

**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
Логвинов Иван
Васильевич

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.

Владелец * / Имя репозитория *

ИванLogvinov11 / lab5_5 ✓

Имена великих репозитория короткие и запоминающиеся. Нужно вдохновение? Как насчет **vigilant-doodle**?

Описание (необязательно)

☒ **Общедоступный**
Любой пользователь Интернета может увидеть этот репозиторий. Вы выбираете, кто может совершить.

☐ **Личное**
Вы выбираете, кто может видеть и фиксировать в этом репозитории.

Инициализируйте этот репозиторий с помощью:
Пропустите этот шаг, если вы импортируете существующий репозиторий.

☒ **Добавьте файл README**
Здесь вы можете написать длинное описание для вашего проекта. [Узнать больше.](#)

Добавить .gitignore
Выберите, какие файлы не отслеживать, из списка шаблонов. [Узнать больше.](#)

.шаблон gitignore: Python ▼

Выберите лицензию
Лицензия сообщает другим пользователям, что они могут и чего не могут делать с вашим кодом. [Узнать больше.](#)

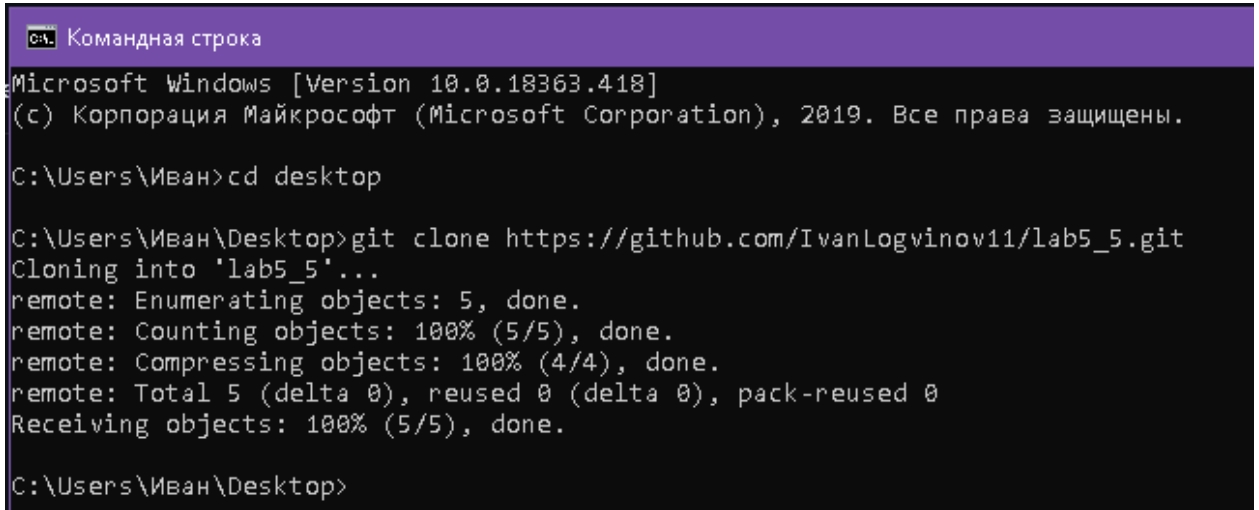
Лицензия: MIT License ▼

Это установит main в качестве ветки по умолчанию. Измените имя по умолчанию в своих [настройках](#).

Вы создаете публичный репозиторий в своем личном кабинете.

Создать репозиторий

Рисунок 1.1 – Создание общедоступного репозитория



```
Командная строка
Microsoft Windows [Version 10.0.18363.418]
(c) Корпорация Майкрософт (Microsoft Corporation), 2019. Все права защищены.

C:\Users\Иван>cd desktop

C:\Users\Иван\Desktop>git clone https://github.com/IvanLogvinov11/lab5_5.git
Cloning into 'lab5_5'...
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (5/5), done.

C:\Users\Иван\Desktop>
```

Рисунок 1.2 – Клонирование созданного репозитория

```
C:\Users\Иван\Desktop>git flow init
Initialized empty Git repository in C:/Users/Иван/Desktop/.git/
No branches exist yet. Base branches must be created now.
Branch name for production releases: [master]
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/Иван/Desktop/.git/hooks]

C:\Users\Иван\Desktop>
```

Рисунок 1.3 – Организация репозитория по модели ветвления git flow

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

Имя	Дата изменения	Тип
.idea	05.12.2022 19:00	Папка с файлами
venv	05.12.2022 18:58	Папка с файлами
example1	05.12.2022 18:59	JetBrains PyCharm
example2	05.12.2022 19:01	JetBrains PyCharm
example3	05.12.2022 19:00	JetBrains PyCharm
example4	05.12.2022 19:00	JetBrains PyCharm
example5	05.12.2022 19:01	JetBrains PyCharm

Рисунок 2.1 – Содержание папки common_files

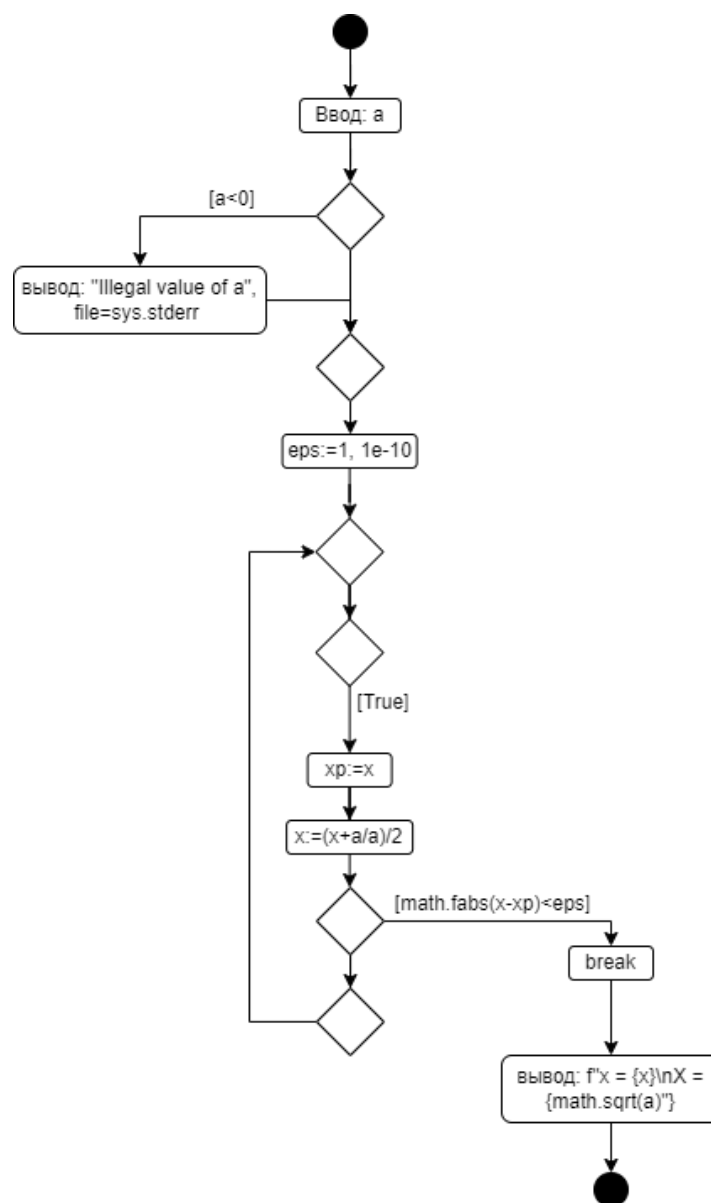


Рисунок 2.2 – UML-диаграмма программы для 4 примеры

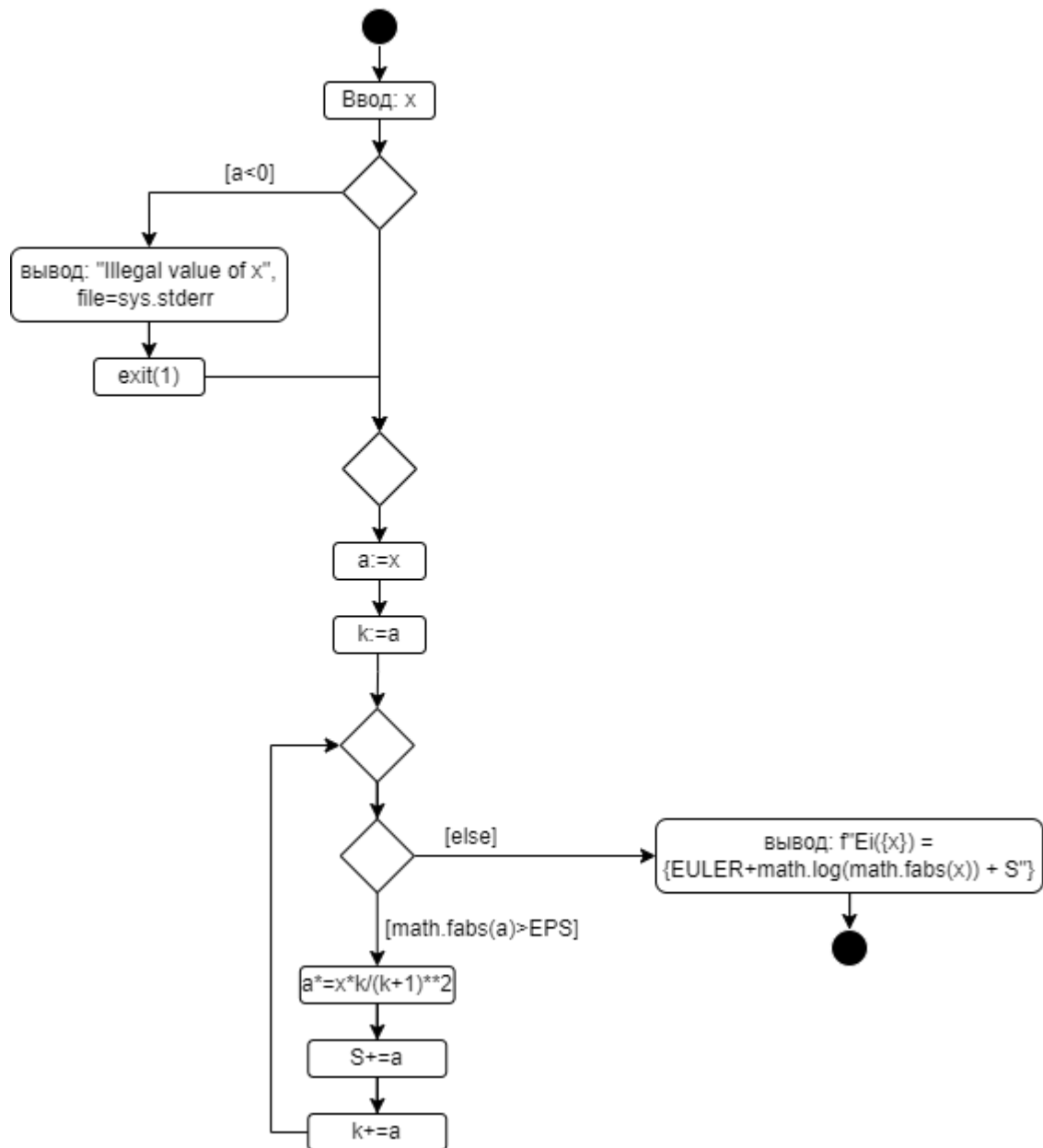


Рисунок 2.3 – UML-диаграмма для программы 5 примера

Индивидуальное задание

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

Вариант 10

Задание 1. Вводится число карандашей . Вывести фразу Я купил N карандашей , согласовав слово "карандаш" с числом .

Код программы:

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

if __name__ == '__main__':

    s = int(input())
    if s==1:
        print(f"Я купил {s} карандаш" )
    elif 1<s<5:
        print(f"Я купил {s} карандаша")
    elif 5<s<11:
        print(f"я купил {s} карандашей")
    elif s==0:
        print(f"я купил {s} карандашей")
```

```
C:\Users\Иван\Desktop\lab5_5\common_files\venv\Scripts\python
1
Я купил 1 карандаш
```

Рисунок 3.1 – Результат работы программы

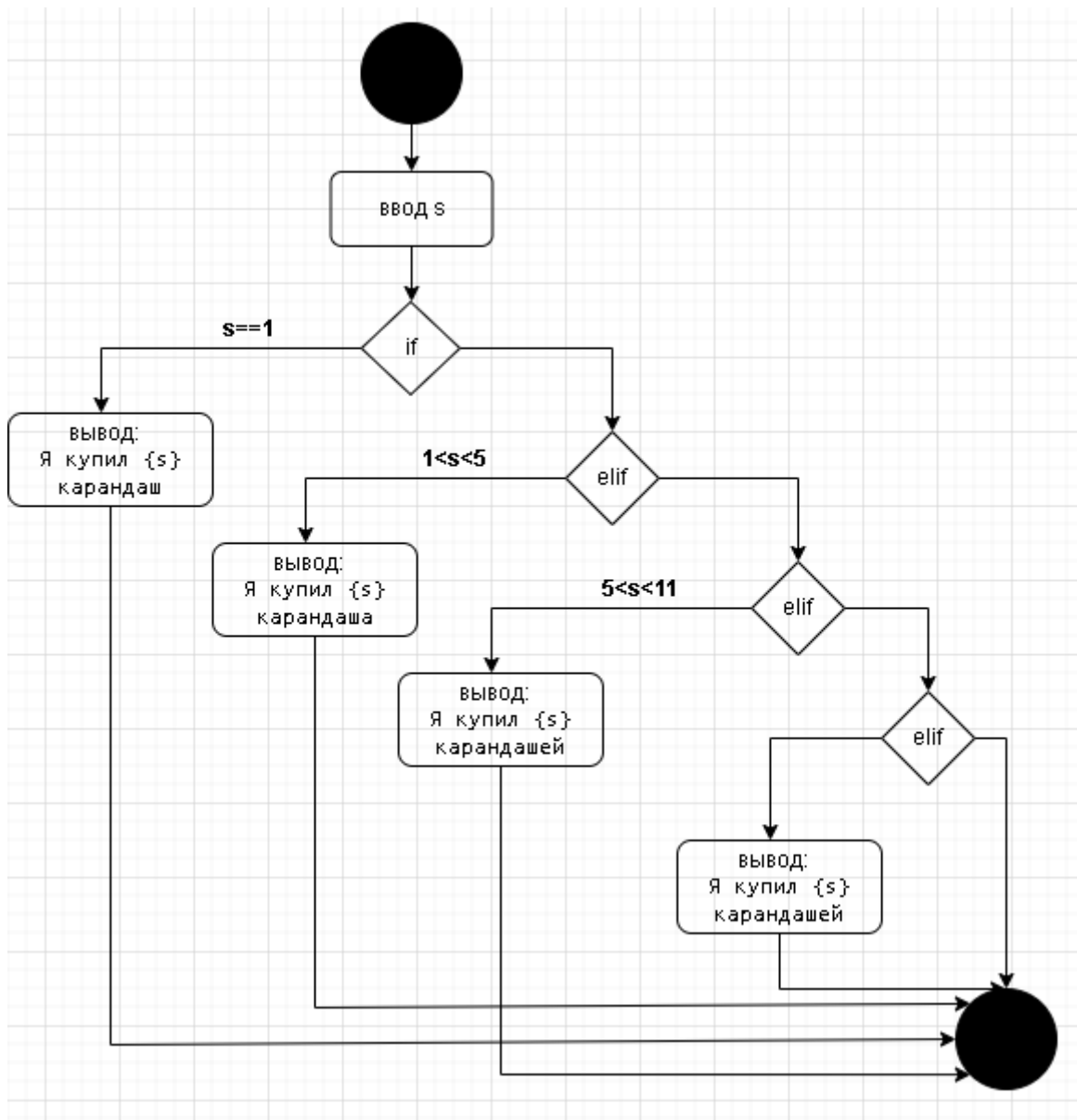


Рисунок 3.2 – UML диаграмма программы

Задание 2. Вывести на экран большее из трёх заданных чисел. Код программы:

```

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

if __name__ == '__main__':

    a, b, c = int(input()), int(input()), int(input())
    mx = a
    if b > mx:
        mx = b
    if c > mx:
        mx = c
    print(mx)
  
```

```

C:\Users\Иван\Desktop\lab5_5\common_files\venv\Scripts\python.exe C:/Users/Иван/Desktop/lab5_5/common_files/ind2.py
12
32
48
45
Process finished with exit code 0

```

Рисунок 3.3 – Результат работы программы

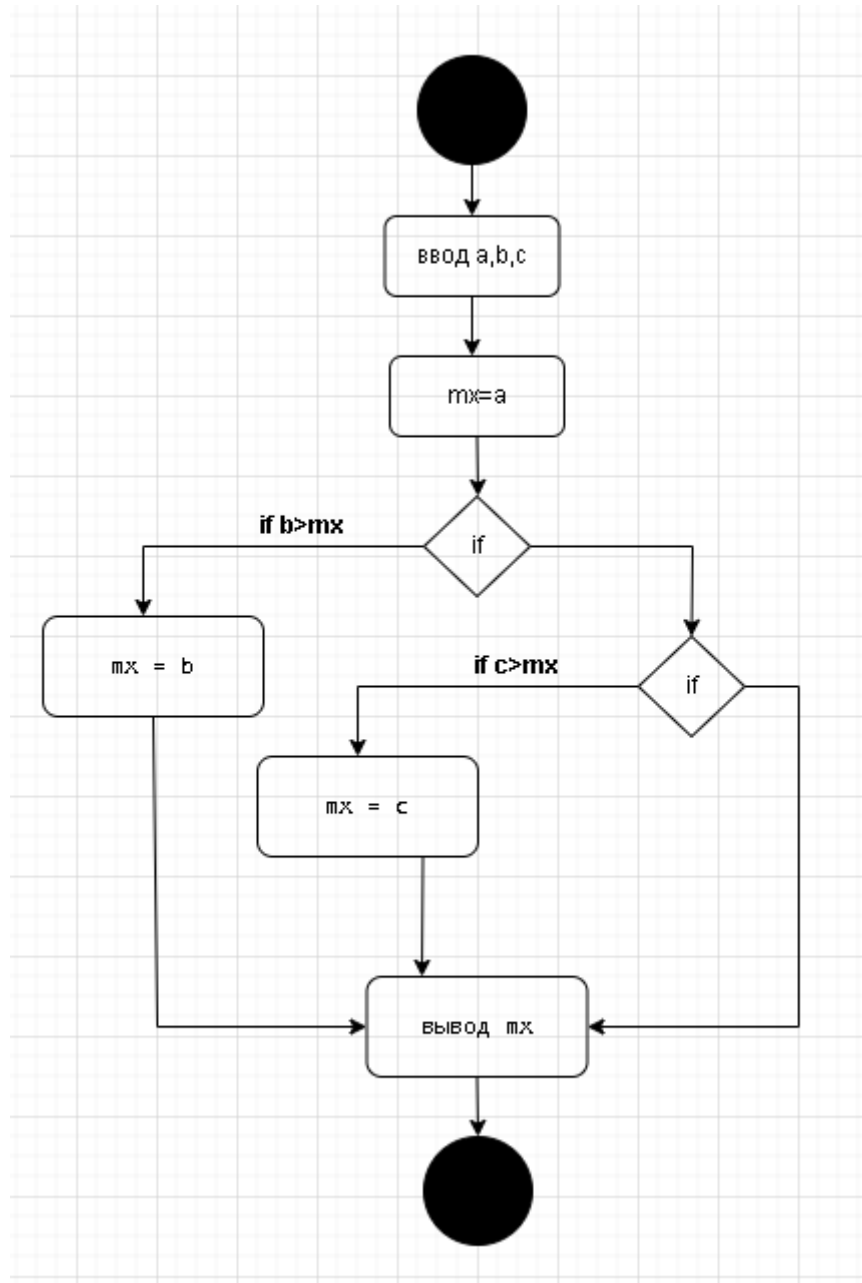


Рисунок 3.4 – UML диаграмма программы

Задание 3. Сколько можно купить быков, коров и телят, платя за быка 10 р., за корову - 5 р., а за теленка - 0,5 р., если на 100 р. надо купить 100 голов скота?

Код программы:

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
if __name__ == '__main__':
    for b in range(11):
        for k in range(21):
            t = 100 - b - k
            if 10*b+5*k+0.5*t==100: print(b, k, t)
```

```
C:\Users\Иван\Desktop\lab5_5\common_files\venv\Scripts\python.exe C:/Users/Иван/Desktop/lab5_5/common_files/ind3.py
1 9 90

Process finished with exit code 0
```

Рисунок 3.5 – Результат работы программы

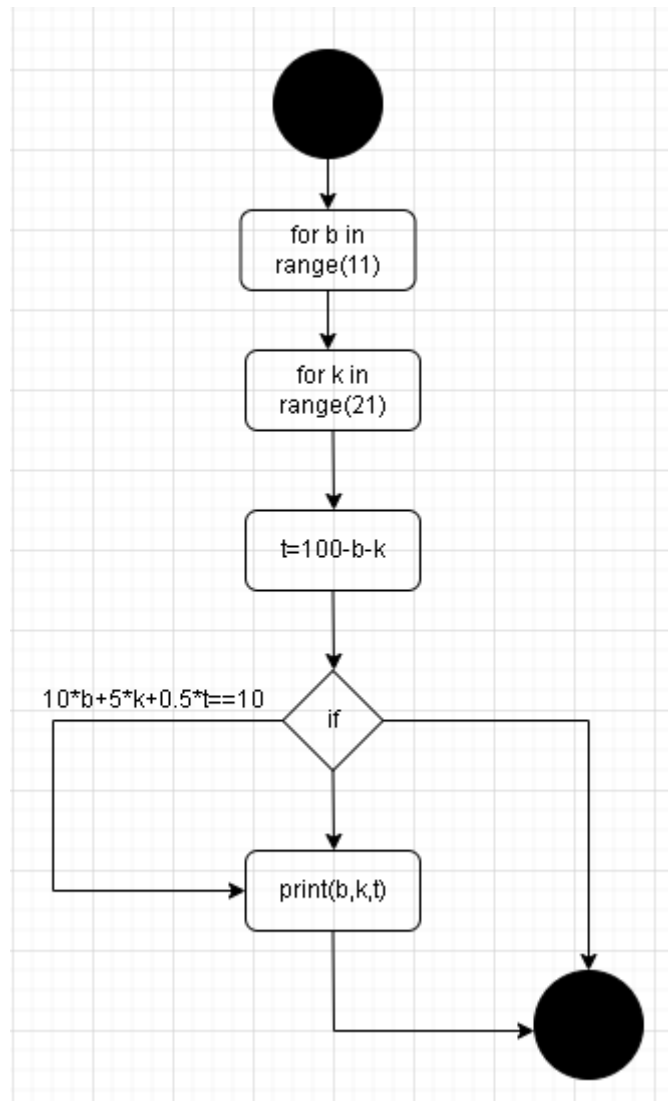


Рисунок 3.6 – UML диаграмма программы

Задание повышенной сложности

$$\text{Si}(x) = \int_0^x \frac{\sin t}{t} dt = \sum_{n=0}^{\infty} \frac{(-1)^n x^{2n+1}}{(2n+1)(2n+1)!}.$$

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

import math
import sys

# Calculation accuracy.
EPS = 1e-10

if __name__ == "__main__":
    x = float(input("Enter value of X: "))
    if x == 0:
        print("Illegal value of X", file=sys.stderr)
        exit(1)
    a = -3 * x ** 2 / 100
    Sum, n = a, 1
    # Find the sum of the terms of the series
    while math.fabs(a) > EPS:
        a *= (-x ** 2 * (2 * n + 1)) / ((2 * n + 3) ** 2 * (2 * n + 2))
        Sum += a
        n += 1
    print(Sum)
```

```
C:\Users\Иван\pythonProject1\venv\Scripts\python.exe C:/Users/Иван/pythonProject1/bhh.py
Enter value of X: 8
-0.43374238953511235

Process finished with exit code 0
|
```

Результат работы задания повышенной сложности

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.