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

Институт цифрового развития
Кафедра инфокоммуникаций

«Лабораторная работа 2.2 Условные операторы и циклы в языке Python»

ОТЧЕТ
по лабораторной работе №5
дисциплины
«Основы программной инженерии»

Выполнил:
Луценко Дмитрий Андреевич
2 курс, группа ПИЖ-б-о-21-1,
09.03.04 «Программная инженерия»,
направленность (профиль) «Разработка и сопровождение программного обеспечения», очная форма обучения

_____
(подпись)

Проверил:
_____
(подпись)

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2022 г.

Лабораторная работа 2.2 Условные операторы и циклы в языке Python.

Цель работы: исследование процесса установки и базовых возможностей языка Python

Ход работы:

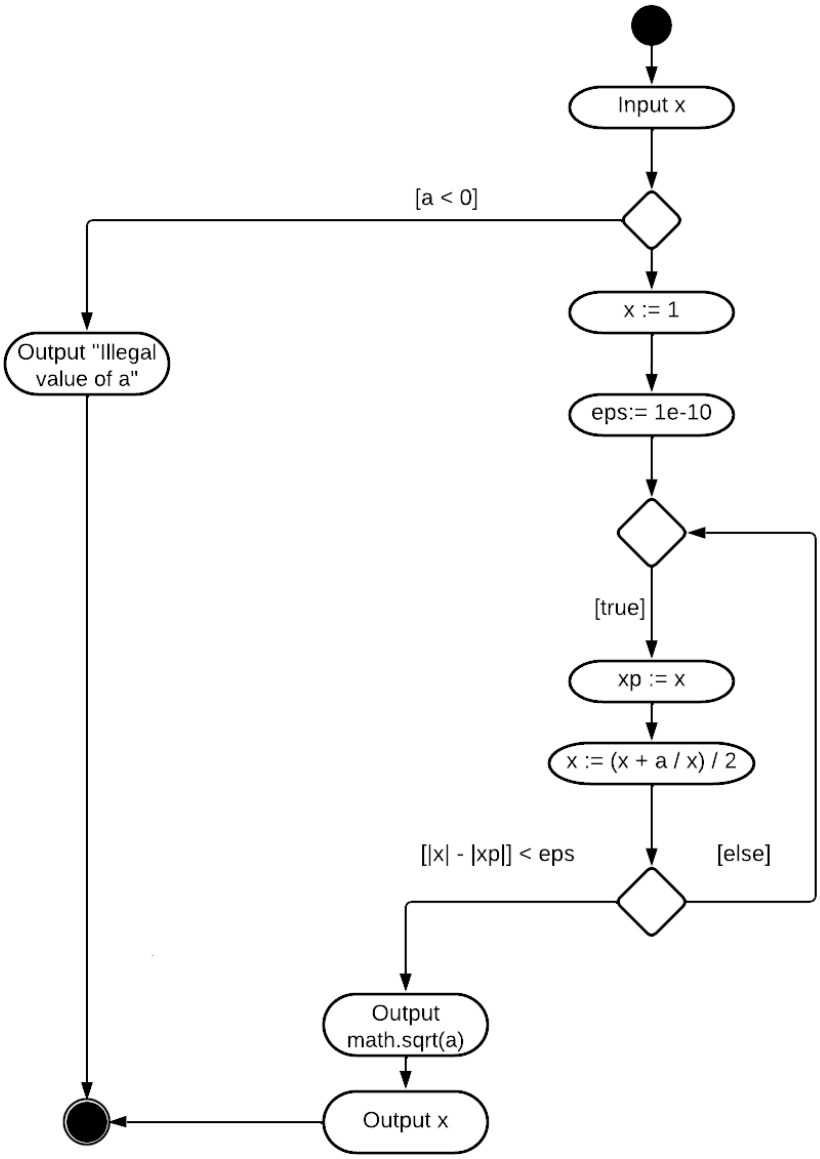

Рисунок 1 – UML-диаграмма деятельности для примера 4

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

import math
import sys


if __name__ == '__main__':
    a = float(input("Enter a: "))
    if a < 0:
        print("Illegal value of a", file=sys.stderr)
        exit(1)

    x, eps = 1, 1e-10
    while True:
        xp = x
        x = (x + a / x) / 2
        if math.fabs(x - xp) < eps:
            break
    print(f"x = {x}\nX = {math.sqrt(a)}")
```

```
Командная строка

E:\GitHub\5laba>4.py
Enter a: 4
x = 2.0
X = 2.0

E:\GitHub\5laba>
```

Рисунок 2 – Пример 4

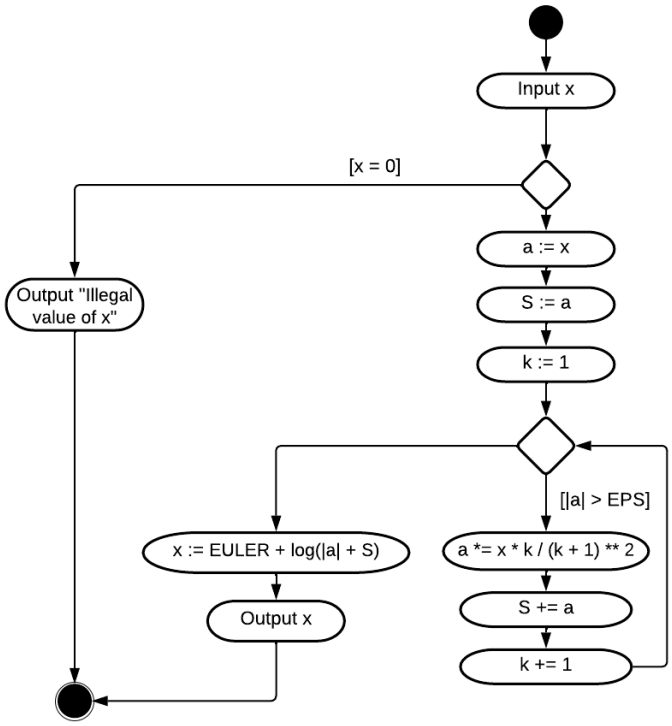

Рисунок 3 – UML-диаграмма деятельности для примера 5

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

import math
import sys


EULER = 0.5772156649015328606
EPS = 1e-10

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

    a = x
    S, k = a, 1

    while math.fabs(a) > EPS:
        a *= x * k / (k + 1) ** 2
        S += a
        k += 1

    print(f"Ei({x}) = {EULER + math.log(math.fabs(x)) + S}")
```

```
Командная строка

E:\GitHub\5laba>5.py
Enter x: 2
Ei(2.0) = 4.954234355999365

E:\GitHub\5laba>
```

Рисунок 4 – Пример 5

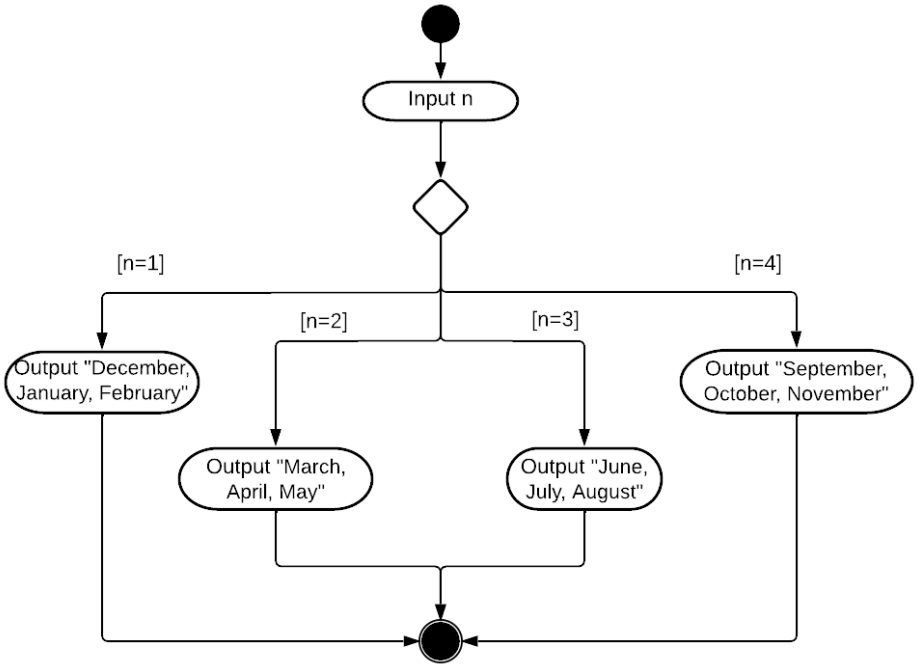

Рисунок 5 – UML-диаграмма деятельности для индивидуального задания 1

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

import sys


if __name__ == '__main__':
    print("Seasons:\n"
          "1 - winter\n"
          "2 - spring\n"
          "3 - summer\n"
          "4 - autumn\n")
    n = int(input("Enter season: "))

    if n == 1:
        print("December, January, February")
    elif n == 2:
        print("March, April, May")
    elif n == 3:
        print("June, July, August")
    elif n == 4:
        print("September, October, November")
    else:
        print("Error!", file=sys.stderr)
        exit(1)
```

```
Командная строка

E:\GitHub\5laba>individual1.py
Seasons:
1 - winter
2 - spring
3 - summer
4 - autumn

Enter season: 3
June, July, August

E:\GitHub\5laba>
```

Рисунок 6 – Индивидуальное задание 1

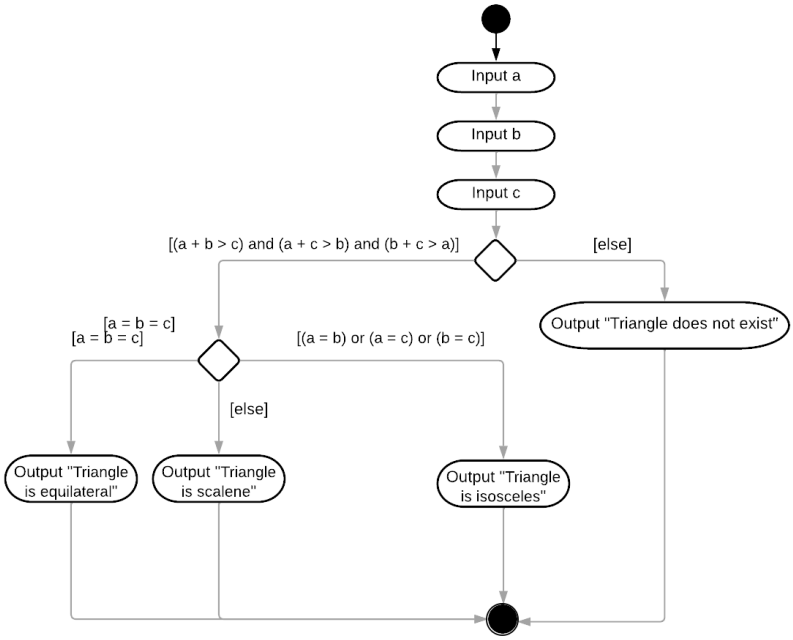

Рисунок 7 – UML-диаграмма деятельности для индивидуального задания 2

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

import sys


if __name__ == '__main__':
    a = float(input("Enter a: "))
    b = float(input("Enter b: "))
    c = float(input("Enter c: "))

    if (a + b > c) and (a + c > b) and (b + c > a):
        if a == b == c:
            print("Triangle is equilateral")
        elif (a == b) or (a == c) or (b == c):
            print("Triangle is isosceles")
        else:
            print("Triangle is scalene")
    else:
        print("Triangle does not exist")
```

```
Командная строка

E:\GitHub\5laba>individual2.py
Enter a: 5
Enter b: 4
Enter c: 2
Triangle is scalene

E:\GitHub\5laba>
```

Рисунок 8 – Индивидуальное задание 2

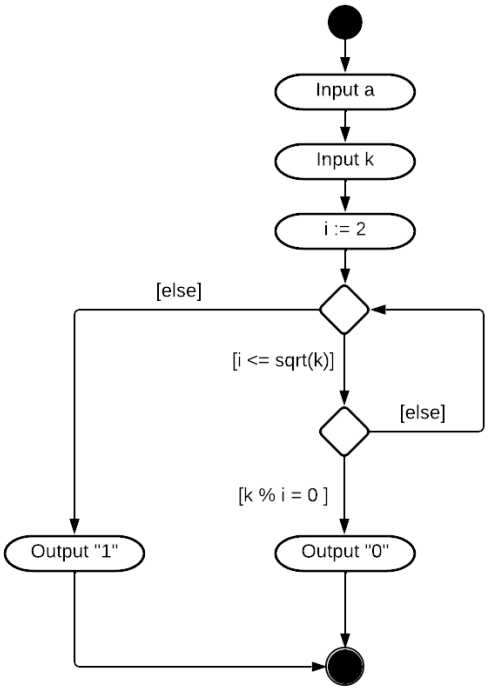

Рисунок 9 – UML-диаграмма деятельности для индивидуального задания 3

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

import sys
import math


if __name__ == '__main__':
    k = int(input("Enter k: "))

    i = 2
    while i <= math.sqrt(k):
        if k % i == 0:
            print("0")
            break
        i += 1
    else:
        print("1")
```

```
Командная строка

E:\GitHub\5laba>individual3.py
Enter k: 5
1

E:\GitHub\5laba>
```

Рисунок 10 – Индивидуальное задание 3

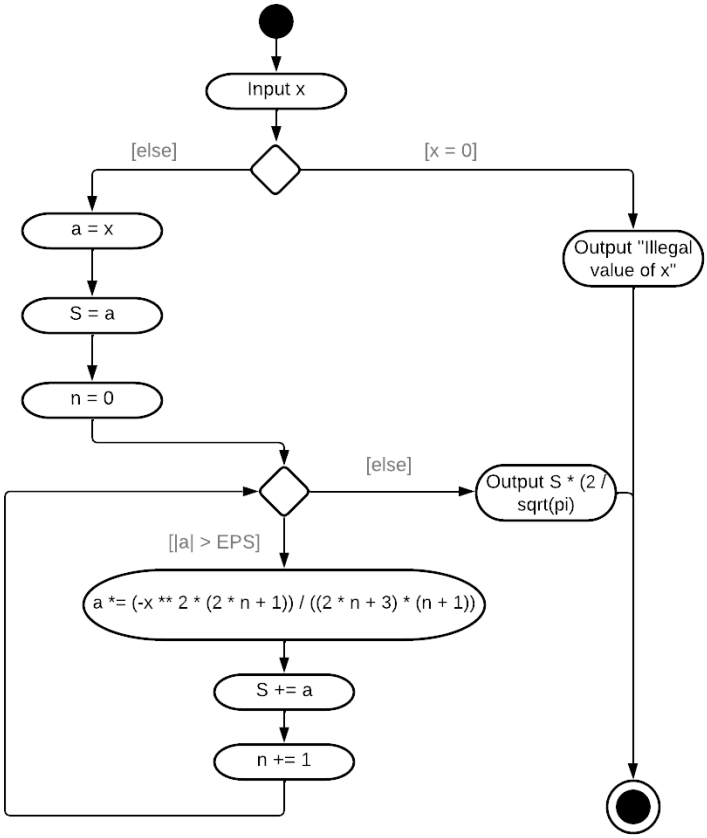

Рисунок 9 – UML-диаграмма деятельности для индивидуального повышенной сложности

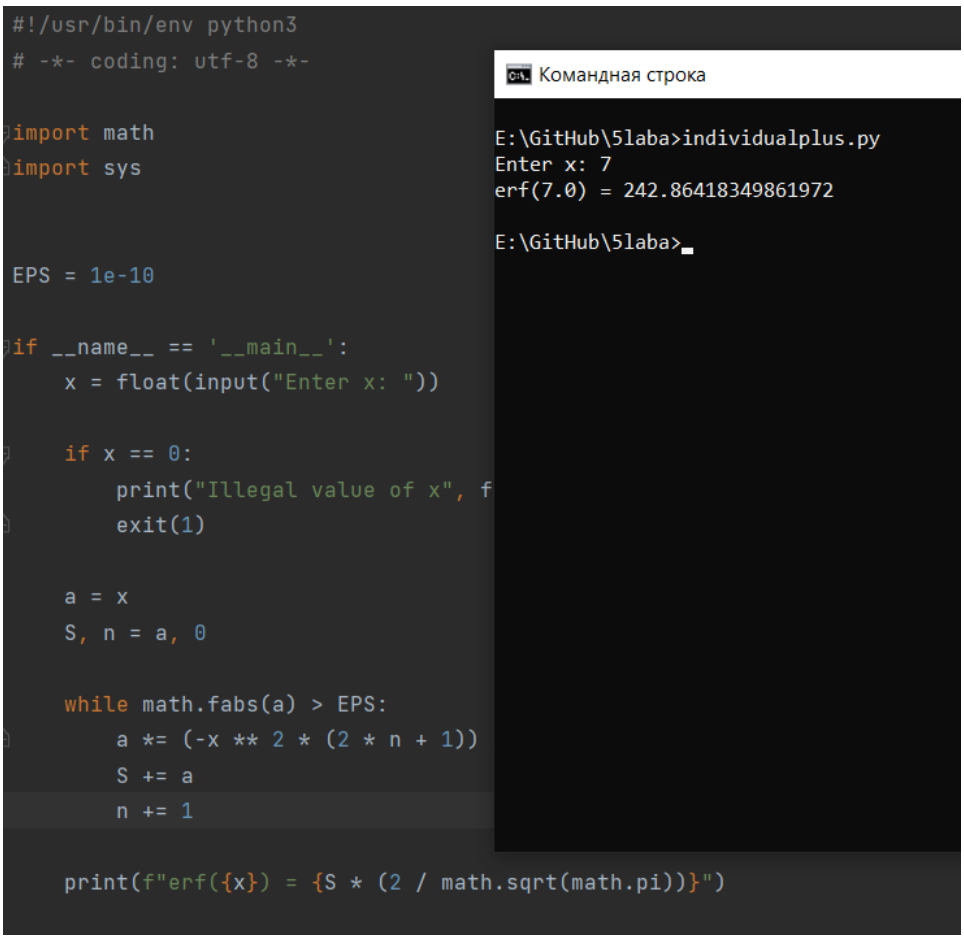

Рисунок 10 – Индивидуальное задание повышенной сложности

Ответы на контрольные вопросы:

1. **Для чего нужны диаграммы деятельности UML?** Диаграммы деятельности – это один из пяти видов диаграмм, применяемых в UML для моделирования динамических аспектов поведения системы. Диаграмма деятельности – это, по существу, блок-схема, которая показывает, как поток управления переходит от одной деятельности к другой,
2. **Что такое состояние действия и состояние деятельности?** Деятельность (Activity) – это продолжающийся во времени неатомарный шаг вычислений в автомате. Деятельности в конечном счете приводят к

выполнению некоего действия (Action), составленного из выполняемых атомарных вычислений, каждое из которых
либо изменяет состояние системы, либо возвращает какое-то значение.

3. **Какие нотации существуют для обозначения переходов и ветвлений в диаграммах деятельности?** Для переходов существуют линии или стрелки, а для ветвления используется ромб (он является точкой ветвления)
4. **Какой алгоритм является алгоритмом разветвляющейся структуры?** Им является алгоритм, в котором вычислительный процесс осуществляется либо по одной, либо по другой ветви, в зависимости от выполнения некоторого условия. Программа разветвляющейся структуры реализует такой алгоритм.
5. **Чем отличается разветвляющийся алгоритм от линейного?** Разветвляющийся алгоритм может пойти только по ветке, удовлетворяющей его условию. Линейный идет только «прямо»
6. **Что такое условный оператор? Какие существуют его формы?** Условный оператор позволяет выполнить определенный набор инструкций в зависимости от некоторого условия. If – elif - else
7. **Какие операторы сравнения используются в Python?** > (больше), < (меньше), >= (больше или равно), <= (меньше или равно), == (равно), != (не равно).
8. **Что называется простым условием? Приведите примеры.** Простое условие – условие, составленное из двух величин. Например, a > 5
9. **Что такое составное условие? Приведите примеры.** Составные условия – это условия, состоящие из нескольких простых условий. Например, (a > 7) or (b = 1)
10. **Какие логические операторы допускаются при составлении сложных условий?** And, or, not
11. **Может ли оператор ветвления содержать внутри себя другие ветвления?** Да, может

12. **Какой алгоритм является алгоритмом циклической структуры?** Алгоритм циклической структуры – это алгоритм, в котором происходит многократное повторение одного и того же участка программы.
13. **Типы циклов в языке Python.** While и for
14. **Назовите назначение и способы применения функции range.** Функция range возвращает неизменяемую последовательность чисел в виде объекта range. Применяется в циклах for
15. **Как с помощью функции range организовать перебор значений от 15 до 0 с шагом 2?** For I in range (15, 0, -2)
16. **Могул ли быть циклы вложенными?**
17. **Как образуется бесконечный цикл и как выйти из него?** Бесконечный цикл образуется тогда, когда условие для выхода из него не выполняется. Выйти из него можно с помощью оператора break
18. **Для чего нужен оператор break ?** Оператор break нужен для досрочного завершения цикла
19. **Где употребляется оператор continue и для чего он используется?** Оператор continue применяется в циклах для их остановки и повторного запуска. При этом код после оператора не выполняется
20. **Для чего нужны стандартные потоки stdout и stderr?** Stdout нужен для вывода данных и информационных сообщений, а поток stderr для вывода сообщений об ошибках.
21. **Как в Python организовать вывод в стандартный поток stderr?** file=sys.stderr
22. **Каково назначение функции exit ?** Функция exit применяется для завершения программы и передачи кода возврата операционной системе