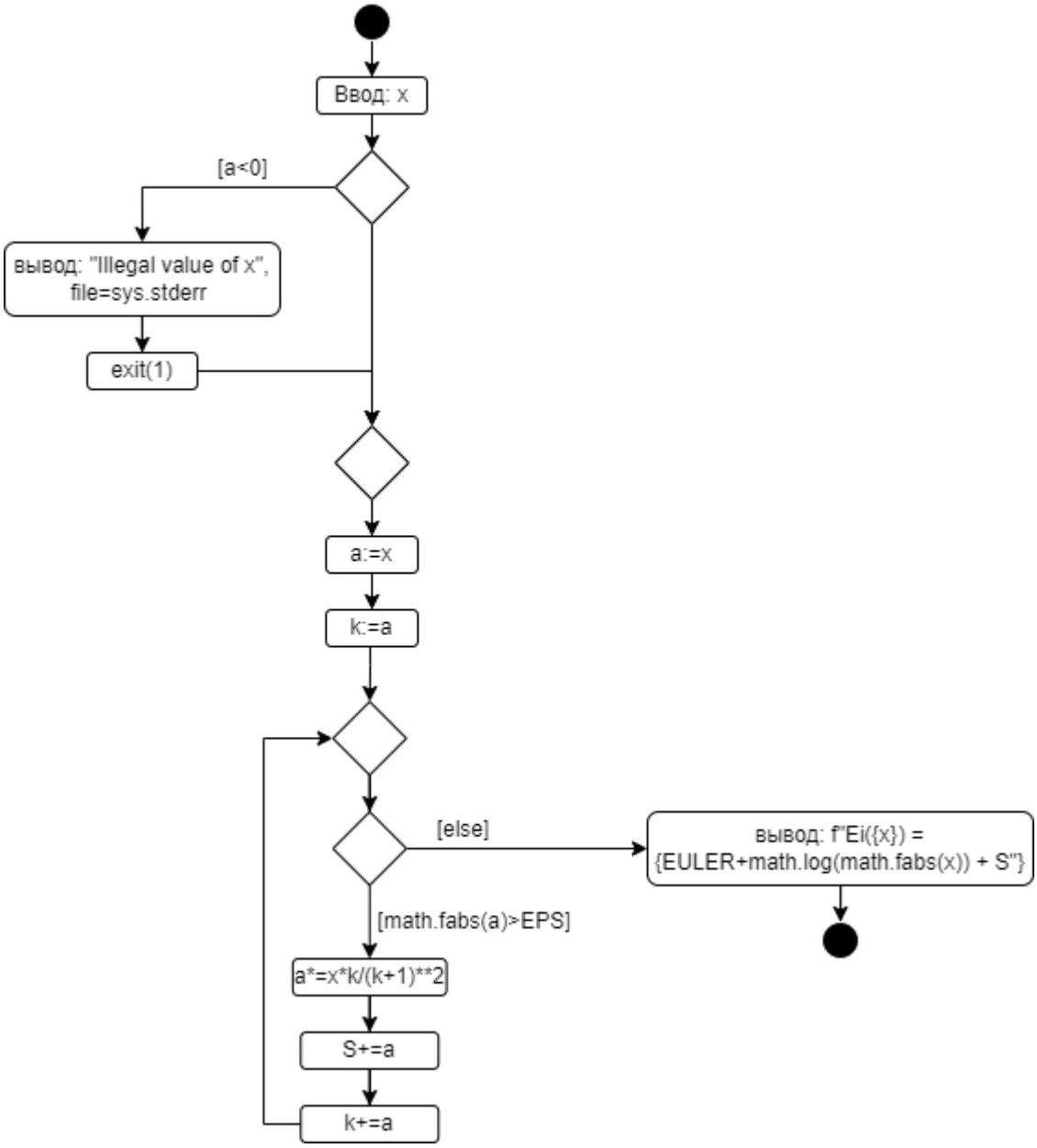

Рисунок 2.3 – UML-диаграмма для программы 5 примера

Индивидуальное задание

3 Было сделано индивидуальное задание, а также задание повышенной сложности согласно вариантам. Была построена UML диаграмма.

Вариант 5

Задание 1. С клавиатуры вводится цифра (от 1 до 4). Вывести на экран названия месяцев, соответствующих времени года с номером (считать зиму временем года No 1).

Код программы:

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

if __name__ == '__main__':
    a = int(input('enter digit (the rage is from 1 to 4): '))
    if a == 1:
        print('it is winter')
    elif a == 2:
        print('it is spring')
    elif a == 3:
        print('it is summer')
    elif a == 4:
        print('it is autumn')
    else:
        print('error choose within a range')
```

```
"C:\Program Files\Python310\python.exe" C:\lbrtt_2.2\lbrtt_2.2\indiv_tasks\indiv1.py
enter digit (the rage is from 1 to 4): 3
it is summer

Process finished with exit code 0
```

Рисунок 3.1 – Результат работы программы

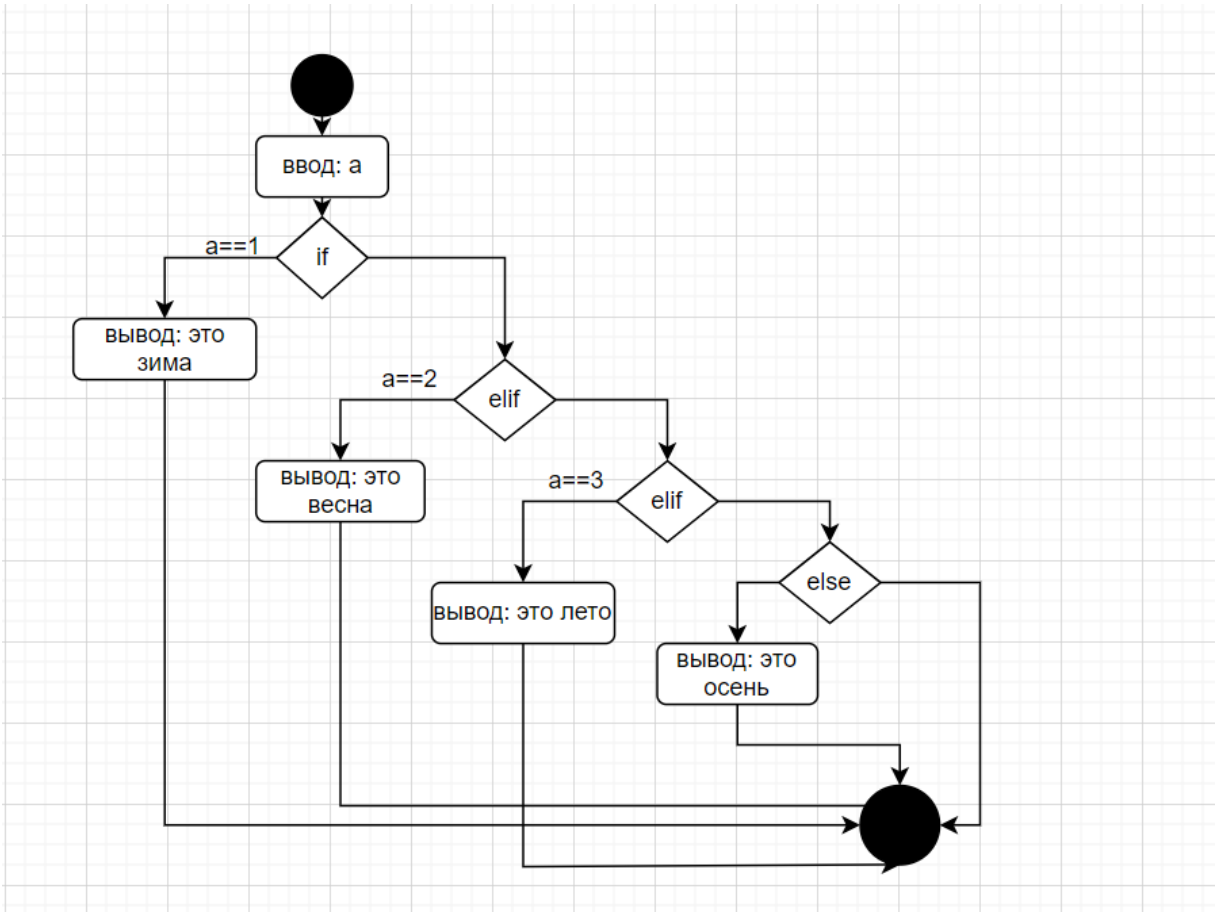

Рисунок 3.2 – UML диаграмма программы

Задание 2. Определить принадлежит ли точка A (a, b) кольцу определяемому окружностями x^2 + y^2 = 1 и x^2 + y^2 = 0.25.

Код программы:

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

if __name__ == '__main__':
    x = int(input('enter a: '))
    y = int(input('enter b: '))
    if x**2 + y**2 == 1:
        print('these coordinates belong to circle x^2 + y^2 = 1')
    elif x**2 + y**2 == 0.25:
        print('these coordinates belong to circle x^2 + y^2 = 0.25')
    else:
        print('these coordinates dont belong to circle')
```

```
C:\Users\мвидео\PycharmProjects\pythonProject19\venv\Scripts\python.
enter a: 0
enter b: 1
these coordinates belong to circle x^2 + y^2 = 1

Process finished with exit code 0
```

Рисунок 3.3 – Результат работы программы

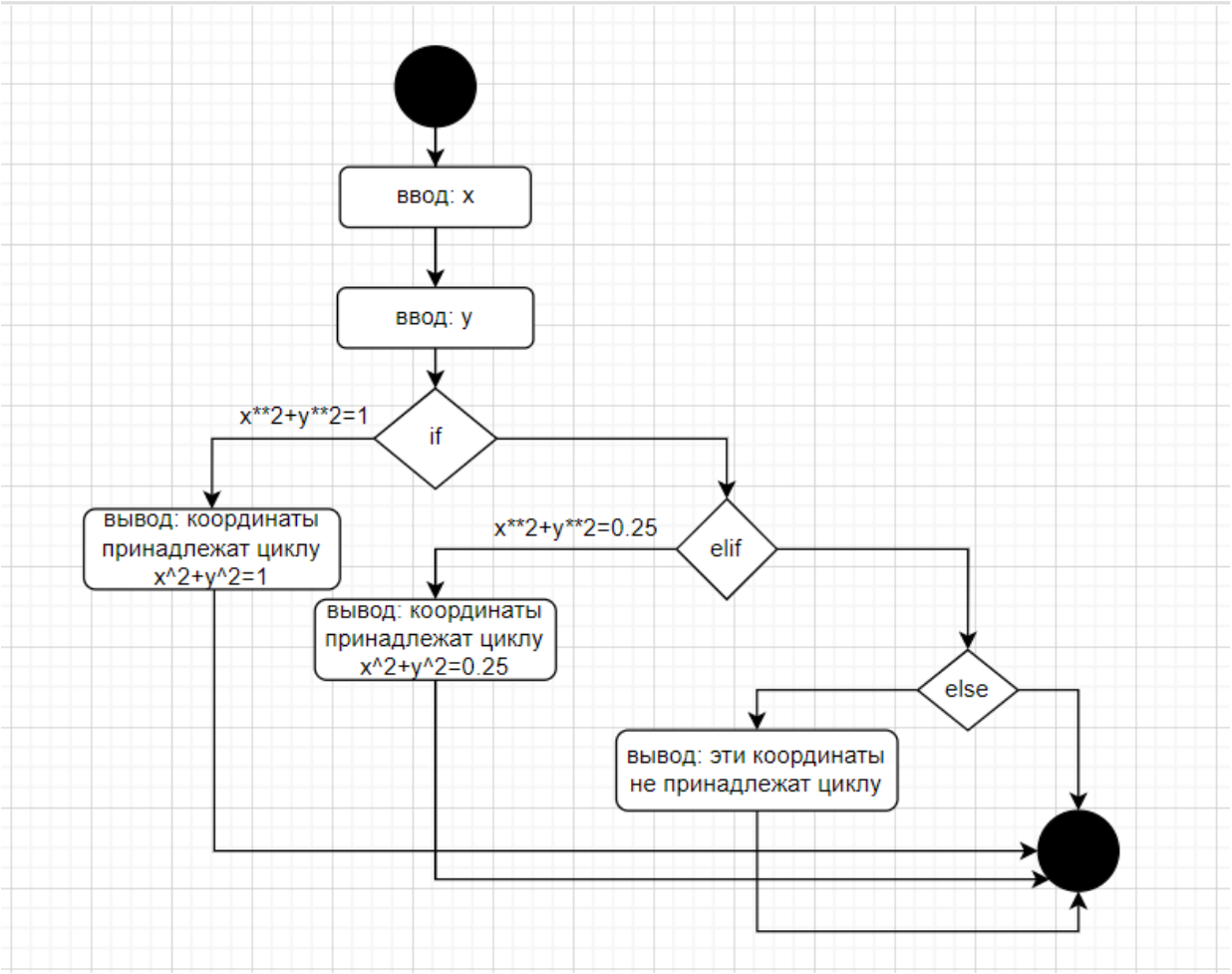

Рисунок 3.4 – UML диаграмма программы

Задание 3. Одноклеточная амеба каждые три часа делится на 2 клетки. Определить, сколько будет клеток через 6 часов.

Код программы:

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

if __name__ == '__main__':
    ameba = 1
    time_list = [1, 2, 3, 4, 5, 6]
    for time in time_list:
        ameba *= 2
        print('after', time, 'hours it will be', ameba, 'cells')
```

```
after 1 hours it will be 2 cells
after 2 hours it will be 4 cells
after 3 hours it will be 8 cells
after 4 hours it will be 16 cells
after 5 hours it will be 32 cells
after 6 hours it will be 64 cells

Process finished with exit code 0
```

Рисунок 3.5 – Результат работы программы

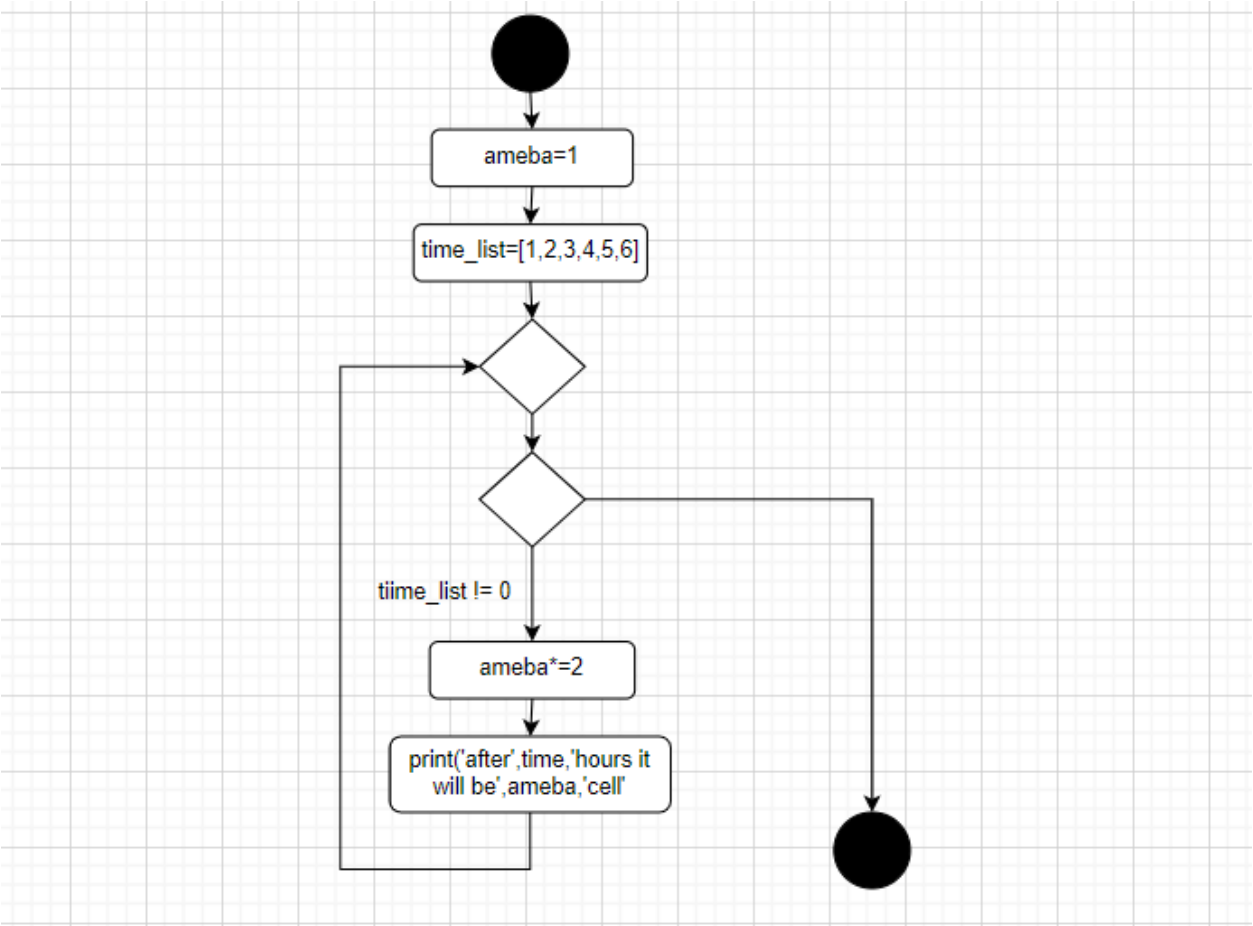

Рисунок 3.6 – UML диаграмма программы

Задание повышенной сложности

4. Интегральный гиперболический косинус:

$$\operatorname{Chi}(x) = \gamma + \ln x + \int_0^x \frac{\operatorname{ch} t - 1}{t}\, dt = \gamma + \ln x + \sum_{n=1}^{\infty} \frac{x^{2n}}{(2n)(2n)!}.$$

Код программы

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
import math
import sys

# Постоянная Эйлера.
EULER = 0.5772156649015328606
# Точность вычислений.
EPS = 1e-10

if __name__ == '__main__':
    x = float(input("Value of x? "))
    if x == 0:
        print("Illegal value of x", file=sys.stderr)
        exit(1)

    a = x
    S, n = a, 1

    # Найти сумму членов ряда.
    while math.fabs(a) > EPS:
        a *= (x * (2*n)) / ((2 * n + 1)**2)
        S += a
        n += 1

    # Вывести значение функции.
    print(f"Fi({x}) = {EULER + math.log(math.fabs(x)) + S}")
```

```
"C:\Program Files\Python310\python.exe" C:\lbrtt_2.2\lbrtt_2.2\up_diff\up_diff5.py
Value of x? 10
Fi(10.0) = 249.9555708149087

Process finished with exit code 0
```

Рисунок 3.6 – Результат работы программы

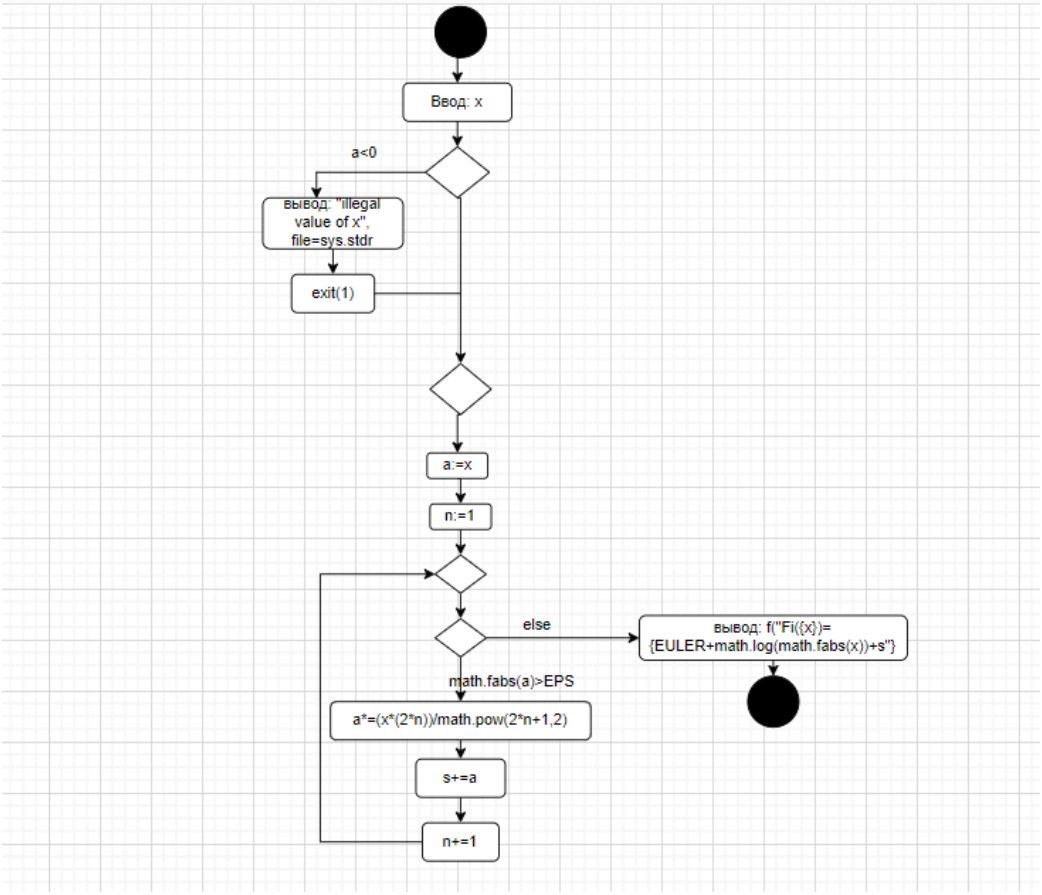

Рисунок 3.6 – UML диаграмма программы

```
C:\lbrtt_2.2\lbrtt_2.2>git add .
warning: in the working copy of 'indiv_tasks/.idea/inspectionProfiles/profiles_settings.xml', LF will b
es it

C:\lbrtt_2.2\lbrtt_2.2>git commit -m "new folders"
[develop c1b7e6c] new folders
 20 files changed, 194 insertions(+)
 create mode 100644 common_tasks/example1.py
 create mode 100644 common_tasks/example2.py
 create mode 100644 common_tasks/example3.py
 create mode 100644 common_tasks/example4.py
 create mode 100644 common_tasks/example5.py
 create mode 100644 diagrams/diff.drawio
 create mode 100644 diagrams/indiv1.drawio
 create mode 100644 diagrams/indiv2.drawio
 create mode 100644 diagrams/indiv3.drawio
 create mode 100644 indiv_tasks/.idea/.gitignore
 create mode 100644 indiv_tasks/.idea/.name
 create mode 100644 indiv_tasks/.idea/indiv_tasks.iml
 create mode 100644 indiv_tasks/.idea/inspectionProfiles/profiles_settings.xml
 create mode 100644 indiv_tasks/.idea/misc.xml
 create mode 100644 indiv_tasks/.idea/modules.xml
 create mode 100644 indiv_tasks/.idea/vcs.xml
 create mode 100644 indiv_tasks/indiv1.py
 create mode 100644 indiv_tasks/indiv2.py
 create mode 100644 indiv_tasks/indiv3.py
 create mode 100644 up_diff/up_diff5.py

C:\lbrtt_2.2\lbrtt_2.2>git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

C:\lbrtt_2.2\lbrtt_2.2>_
```

Рисунок 3.7 – Был сделан коммит изменений

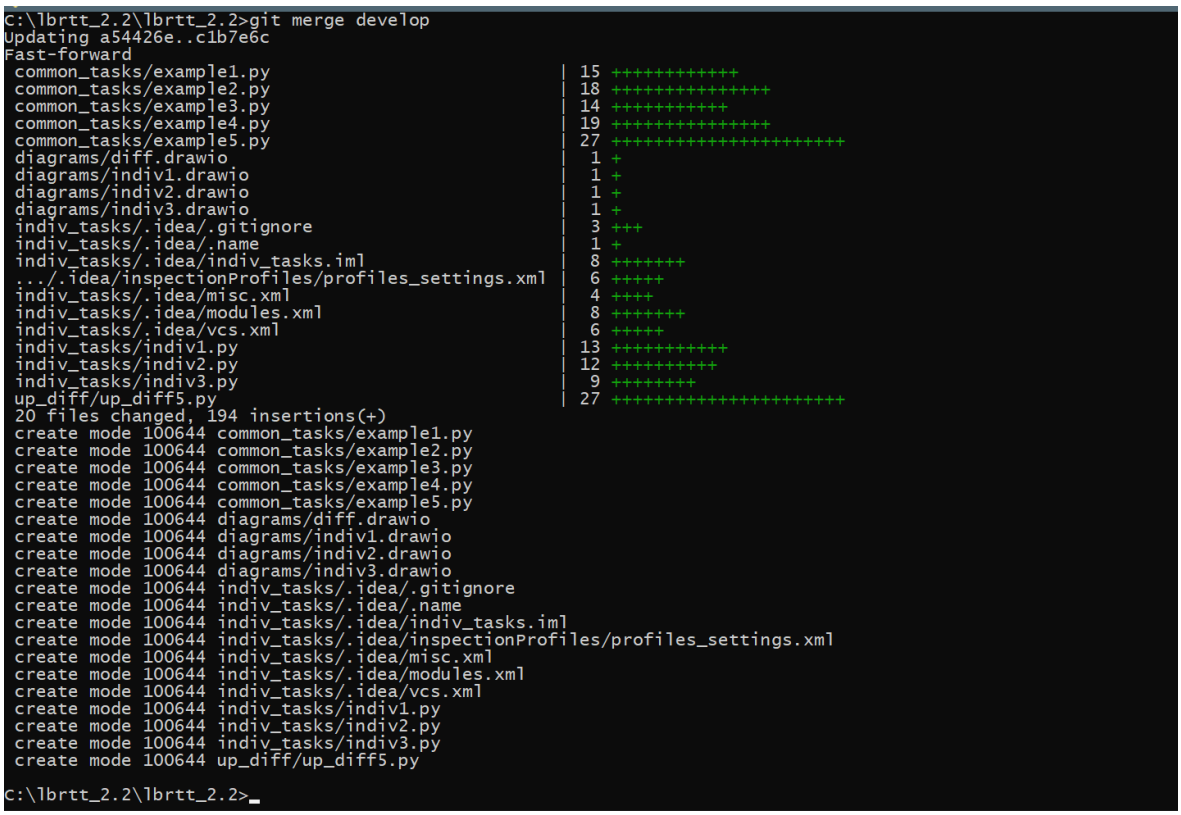

Рисунок 3.8 – Было осуществлено слияние веток main и develop

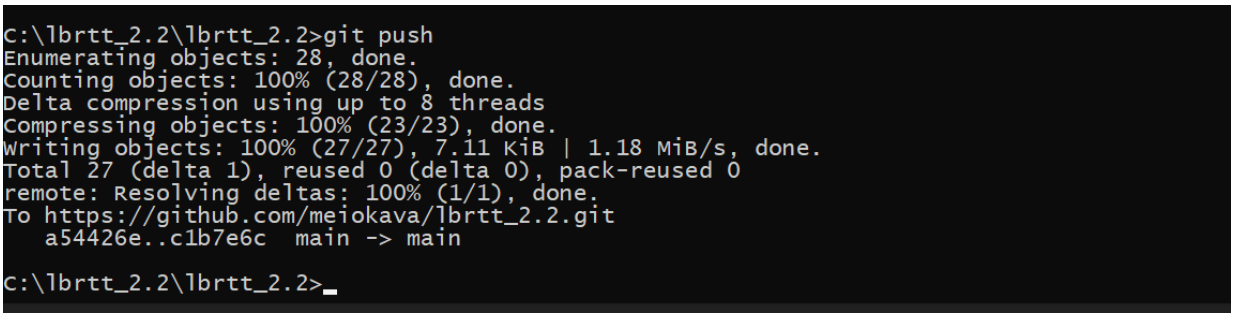

Рисунок 3.9 – Была осуществленная отправка изменений на удаленный сервер

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

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, 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.