

РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное
образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Кафедра инфокоммуникаций

Отчет по лабораторной работе № 3
по дисциплине «Программирование на Python»

Выполнил студент группы ИВТ-б-о-24-1:

Хубиев Роберт Эльбрусович

«28» декабря 2025г.

Подпись студента ____ Хубиев ____

Работа защищена « » ____ 2025г.

Проверил Воронкин Р.А. ____

(подпись)

Ставрополь 2025

Ссылка на репозиторий: <https://github.com/truebobsuncle/LAR3>

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.

Примеры:

Пример 4. Найти значение квадратного корня $x = \sqrt{a}$ из положительного числа a , вводимого с клавиатуры, с некоторой заданной точностью ε с помощью рекуррентного соотношения:

$$x_{n+1} = \frac{1}{2} \cdot \left(x_n + \frac{a}{x_n} \right).$$

В качестве начального значения примем $x_0 = 1$. Цикл должен выполняться до тех пор, пока не будет выполнено условие $|x_{n+1} - x_n| \leq \varepsilon$. Сравните со значением квадратного корня, полученным с использованием функций стандартной библиотеки. Значение $\varepsilon = 10^{-10}$.

Решение: Напишем программу для решения поставленной задачи.

Рисунок 1. Пример 4

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

import math
import sys

if __name__ == '__main__':
    a = float(input("Value of 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}")
    print(f"x = {math.sqrt(a)}")
```

```

C:\anaconda3\python.exe C:\Users\Роберт\Desktop\LAR3\src\example1.py
Value of a? 25
x = 5.0
x = 5.0

Process finished with exit code 0

```

Рисунок 2. Результат выполнения кода

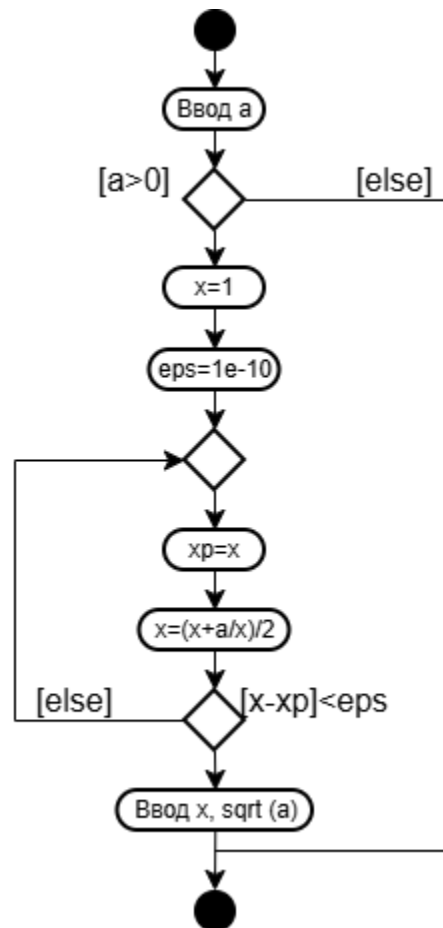


Рисунок 3. UML-диаграмма для 4 примера

Пример 5. Вычислить значение специальной (интегральной показательной) функции

$$\text{Ei}(x) = \int_{-\infty}^x \frac{\exp t}{t} dt = \gamma + \ln x + \sum_{k=1}^{\infty} \frac{x^k}{k \cdot k!},$$

где $\gamma = 0.5772156649 \dots$ - постоянная Эйлера, по ее разложению в ряд с точностью $\varepsilon = 10^{-10}$, аргумент x вводится с клавиатуры.

Рисунок 5. Пример 5

```

#!/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, k = a, 1

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

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

```

```

C:\anaconda3\python.exe C:\Users\Роберт\Desktop\LAR3\ex5.py
Value of x? 5
Ei(5.0) = 40.18527535579794

Process finished with exit code 0

```

Рисунок 6. Результат выполнения кода

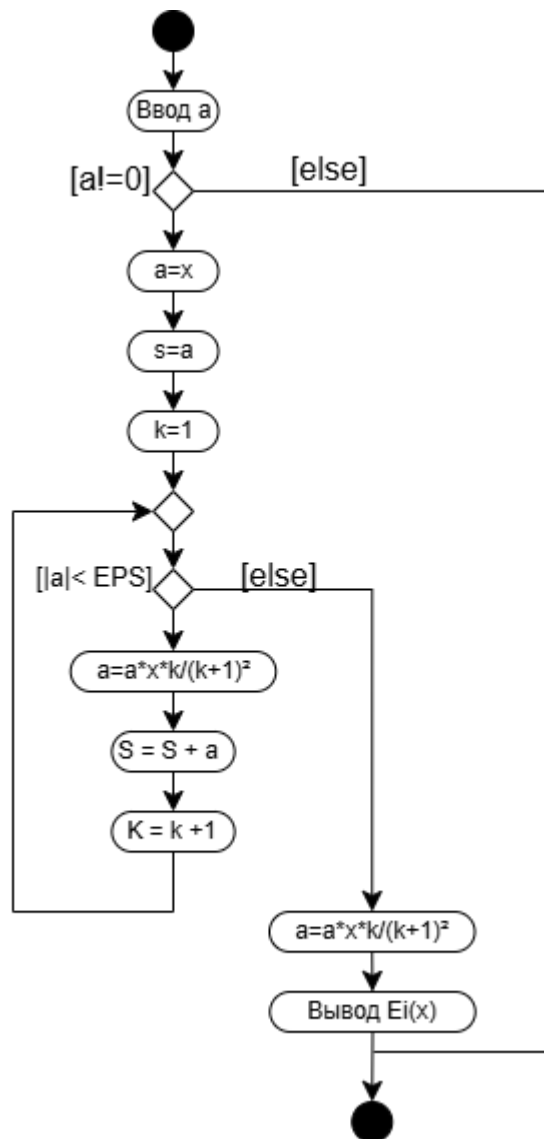


Рисунок 7. UML-диаграмма для 5 примера

Индивидуальные задания

Задание 1

Решить задачу согласно варианта, составить UML-диаграмму деятельности и программу с использованием конструкций ветвления. Номер варианта необходимо получить у преподавателя.

3. Дано число m ($1 < m \leq 7$). Вывести на экран название дня недели, который соответствует этому номеру

```

days = ["Понедельник", "Вторник", "Среда", "Четверг", "Пятница", "Суббота",
"Воскресенье"]
m = int(input("Номер дня: "))
print(days[m-1])

```

```

C:\anaconda3\python.exe C:\Users\Роберт\Desktop\LAR3\ind1.py
Номер дня: 6
Суббота
Process finished with exit code 0

```

Рисунок 8. Результат выполнения кода

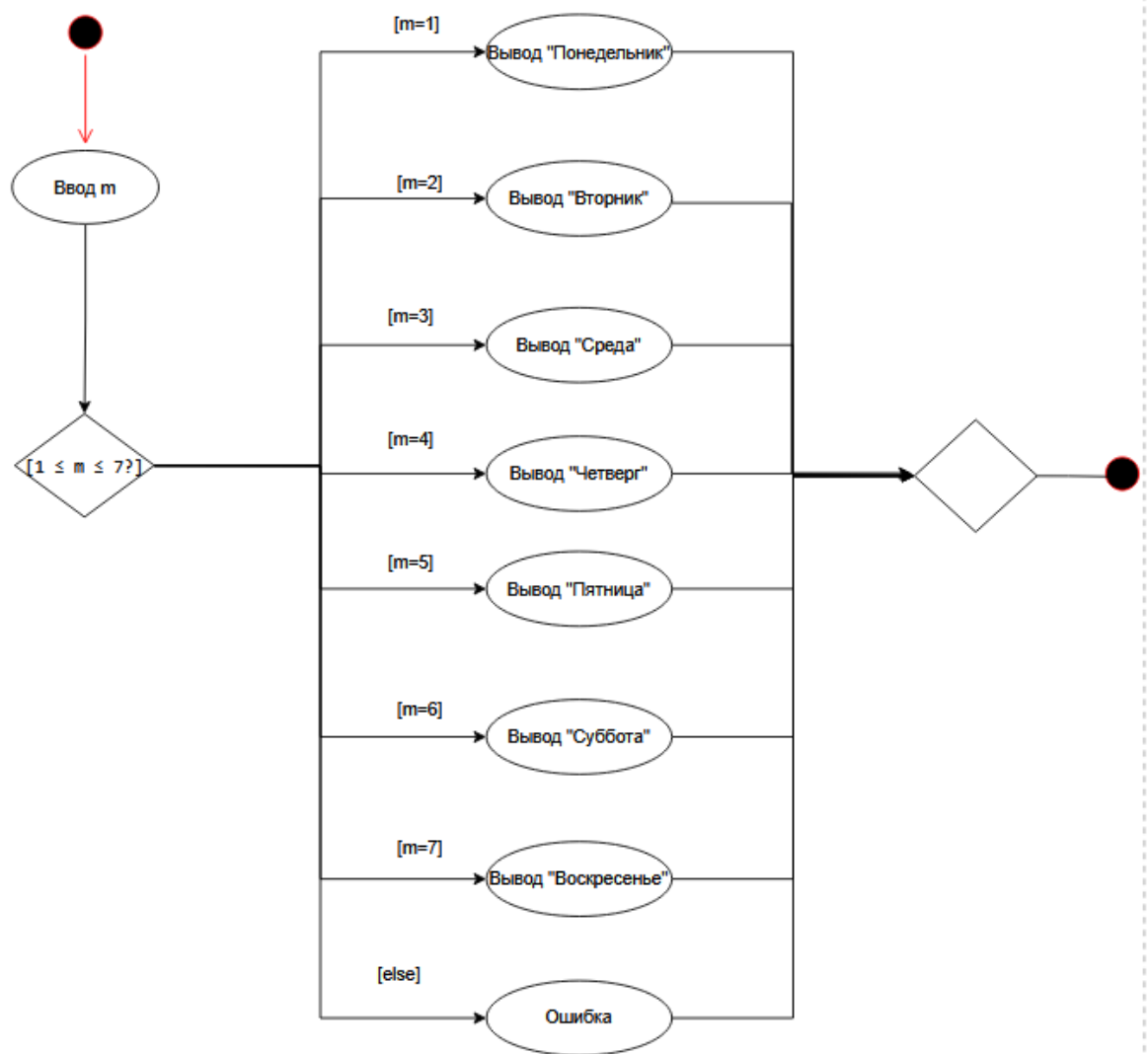


Рисунок 9. UML-диаграмма для 1 задания

Задание 2

Из трех действительных чисел a , b и c выбрать те, модули которых не меньше 4.

```

a = float(input("a = "))
b = float(input("b = "))
c = float(input("c = "))

```

```
print("Числа, модуль которых  $\geq 4$ :")  
  
if abs(a) >= 4:  
    print(a)  
if abs(b) >= 4:  
    print(b)  
if abs(c) >= 4:  
    print(c)
```

```
C:\anaconda3\python.exe C:\Users\Роберт\Desktop\LAR3\ind2.py  
a = 34  
b = 21  
c = 3  
Числа, модуль которых  $\geq 4$ :  
34.0  
21.0
```

Рисунок 10. Код выполненной программы

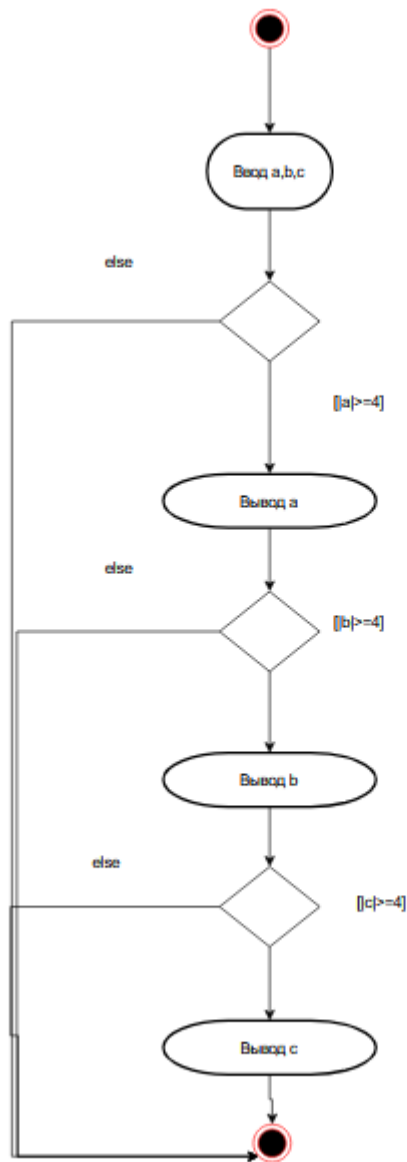


Рисунок 11. UML-диаграмма для задания 2

Задание 3

Начав тренировки, спортсмен пробежал 10 км. Каждый следующий день он увеличивал дневную норму на 10% от нормы предыдущего дня. Какой суммарный путь пробежит спортсмен за 7 дней?

```

if __name__ == '__main__':
    # Начальная дистанция
    daily_distance = 10.0 # км
    total_distance = 0.0

    print("Расчет дистанции за 7 дней:")
    print(f"День 1: {daily_distance:.2f} км")

    total_distance += daily_distance
  
```



```
# Расчет для дней 2-7
for day in range(2, 8):
    daily_distance = daily_distance * 1.10 # Увеличение на 10%
    total_distance += daily_distance
    print(f"День {day}: {daily_distance:.2f} км")

print(f"\nСуммарный путь за 7 дней: {total_distance:.2f} км")
```

```
C:\anaconda3\python.exe C:\Users\Роберт\Desktop\LAR3\ind3
Расчет дистанции за 7 дней:
День 1: 10.00 км
День 2: 11.00 км
День 3: 12.10 км
День 4: 13.31 км
День 5: 14.64 км
День 6: 16.11 км
День 7: 17.72 км

Суммарный путь за 7 дней: 94.87 км

Process finished with exit code 0
```

Рисунок 12. Результат выполнения кода

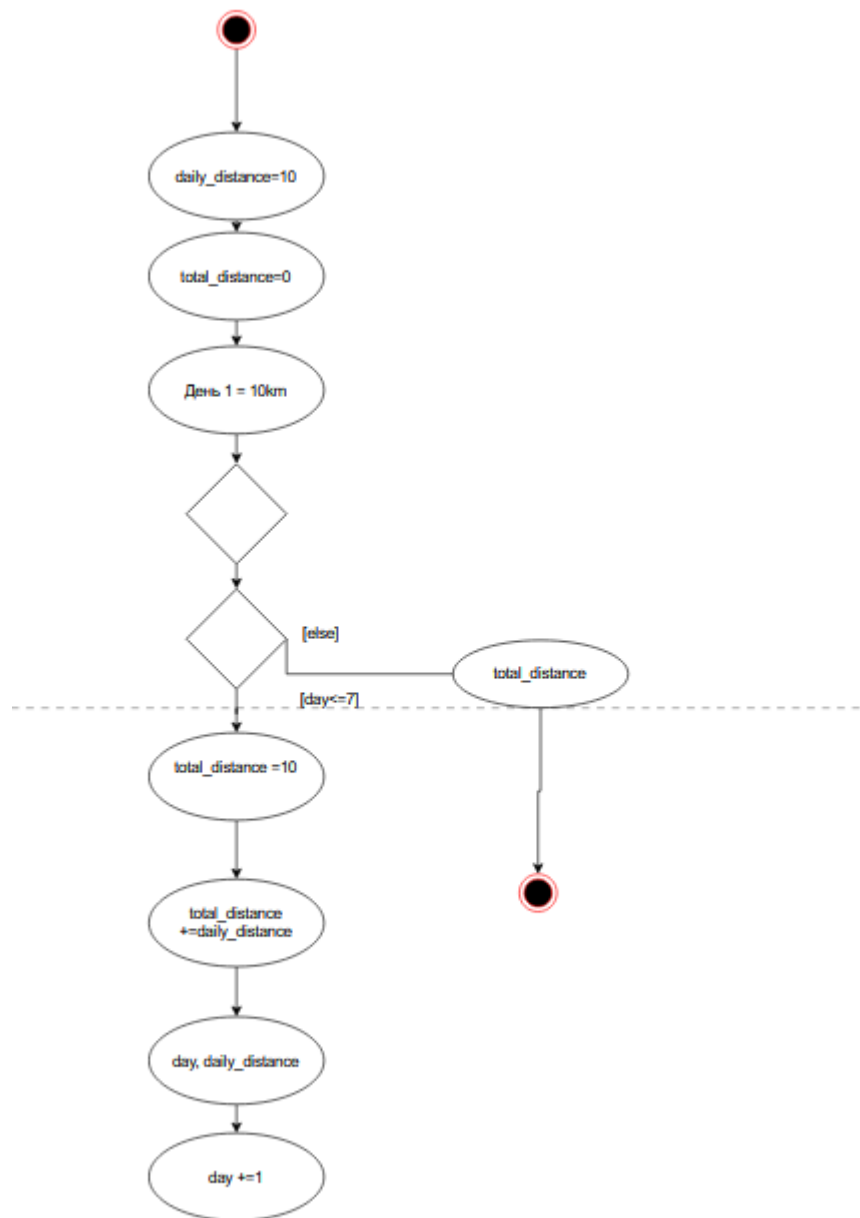


Рисунок 13. UML-диаграмма для 3 задания

Ответы на контрольные вопросы

1. Для чего нужны диаграммы деятельности UML?

Диаграммы деятельности UML используются для моделирования бизнес-процессов и рабочих процессов, визуализации последовательности действий, решений и потоков управления в системе.

2. Что такое состояние действия и состояние деятельности?

- Состояние действия - атомарная операция, которая не может быть прервана (например, простое вычисление)
- Состояние деятельности - составная операция, которая может быть декомпозирована и прервана

3. Какие нотации существуют для обозначения переходов и ветвлений?

- Переходы - простые стрелки между действиями
- Ветвления - ромбы с условиями [условие]
- Начало/Конец - закрашенный круг и круг с границей

4. Какой алгоритм является алгоритмом разветвляющейся структуры?

Алгоритм, в котором вычислительный процесс осуществляется по разным ветвям в зависимости от выполнения условий (if-elif-else).

5. Чем отличается разветвляющийся алгоритм от линейного?

- Линейный - последовательное выполнение операций
- Разветвляющийся - выполнение разных операций в зависимости от условий.

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

Формы оператора if:

if условие:

if условие: ... else:

if условие: ... elif: ... else:

7. Какие операторы сравнения используются в Python?

==, !=, <, >, <=, >=, is, is not, in, not in

8. Что называется простым условием?

Условие с одним логическим выражением:

возраст >= 18

9. Что такое составное условие?

Условие с несколькими выражениями, соединенными логическими операторами:

возраст \geq 18 and возраст \leq 65

10. Какие логические операторы допускаются?

and, or , not

11. Может ли оператор ветвления содержать внутри себя другие ветвления?

Да, операторы if могут содержать другие операторы if внутри (вложенные условия).

12. Какой алгоритм является алгоритмом циклической структуры?

Алгоритм, в котором многократно повторяется выполнение одной и той же последовательности операций.

13. Типы циклов в языке Python:

- while- выполняется пока условие истинно
- for-выполняется для каждого элемента последовательности

14. Назначение и способы применения функции range:

Функция range генерирует последовательности чисел для использования в циклах:

range(stop)

range(start, stop)

range(start, stop, step)

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

range(15, -1, -2) # 15, 13, 11, 9, 7, 5, 3, 1

16. Могут ли быть циклы вложенными?

Да, циклы могут быть вложенными (один цикл внутри другого).

17. Как образуется бесконечный цикл и как выйти из него?

while True: # Бесконечный цикл

if условие:

`break` *# Выход из цикла*

18. Для чего нужен оператор `break`?

Для досрочного прерывания выполнения цикла.

19. Где употребляется оператор `continue`?

Используется в циклах для перехода к следующей итерации, пропуская оставшийся код текущей итерации.

20. Для чего нужны стандартные потоки `stdout` и `stderr`?

- `stdout` - для вывода обычных данных
- `stderr` - для вывода сообщений об ошибках

21. Как организовать вывод в `stderr`?

```
import sys
print("Ошибка!", file=sys.stderr)
```

22. Назначение функции `exit`?

Завершение выполнения программы с кодом возврата:

`exit(0)` Успешное завершение

`exit(1)` Завершение с ошибкой

Вывод: в ходе лабораторной работы были на практике изучены условные операторы и циклы.

