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

Институт цифрового развития
Кафедра инфокоммуникаций

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №5
дисциплины «Программирование на Python»
Вариант 5.

Выполнила:
Михеева Елена Александровна
2 курс, группа ИВТ-б-з-20-1,
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...,

_____
(подпись)

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2023 г.

Тема: Условные операторы и циклы в языке Python.

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.

Порядок выполнения работы:

1. Запустили и протестировали с различными входными данными программы из примеров 1-5.

```
(base) MacBook-Pro-Elena:Python_2.2 mikheeva$ python3 example1.py
Value of x? -5
y = 50.28366218546323
(base) MacBook-Pro-Elena:Python_2.2 mikheeva$ python3 example1.py
Value of x? 3
y = 4.0
(base) MacBook-Pro-Elena:Python_2.2 mikheeva$ python3 example1.py
Value of x? 8
y = -63.01064175337662
```

Рисунок 1. Запуск программы примера 1.

```
(base) MacBook-Pro-Elena:Python_2.2 mikheeva$ python3 example2.py
Введите номер месяца: 12
Зима
(base) MacBook-Pro-Elena:Python_2.2 mikheeva$ python3 example2.py
Введите номер месяца: 67
Ошибка!
(base) MacBook-Pro-Elena:Python_2.2 mikheeva$ python3 example2.py
Введите номер месяца: 9
Осень
```

Рисунок 2. Запуск программы примера 2.

```
(base) MacBook-Pro-Elena:Python_2.2 mikheeva$ python3 example3.py
Value of n? 4
Value of x? 7.89
S = 3.322603833435919
(base) MacBook-Pro-Elena:Python_2.2 mikheeva$ python3 example3.py
Value of n? 7
Value of x? 0.9299
S = 0.4259845626069086
```

Рисунок 3. Запуск программы примера 3.

```
(base) MacBook-Pro-Elena:Python_2.2 mikheeva$ python3 example4.py
Value of a? -4
Illegal value of a
(base) MacBook-Pro-Elena:Python_2.2 mikheeva$ python3 example4.py
Value of a? 0.0562
x = 0.23706539182259395
X = 0.23706539182259395
```

Рисунок 4. Запуск программы примера 4.

```
(base) MacBook-Pro-Elena:Python_2.2 mikheeva$ python3 example5.py
Value of x? 0
Illegal value of x
(base) MacBook-Pro-Elena:Python_2.2 mikheeva$ python3 example5.py
Value of x? 4.0912
Ei(4.0912) = 20.919385514594495
```

Рисунок 5. Запуск программы примера 5.

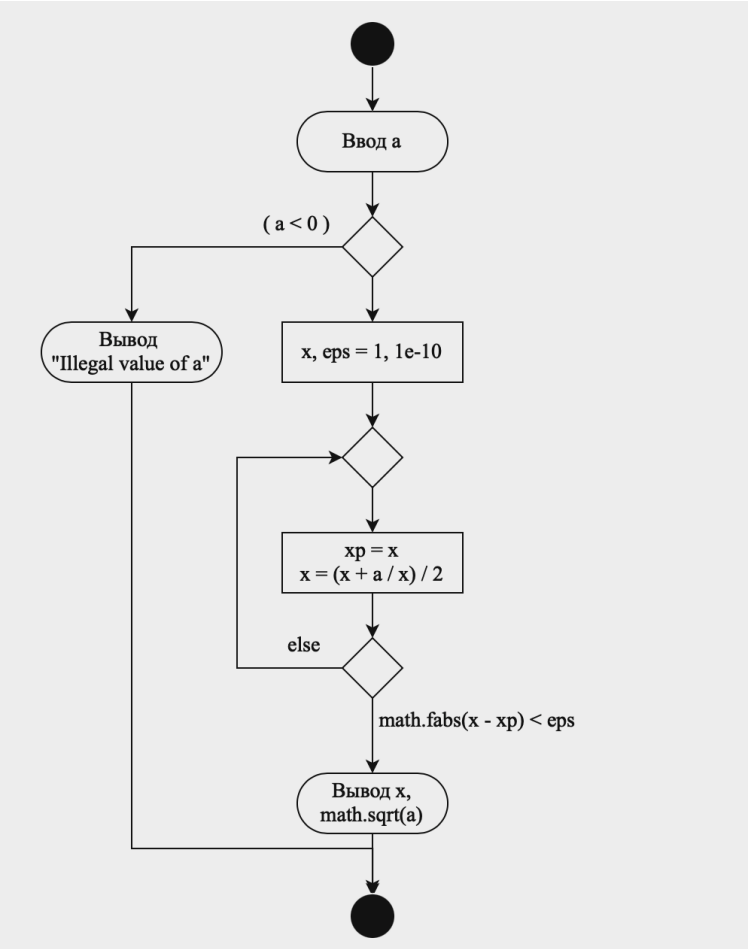

Рисунок 6. Диаграмма для примера №4

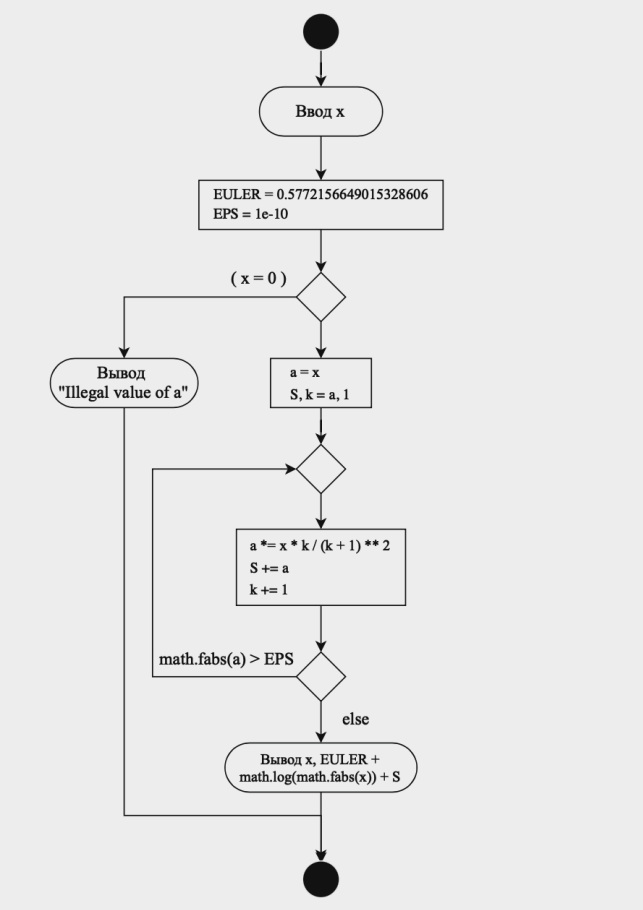

Рисунок 7. Диаграмма для примера №5

2. Приступили к выполнению индивидуальных заданий согласно Варианту №5.

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

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

import sys

if __name__ == '__main__':
    m = int(input("Enter the season number... "))
    if (m < 1) or (m > 4):
        print("Illegal value of m", file=sys.stderr)
        exit(1)
    if m == 1:
        print("Зима: \nдекабрь, январь, февраль.")
    elif m == 2:
        print("Весна: \nмарт, апрель, май.")
    elif m == 3:
        print("Лето: \nиюнь, июль, август.")
    elif m == 4:
        print("Осень: \nсентябрь, октябрь, ноябрь.")

```

Рисунок 8. Индивидуальное задание №1

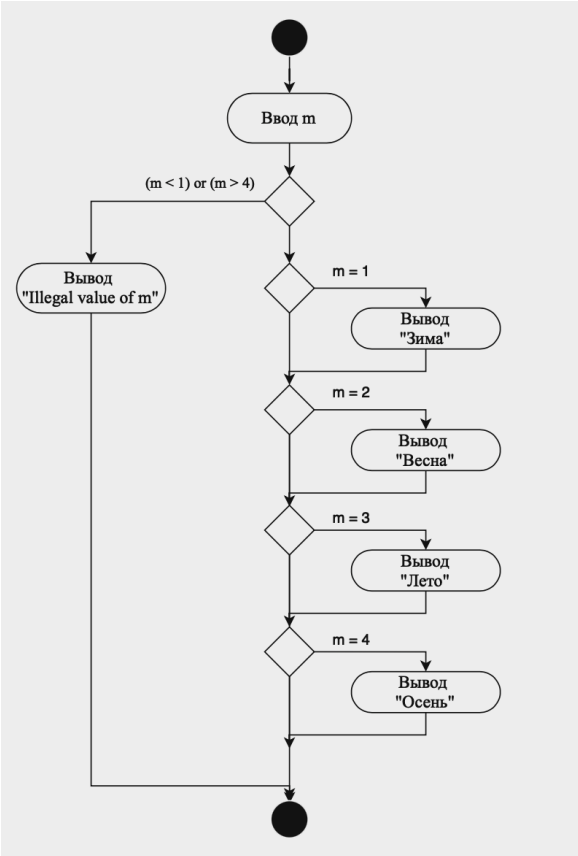

Рисунок 9. Диаграмма индивидуального задания №1

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

# Задание №2
# Определить принадлежит ли точка кольцу определяемому окружностями
# x^2 + y^2 = 1 и x^2 + y^2 = 0.25

if __name__ == '__main__':
    x = float(input("Enter x... "))
    y = float(input("Enter y... "))
    if ((x ** 2 + y ** 2) <= 1) and ((x ** 2 + y ** 2) >= 0.25):
        print(f"The point ({x}, {y}) belongs to the ring!")
    else:
        print(f"The point ({x}, {y}) does not belongs to the ring!")

```

Рисунок 10. Индивидуальное задание №2

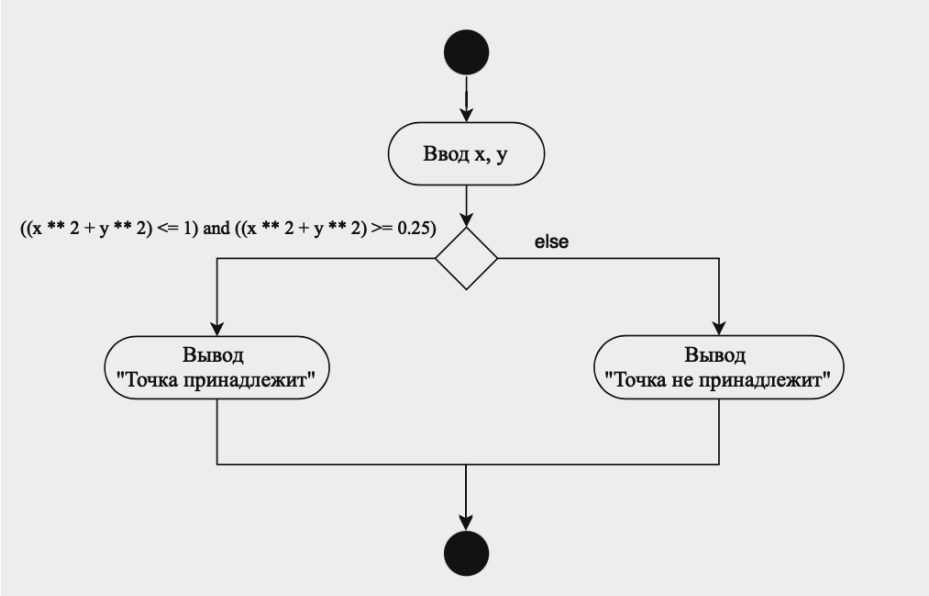

Рисунок 11. Диаграмма индивидуального задания №2

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

# Задание №3
# Одноклеточная амеба каждые три часа делится на 2 клетки.
# Определить, сколько будет клеток через 6 часов.


def ameba(n, hours):
    while hours >= 3:
        n += n
        hours -= 3
    return n


if __name__ == '__main__':
    n = int(input("Enter the number of amebas... "))
    hours = int(input("Enter the hours... "))
    print(f"The number of {n} amoebas after {hours} hours is", end=" ")
    print(ameba(n, hours))
```

```
ПРОБЛЕМЫ  ВЫХОДНЫЕ ДАННЫЕ  ТЕРМИНАЛ  ···  zsh

(base) mikheeva@MacBook-Pro-Elena Python_2.2 % python3 individual_3.py
Enter the number of amebas... 1
Enter the hours... 6
The number of 1 amoebas after 6 hours is 4
```

Рисунок 12. Индивидуальное задание №3

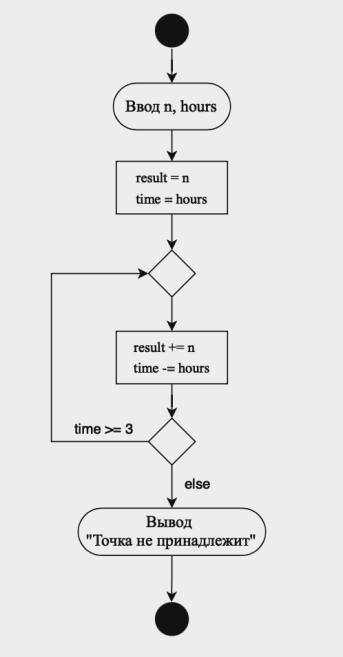

Рисунок 13. Диаграмма индивидуального задания №3

3. Выполнили задание повышенной сложности: составить UML-диаграмму деятельности, программу и произвести вычисление значения специальной функции по ее разложению в ряд с точностью $\varepsilon = 10^{-10}$, аргумент функции вводится с клавиатуры.

5. Первый интеграл Френеля:

$$C(x) = \int_0^x \cos\left(\frac{\pi}{2}t^2\right) dt = \sum_{n=0}^{\infty} \frac{(-1)^n (\pi/2)^{2n}}{(2n)!(4n+1)}.$$

Общий член последовательности:

$$a_n = \frac{(-1)^n (\pi/2)^{2n} x^{(4n+1)}}{(2n)!\,(4n+1)}$$

Рекуррентное отношение членов последовательности:

$$a_{n+1} = -\frac{(\pi/2)^2 (4n+1) x^4}{(4n^2+6n+2)(4n+5)} a_n$$

Нулевой член ряда: x.

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

# Задание повышенной сложности
# Вычисление значения специальной функции по ее разложению в ряд
# Первый интеграл Френеля

import math
import sys

if __name__ == '__main__':
    EPS = 1e-10
    x = float(input("Enter x... "))

    if x == 0:
        print("Illegal value of n", file=sys.stderr)
        exit(1)

    result = x
    sum = x
    n = 1

    while math.fabs(result) > EPS:
        result *= (-(x**4)*(math.pi/2)**2 * (4))/((4*n**2 + 6*n + 2)*(4*n + 5))
        sum += result
        n += 1
    print(sum)
```

Рисунок 14. Задание повышенной сложности

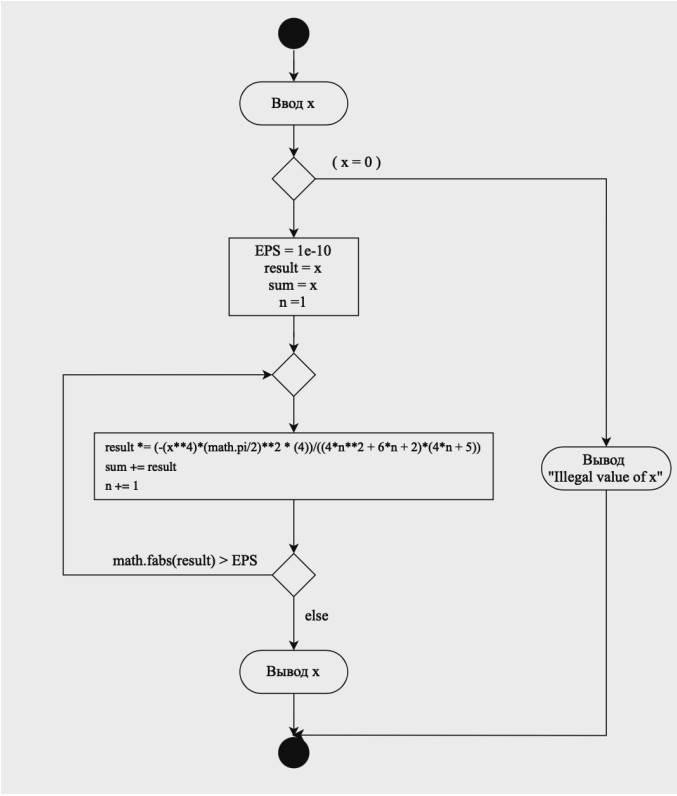

Рисунок 15. Диаграмма для индивидуального задания повышенной сложности

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, однако, по сравнению с последней, у ней есть явные преимущества: поддержка многопоточности и объектно-ориентированного проектирования.

2. Что такое состояние действия и состояние деятельности?

В потоке управления, моделируемом диаграммой деятельности, происходят различные события. Можно вычислить выражение, в результате чего изменяется значение некоторого атрибута или возвращается некоторое значение. Также, например, можно выполнить операцию над объектом, послать ему сигнал или даже создать его или уничтожить. Все эти выполняемые атомарные вычисления называются состояниями действия, поскольку каждое из них есть состояние системы, представляющее собой выполнение некоторого действия.

Состояния действия не могут быть подвергнуты декомпозиции. Кроме того, они атомарны. Это значит, что внутри них могут происходить различные события, но выполняемая в состоянии действия работа не может быть прервана. Обычно предполагается, что длительность одного состояния действия занимает неощутимо малое время.

В противоположность этому состояния деятельности могут быть подвергнуты дальнейшей декомпозиции, вследствие чего выполняемую деятельность можно представить с помощью других диаграмм деятельности. Состояния деятельности не являются атомарными, то есть могут быть прерваны. Предполагается, что для их завершения требуется заметное время. Можно считать, что состояние действия — это частный вид состояния деятельности, а конкретнее - такое состояние, которое не может быть подвергнуто дальнейшей декомпозиции. А 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?

Когда действие или деятельность в некотором состоянии завершается, поток управления сразу переходит в следующее состояние действия или деятельности. Для описания этого потока используются переходы, показывающие путь из одного состояния действия или деятельности в другое. В UML переход представляется простой линией со стрелкой.

Поток управления должен где-то начинаться и заканчиваться (разумеется, если это не бесконечный поток, у которого есть начало, но нет конца). Можно задать как начальное состояние (закрашенный кружок), так и конечное (закрашенный кружок внутри окружности).

4. Какой алгоритм является алгоритмом разветвляющейся структуры?

Простые последовательные переходы встречаются наиболее часто, но их одних недостаточно для моделирования любого потока управления. Как и в блок-схеме, вы можете включить в модель ветвление, которое описывает различные пути выполнения в зависимости от значения некоторого булевского выражения. Как видно из рис. 4.3, точка ветвления представляется ромбом. В точку ветвления может входить ровно один переход, а выходить - два или более. Для каждого исходящего перехода задается булевское выражение, которое вычисляется только один раз при входе в точку ветвления. Ни для каких двух исходящих переходов эти сторожевые условия не должны одновременно принимать значение «истина», иначе поток управления окажется неоднозначным. Но эти условия должны покрывать все возможные варианты, иначе поток остановится.

Для удобства разрешается использовать ключевое слово else для пометки того из исходящих переходов, который должен быть выбран в случае, если условия, заданные для всех остальных переходов, не выполнены.

Реализовать итерацию можно, если ввести два состояния действия - в первом устанавливается значение счетчика, во втором оно увеличивается - и точку ветвления, вычисление в которой показывает, следует ли прекратить итерации.

5. Чем отличается разветвляющийся алгоритм от линейного?

Линейный алгоритм - алгоритм, все этапы которого выполняются однократно и строго последовательно.

Разветвляющийся алгоритм – это 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. В программе разветвляющейся структуры имеется один или несколько условных операторов. Для программной реализации условия используется логическое выражение. В сложных структурах с большим числом ветвей применяют оператор выбора.

6. Что такое условный оператор? Какие существуют его формы?

Условные операторы – это специальные конструкции, благодаря которым в программе происходит ветвление в зависимости от условий. Условный оператор имеет полную и краткую формы.

7. Какие операторы сравнения используются в Python?

If, elif, else.

8. Что называется простым условием? Приведите примеры.

Простое условие в программировании — это выражение, которое может быть истинным или ложным. Оно используется для принятия решений в коде на основе значения переменных или других условий.

9. Что такое составное условие? Приведите примеры.

Составное условие – логическое выражение, содержащее несколько

простых условий, объединенных логическими операциями. Это операции not, and, or.

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? Алгоритм циклической структуры — это алгоритм, в котором происходит многократное повторение одного и того же участка программы. Такие повторяемые участки вычислительного процесса называются циклами.

13. Типы циклов в языке Python.

В Python есть 2 типа циклов: - цикл while, - цикл for.

Оператор for выполняет указанный набор инструкций заданное

количество раз, которое определяется количеством элементов в наборе. Оператор цикла while выполняет указанный набор инструкций до тех пор, пока условие цикла истинно.

14. Назовите назначение и способы применения функции range. Функция range возвращает неизменяемую последовательность чисел в виде объекта range.

Параметры функции:

start - с какого числа начинается последовательность. По умолчанию - 0

stop - до какого числа продолжается последовательность чисел. Указанное число не включается в диапазон

step - с каким шагом растут числа. По умолчанию 1

15. Как с помощью функции range организовать перебор значений от 15 до 0 с шагом 2?

for i in 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. Чтобы выйти из цикла нужно использовать оператор break.

18. Для чего нужен оператор break?

Оператор break предназначен для досрочного прерывания работы цикла while.

19. Где употребляется оператор continue и для чего он используется?

Оператор continue запускает цикл заново, при этом код, расположенный после данного оператора, не выполняется.

20. Для чего нужны стандартные потоки stdout и stderr?

Ввод и вывод распределяется между тремя стандартными потоками: stdout — стандартный вывод (экран), stderr — стандартная ошибкаи (вывод ошибок на экран)

21. Как в Python организовать вывод в стандартный поток stderr? Во-первых нужно импортировать sys, а дальше использовать print(...,

file=sys.stderr).

22. Каково назначение функции exit?

Функция exit() модуля sys - выход из Python.