

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

Институт Перспективной инженерии
Департамент цифровых и робототехнических систем и электроники

ОТЧЕТ
ПО ПРАКТИЧЕСКОЙ РАБОТЕ № 3
дисциплины «Программирование на Python»

Выполнил: Мендеш Пашкоал Педру
1 курс, группа ИВТ-б-о-24-1,
09.03.01 «Информатика и Вычислительная
техника», направленность (профиль)
«Программное обеспечение средств
вычислительной техники и
автоматизированных систем», очная форма
обучения

(подпись)

Руководитель практики:
Воронкин Роман А. доцент факультета
цифровых, робототехнических систем и
электроники института перспективной
инженерии.

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2025 г.

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.

Выполненные этапы:

1. Изучили теоретический материал по условным операторам и циклам в Python
2. Создали программный код в среде PyCharm:
 - 4 индивидуальных задания
 - 2 примера из лабораторной работы
3. Создали общедоступный репозиторий на GitHub:
 - Название репозитория : Laboratorio-python-condicionais-ciclos
 - Ссылка: <https://github.com/Pascoalpm/Laboratorio-python-condicionais-ciclos>
4. Организовали структуру репозитория:
 - Папка src/tarefas/ - файлы заданий
 - Папка src/exemplos/ - файлы примеров
5. Добавили все файлы в репозиторий и выполнили коммиты

1. 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.

Рис. 1. Пример 4

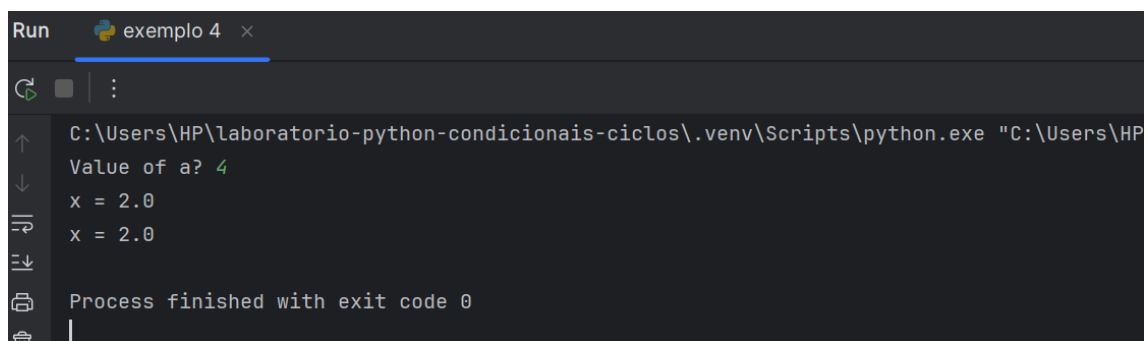
```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

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)}")
```



```
Run  exemplo 4  x
C:\Users\HP\laboratorio-python-condicionais-ciclos\.venv\Scripts\python.exe "C:\Users\HP
Value of a? 4
x = 2.0
x = 2.0
Process finished with exit code 0
```

Рис. 2. Код, выполненный в PyCharm

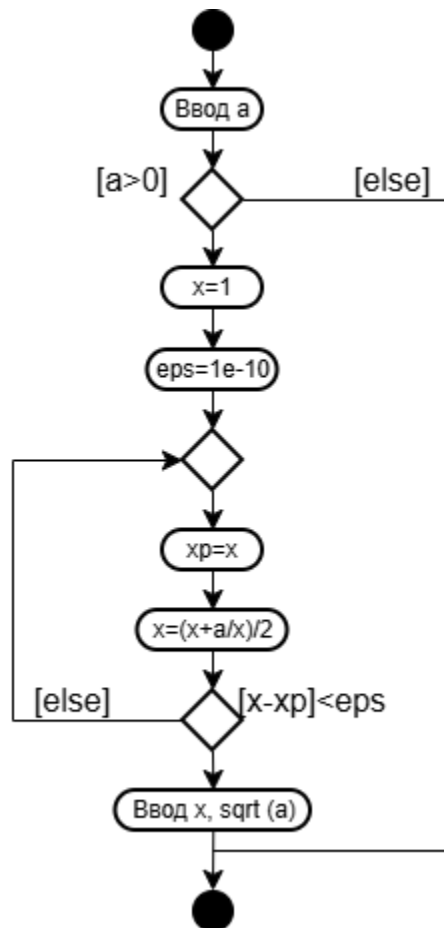


Рис. 3. UML-диаграмма примера 4

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.

Рис. 4. 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}")

```

Run Exemplo 5

```

C:\Users\HP\laboratorio-python-condicionais-ciclos\.venv\Scripts\python.exe "C:\Users\HP\labora
Value of x? 3
Ei(3.0) = 9.93383257061422
Process finished with exit code 0

```

Рис. 5. Выполненный код из примера 5

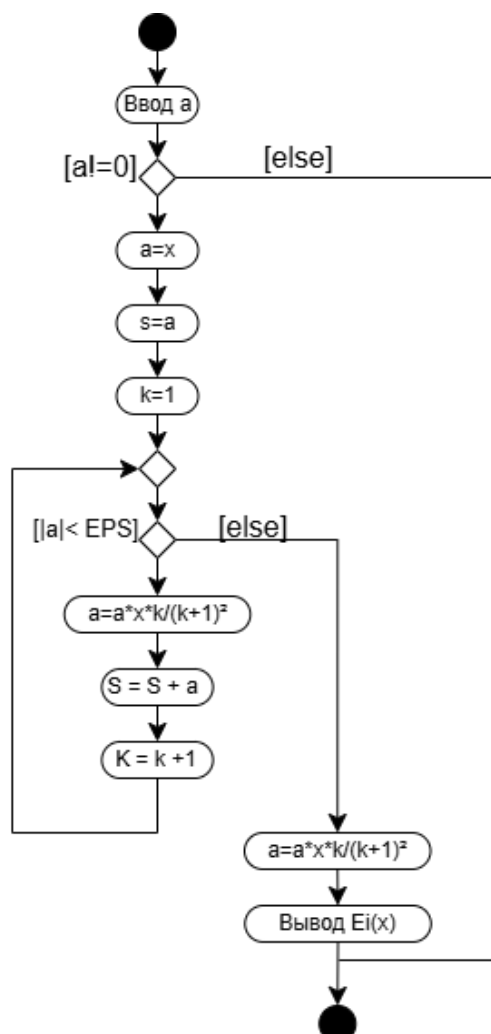


Рис. 6. UML-диаграмма примера 5

Индивидуальные задания

Задание 1

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

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

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

import sys

if __name__ == '__main__':
    # Ввод номера дня недели
    m = int(input("Введите номер дня недели (1-7): "))

    # Проверка и вывод соответствующего дня
    if m == 1:
        print("Понедельник")
    elif m == 2:
        print("Вторник")
    elif m == 3:
        print("Среда")
    elif m == 4:
        print("Четверг")
    elif m == 5:
        print("Пятница")
    elif m == 6:
        print("Суббота")
    elif m == 7:
        print("Воскресенье")
    else:
        print("Ошибка: номер должен быть от 1 до 7", file=sys.stderr)
        exit(1)
```

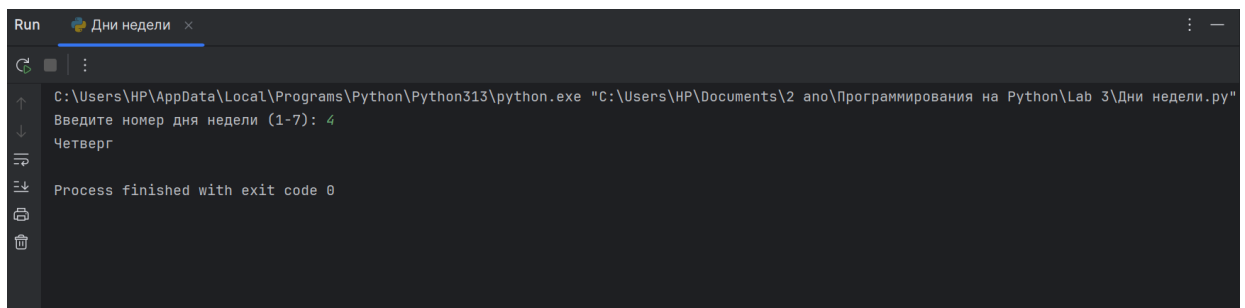


Рис. 7. Код, выполненный из индивидуальной задание 1

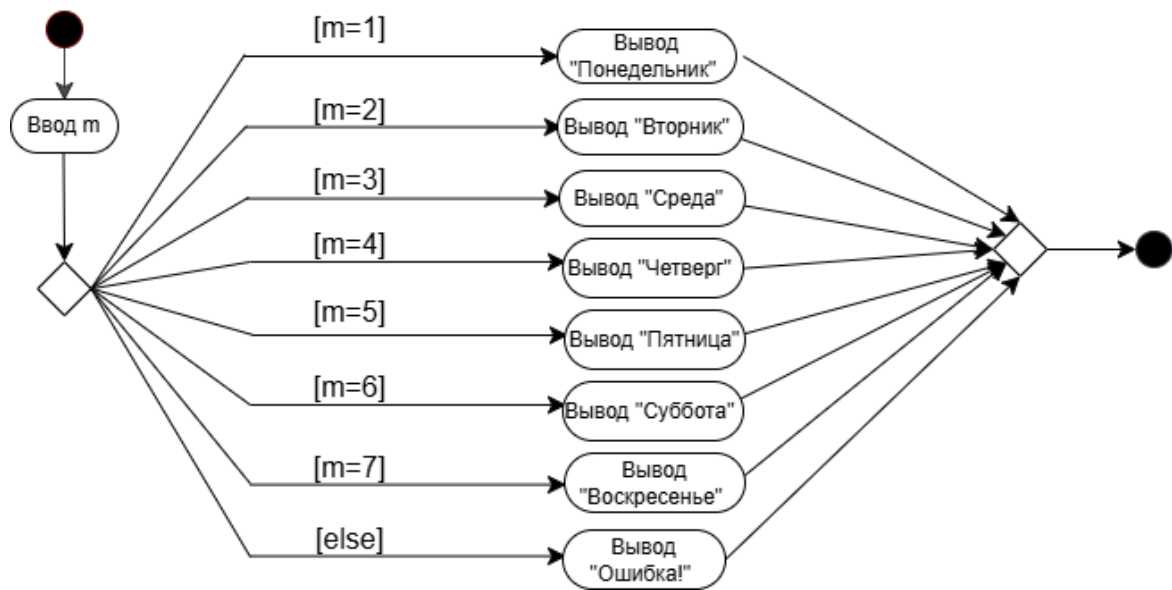


Рис. 8. UML-диаграмма задание 1

Задание 2

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

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

```

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

if __name__ == '__main__':
    # Ввод трех действительных чисел
    a = float(input("Введите число a: "))
    b = float(input("Введите число b: "))
    c = float(input("Введите число c: "))

    print("Числа, модуль которых не меньше 4:")

    # Проверка каждого числа
    if abs(a) >= 4:
        print(f"a = {a}")

    if abs(b) >= 4:
        print(f"b = {b}")

    if abs(c) >= 4:
        print(f"c = {c}")
  
```

```
Run  tarefa2_modulo_numeros x
C:\Users\HP\AppData\Local\Programs\Python\Python313\python.exe "C:\Users\HP\Documents\2 ano\Программирования на Python\Lab 3\tarefa2_modulo
Введите число a: 3.5
Введите число b: -6
Введите число c: 4
Числа, модуль которых не меньше 4:
b = -6.0
c = 4.0
Process finished with exit code 0
```

Рис. 9. Код, выполненный из индивидуальной задание 2

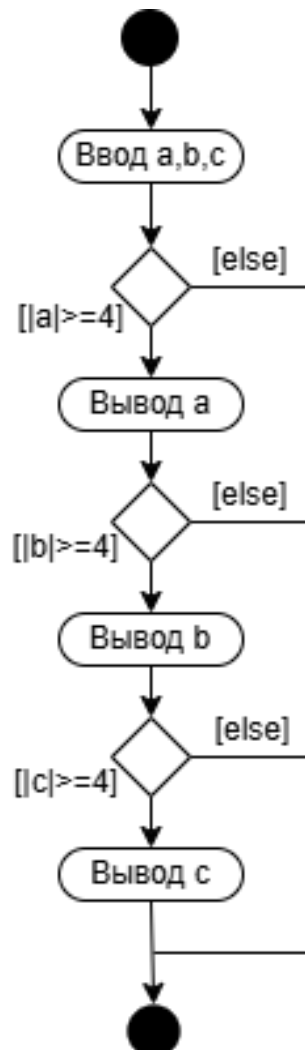


Рис. 10. UML-диаграмма задание 2

Задание 3

Составить UML-диаграмму деятельности и программу с использованием конструкций цикла для решения задачи. Номер варианта необходимо получить у преподавателя.

4. 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?

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

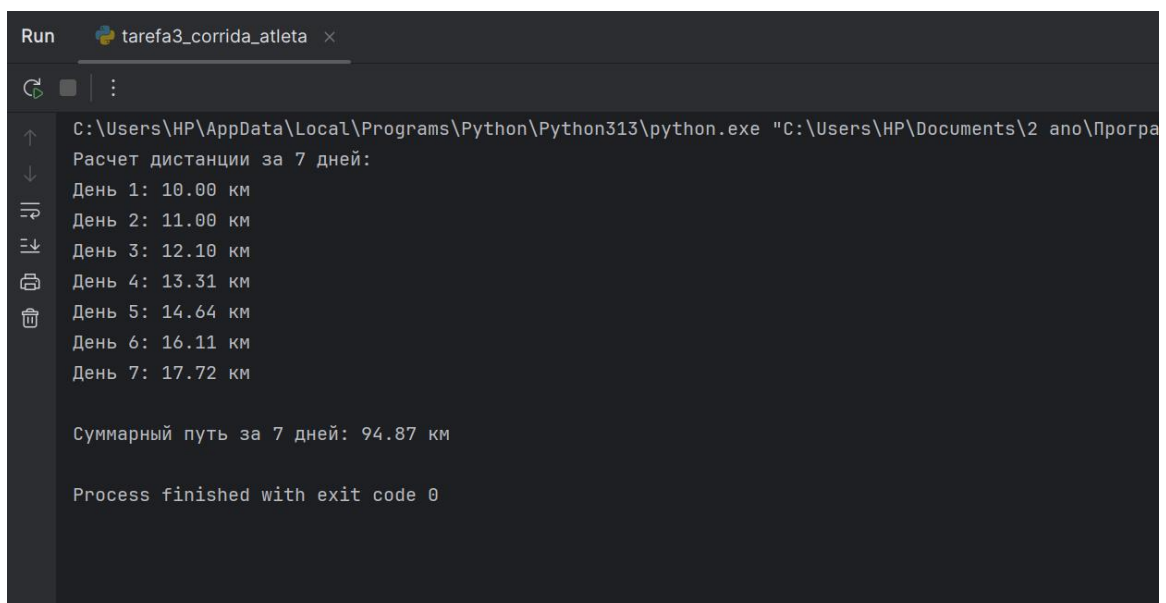
if __name__ == '__main__':
    # Начальная дистанция
    daily_distance = 10.0 # км
    total_distance = 0.0

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

    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("День {}: {:.2f} км".format(day, daily_distance))

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



```
Run  tarefa3_corrida_atleta x
C:\Users\HP\AppData\Local\Programs\Python\Python313\python.exe "C:\Users\HP\Documents\2 ano\Програ
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
```

Рис. 11. Код, выполненный из индивидуальной задание 2

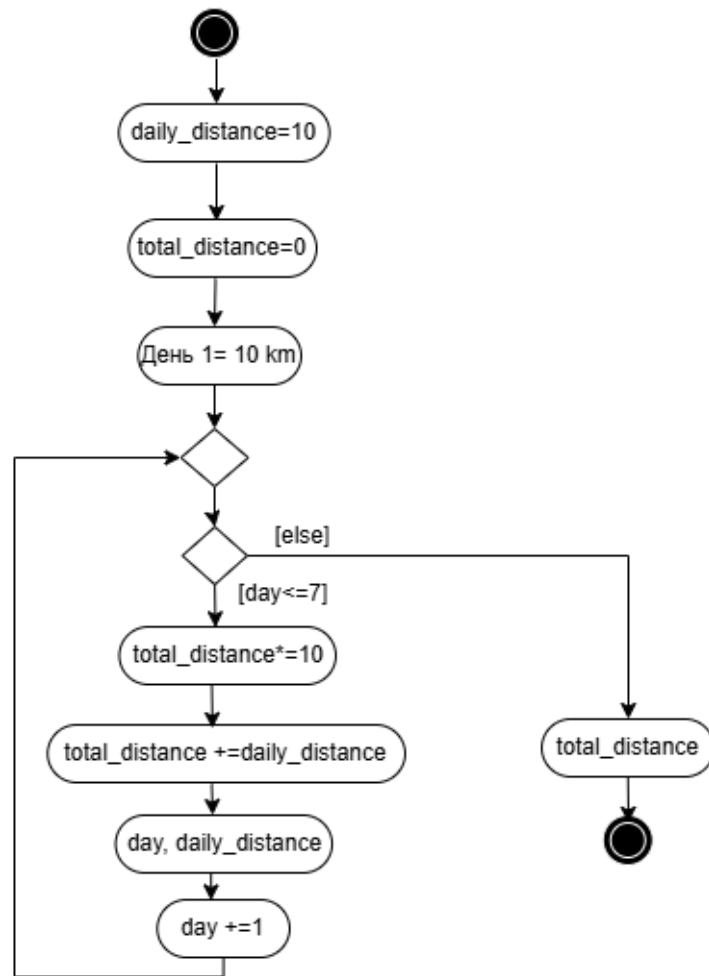


Рис. 12. UML-диаграмма задание 3

Задание повышенной сложности

Составить UML-диаграмму деятельности, программу и произвести вычисления значения специальной функции по ее разложению в ряд с точностью $\epsilon = 10^{-10}$, аргумент функции x вводится с клавиатуры. Номер варианта необходимо получить у преподавателя.

3. Интегральный гиперболический синус:

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

Рис 13. Задание 3 (вариант 3)

```

import math

# Precisão desejada
EPS = 1e-10

def shi_series(x: float, eps: float = EPS) -> float:
    # первый срок a0 = x
    a = x
    s = 0.0
    k = 0
    # Сложить, пока текущий член больше eps по модулю
    while abs(a) > eps:
        s += a
        # рекурсивное отношение:
        # a[k+1] = a_k * x^2 * (k + 1/2) / ((k + 1) * (2k + 3)^2)
        denom = (k + 1) * (2 * k + 3) * (2 * k + 3)
        a = a * (x * x) * (k + 0.5) / denom
        k += 1
    return s

if __name__ == "__main__":
    x = float(input("Value of x? "))

    result = shi_series(x, EPS)
    print(f"Shi({x}) = {result:.12g}")

```

Run tarefa_avancada_shi_function

C:\Users\HP\AppData\Local\Programs\Python\Python313\python.exe "C:\Users\HP\Documents\2 ano\Программирования на Python\Lab 3\tarefa_avancada"

Value of x? 2

Shi(2.0) = 2.50156743333

Process finished with exit code 0

Рис. 14. Код, выполненный из повышенной сложности Задание

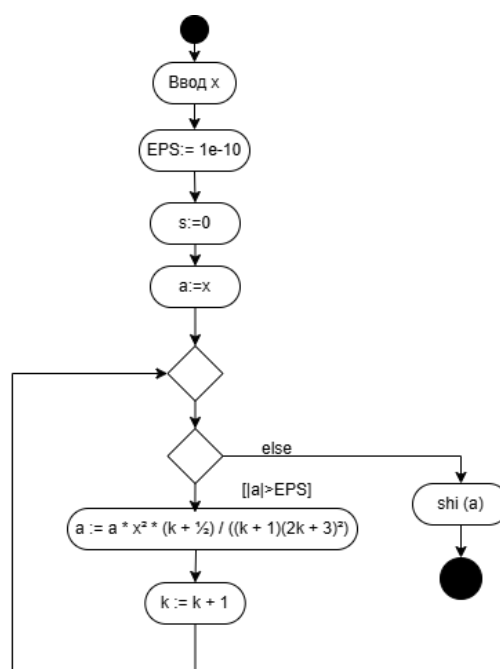


Рис. 15. UML-диаграмма из повышенной сложности Задание

Добавили все файлы в репозиторий и выполнили коммиты

```
C:\Users\HP\laboratorio-python-condicionais-ciclos>dir src\tarefas
0 volume na unidade C é Windows
0 Número de Série do Volume é FEAB-2D4F

Pasta de C:\Users\HP\laboratorio-python-condicionais-ciclos\src\tarefas

15.10.2025  20:18    <DIR>          .
15.10.2025  20:10    <DIR>          ..
15.10.2025  14:33             788 tarefa1_dias_semana.py
15.10.2025  14:43             584 tarefa2_modulo_numeros.py
15.10.2025  14:55             712 tarefa3_corrida_atleta.py
15.10.2025  14:57           1 930 tarefa_avancada_shi_function
.py
                                4 arquivo(s)          4 014 bytes
                                2 pasta(s)      9 369 489 408 bytes disponíveis
```

Рис. 16. Добавление задания в репозиторий

```
C:\Users\HP\laboratorio-python-condicionais-ciclos>git add src/exemplos/

C:\Users\HP\laboratorio-python-condicionais-ciclos>git commit -m "feat: adiciona exemplos 4 e 5 do material"
[main f920229] feat: adiciona exemplos 4 e 5 do material
 2 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 Exemplo 4.py
 create mode 100644 Exemplo 5.py

C:\Users\HP\laboratorio-python-condicionais-ciclos>git push origin main
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 401 bytes | 401.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/Pascoalpm/laboratorio-python-condicionais-ciclos.git
 16af3f1..f920229  main -> main
```

Рис. 17. Добавление примеры в репозиторий

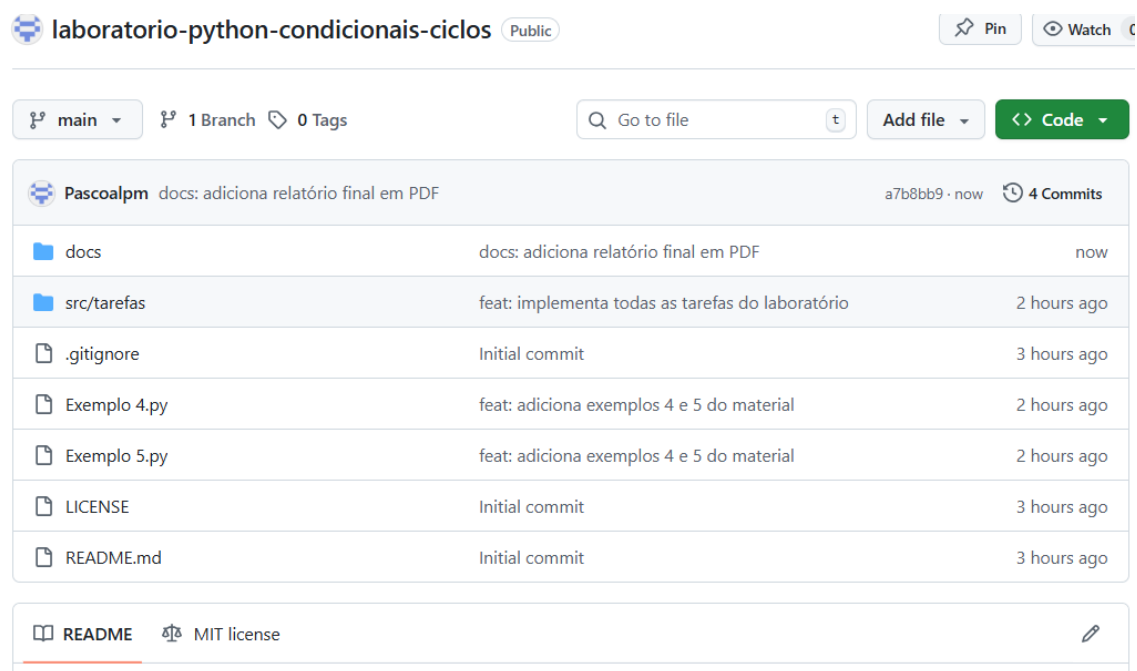


Рис. 18. Мой репозиторий со всеми добавленными файлами

Вывод

В ходе лабораторной работы были успешно выполнены все поставленные задачи.

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 >= 18 and возраст <= 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
```