1инистерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего образования

«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт перспективной инженерии

Департамент цифровых, робототехнических систем и электроники

**ОТЧЕТ**

**ПО ПРАКТИЧЕСКОЙ РАБОТЕ №3**

**дисциплины «Программирование на Python»**

Вариант №11

|  |  |  |  |
| --- | --- | --- | --- |
|  | | Выполнила:  Ковжого Елизавета Андреевна  2 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 Р. А., доцент департамента 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 u continue, позволяющих реализовывать разветвляющиеся алгоритмы и алгоритмы циклической структуры.

Порядок выполнения работы:

Адес репозитория: https://github.com/LissKovzogo/Python\_LAB\_33.git

1. Создали, настроили и клонировали репозиторий Python\_LAB\_3.
2. Создали проект PyCharm в папке репозитория.
3. Проработали все примеры лабораторной работы и создали для каждого отдельный модуль Python.

Листинг кода примера №1:

#!/usr/bin/env python3

# -\*- coding: utf-8 -\*-

import math

if \_\_name\_\_ == "\_\_main\_\_":

x = float(input("Value of x? "))

if x <= 0 :

y = 2 \* x \* x + math.cos(x)

elif x < 5:

y = x+1

else:

y = math.sin(x) - x\*x

print(f"y = {y}")

Листинг кода примера №2:

#!/usr/bin/env python3

# -\*- coding: utf-8 -\*-

import sys

if \_\_name\_\_ == "\_\_main\_\_":

n = float(input("Введите номер месяца: "))

if n == 1 or n ==2 or n == 12: print("Зима")

elif n == 3 or n ==4 or n == 5: print("Весна")

elif n == 6 or n ==7 or n == 8: print("Лето")

elif n == 9 or n ==10 or n == 11: print("Осень")

else:

print("Ошибка!", file= sys.stderr)

exit(1)

Листинг кода примера № 3:

#!/usr/bin/env python3

# -\*- coding: utf-8 -\*-

import math

if \_\_name\_\_ == "\_\_main\_\_":

n = int(input("Value of n? "))

x = float(input("Value of x? "))

s = 0.0

for k in range(1, n+1):

a = math.log(k\*x)/(k\*k)

s +=a

print(f"s = {s}")

Листинг кода примера № 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}\nx = {math.sqrt(a)}")

Листинг кода примера № 5:

#!/usr/bin/env python3

# -\*- coding: utf-8 -\*-

import math

import sys

eps = 1e-10

EULER = 0.5772156649015328606

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

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}")

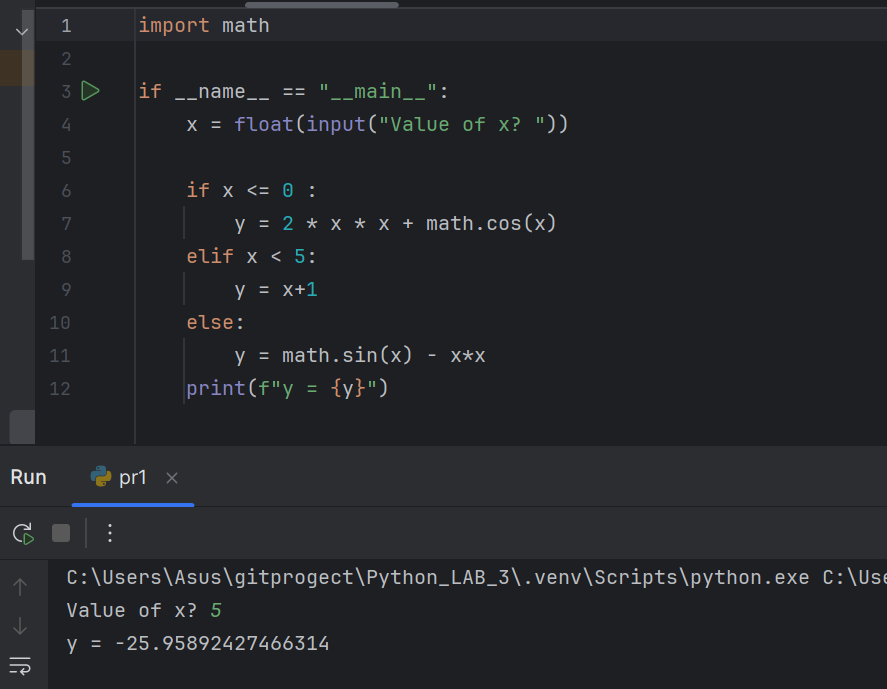


Рис. 1 — Работа с примером № 1

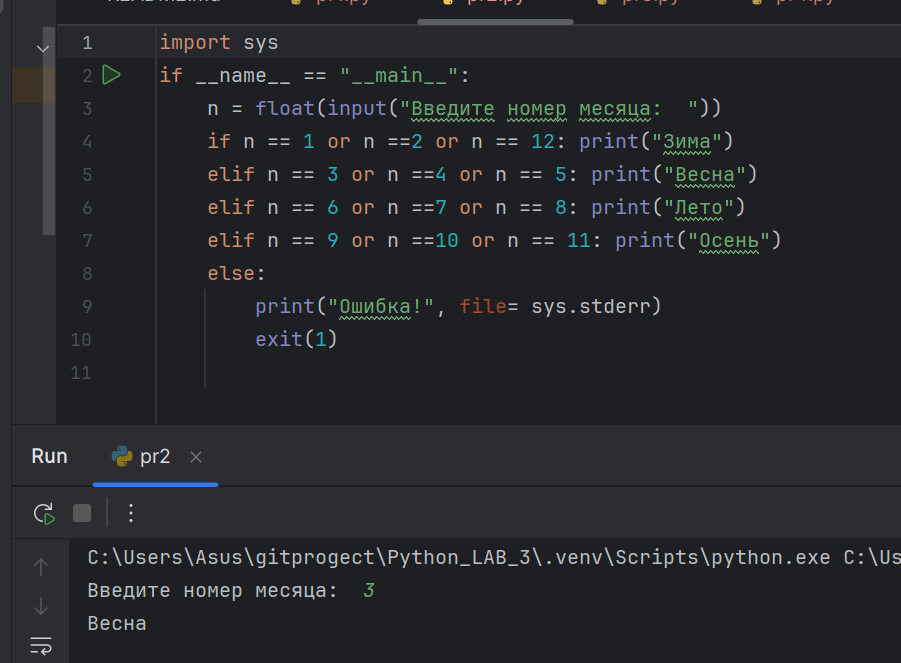


Рис. 2 — Работа с примером № 2

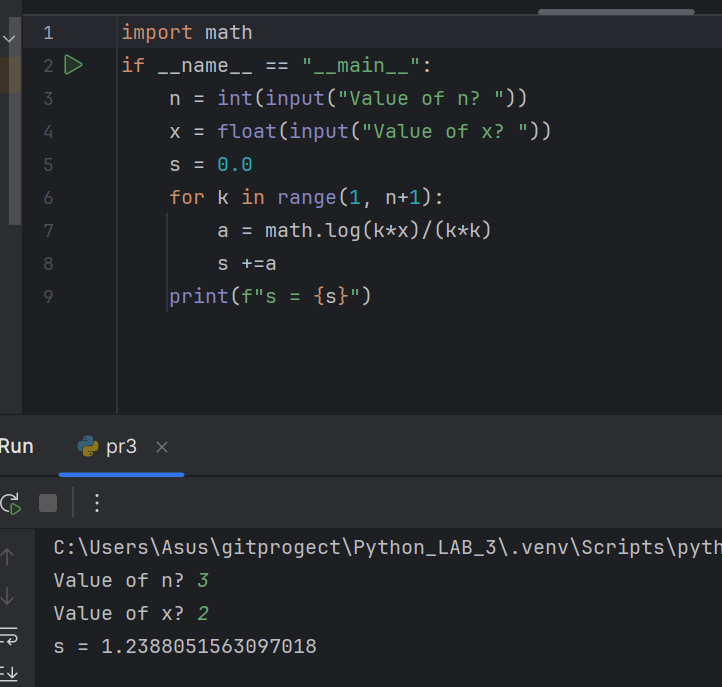


Рис. 3 — Работа с примером № 3

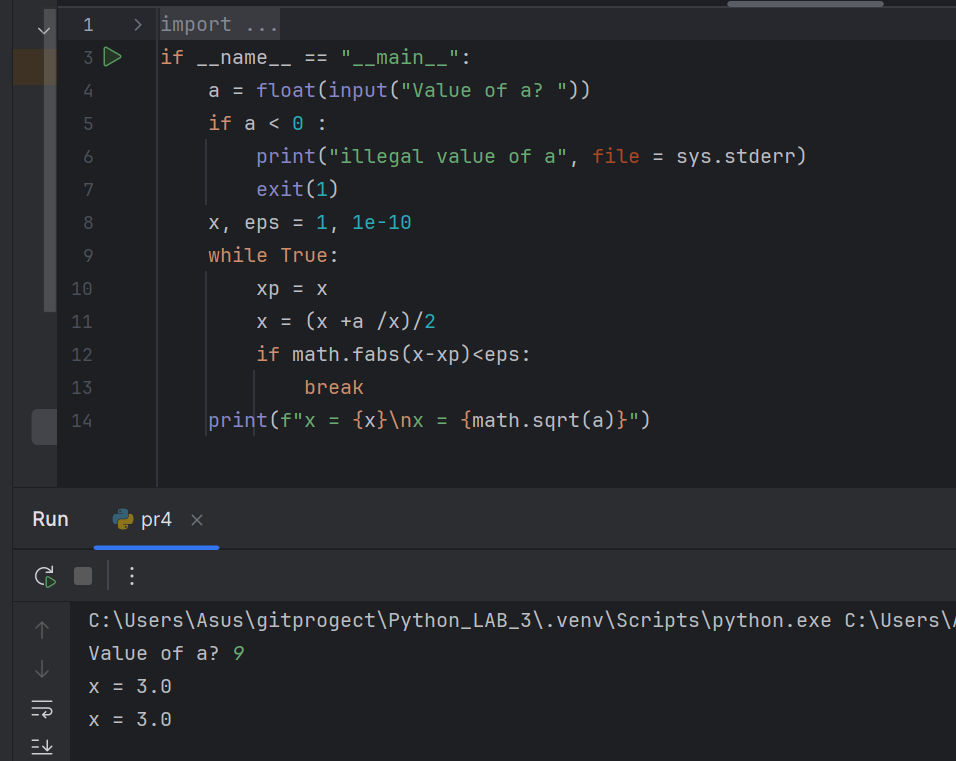


Рис. 4 — Работа с примером № 4

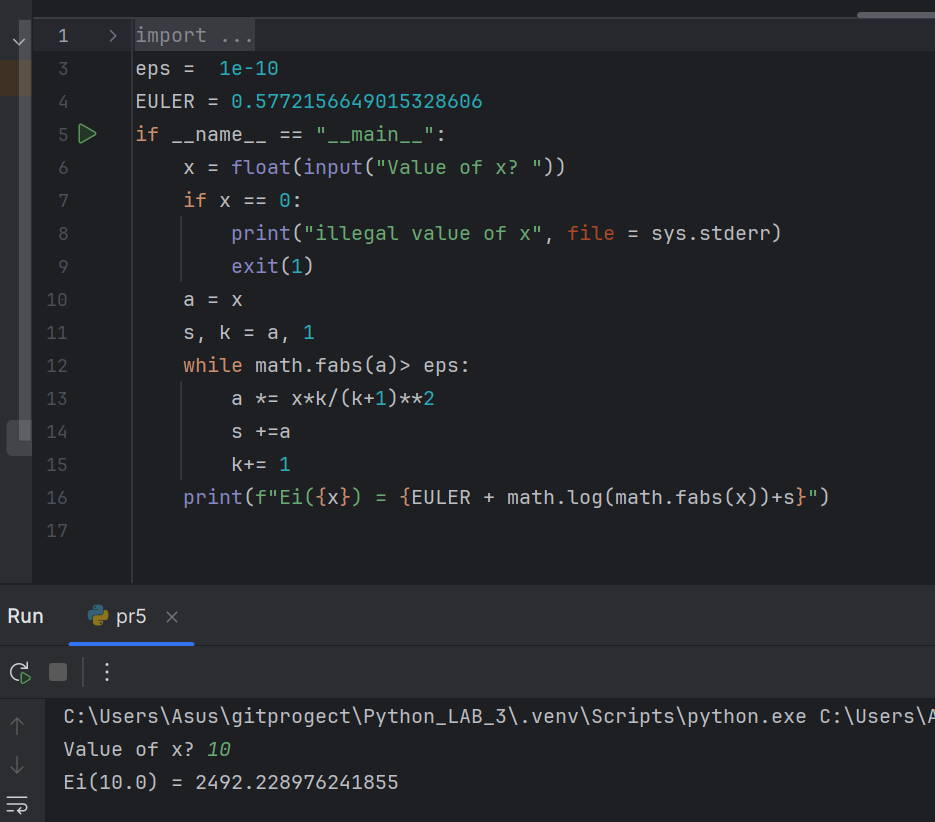


Рис. 5 — Работа с примером № 5

1. Построили UML-диаграммы деятельности для примеров №4 и №5.

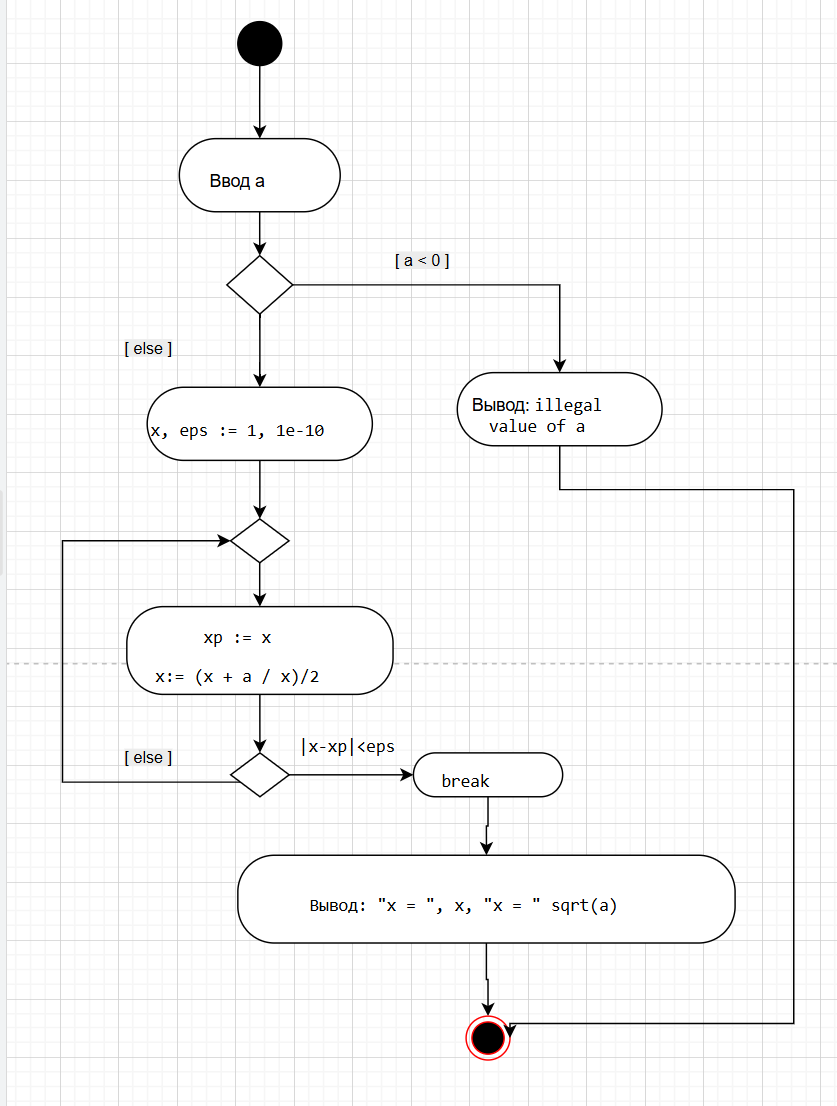


Рис. 6 — UML-диаграммы деятельности для примеров №4

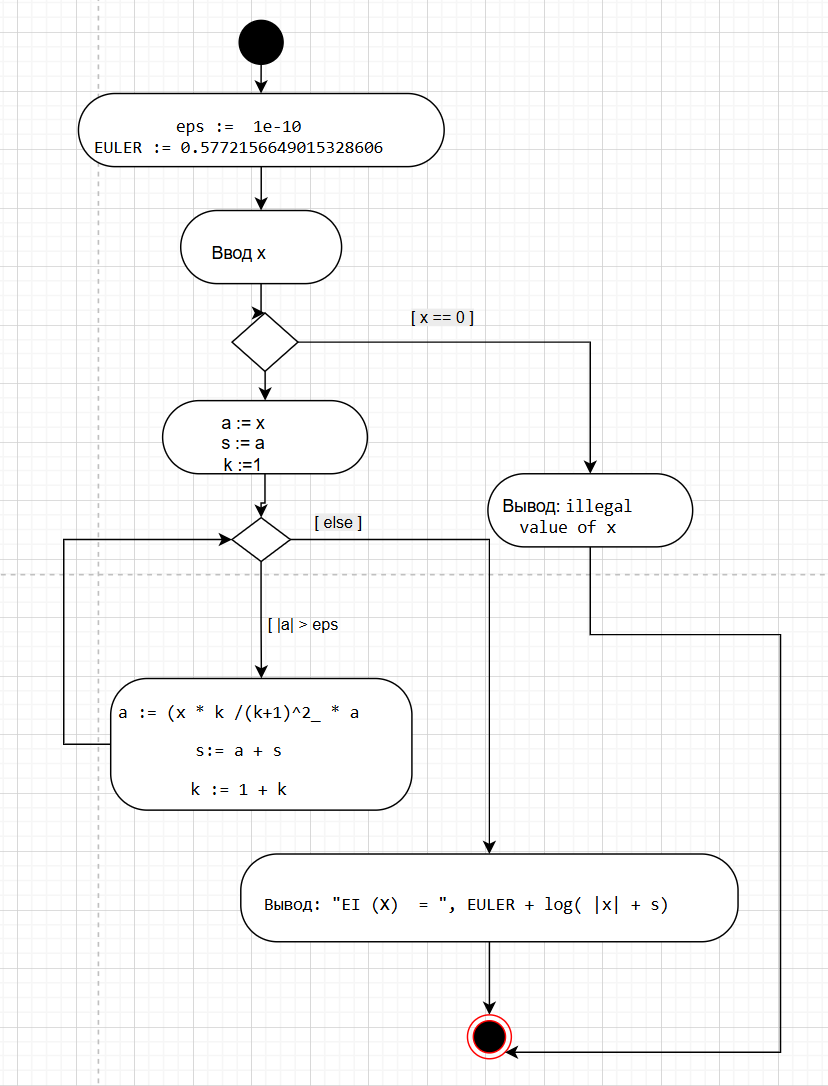


Рис. 7 — UML-диаграммы деятельности для примеров №5

1. Выполнили индивидуальные задания согласно варианту и построили UML-диаграммы деятельности.

Индивидуальное задание №1: Студенты убирают урожай помидоров. При сборе до 50 кг в день работа оплачивается из расчёта 30 коп. за 1 кг; при сборе от 50 до 75 кг в день 50 коп. за 1 кг; при сборе от 75 до 90 кг в день 65 коп. за 1 кг; при сборе свыше 90 кг в день 70 коп. за 1 кг плюс 20 руб. премия. Студент собрал Х кг за день. Определить его заработок.

Индивидуальное задание №2: Составить программу, выясняющую делится ли натуральное число х нацело на натуральное число у.

Индивидуальное задание №3: Дано натуральное число п. Получить все его натуральные делители.

Листинг кода индивидуального задания №1:

#!/usr/bin/env python3

# -\*- coding: utf-8 -\*-

if \_\_name\_\_ == "\_\_main\_\_":

x = int(input("Value of x? "))

zp = 0

if x <= 50:

zp = 30 \* x

elif 50 < x <= 75:

zp = 50 \* x

elif 75 < x <= 90:

zp = 65 \* x

else: zp = 70 \* x + 2000

print(f"Заработок в копейках = {zp}, "

f"Зараб оток в рублях = {zp/100}")

Листинг кода индивидуального задания №2:

#!/usr/bin/env python3

# -\*- coding: utf-8 -\*-

if \_\_name\_\_ == "\_\_main\_\_":

x = int(input("Value of x? "))

y = int(input("Value of y? "))

if x % y == 0:

print("x делится на у без остатка")

else:

print("x не делится на у без остатка")

Листинг кода индивидуального задания №3:

#!/usr/bin/env python3

# -\*- coding: utf-8 -\*-

import math

if \_\_name\_\_ == "\_\_main\_\_":

n = int(input("Value of n? "))

divisors = []

for i in range(1, int(math.sqrt(n) + 1)):

if n % i == 0:

divisors.append(n//i)

divisors.append(i)

print(f"Делители числа {n}: {divisors}")

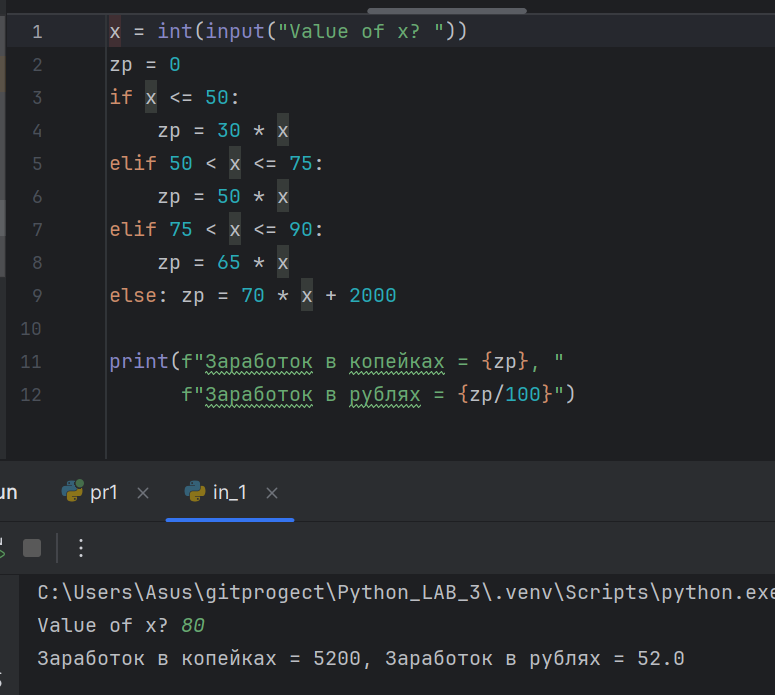


Рис. 8 — Индивидуальное задание №1

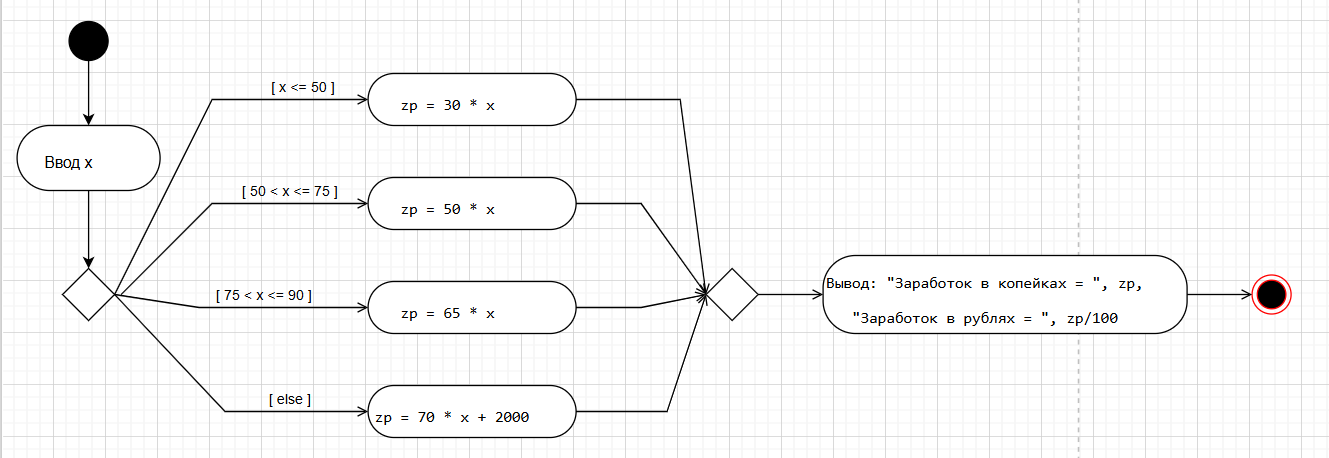


Рис. 9 — UML-диаграммы деятельности для индивидуальное задания №1

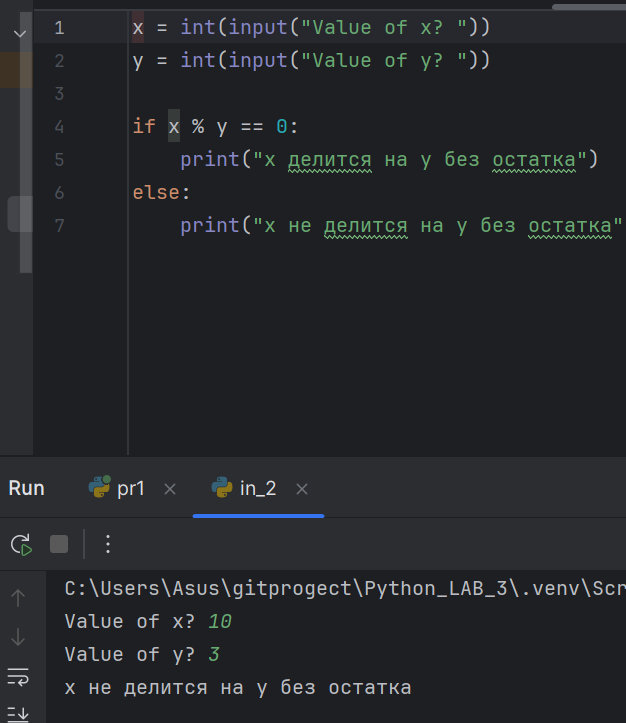


Рис. 10 — Индивидуальное задание №2

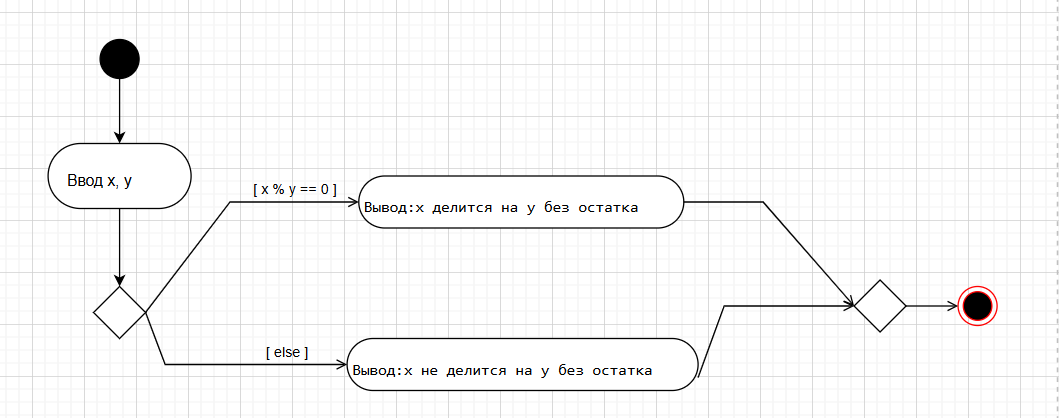


Рис. 11 — UML-диаграммы деятельности для индивидуальное задания №2

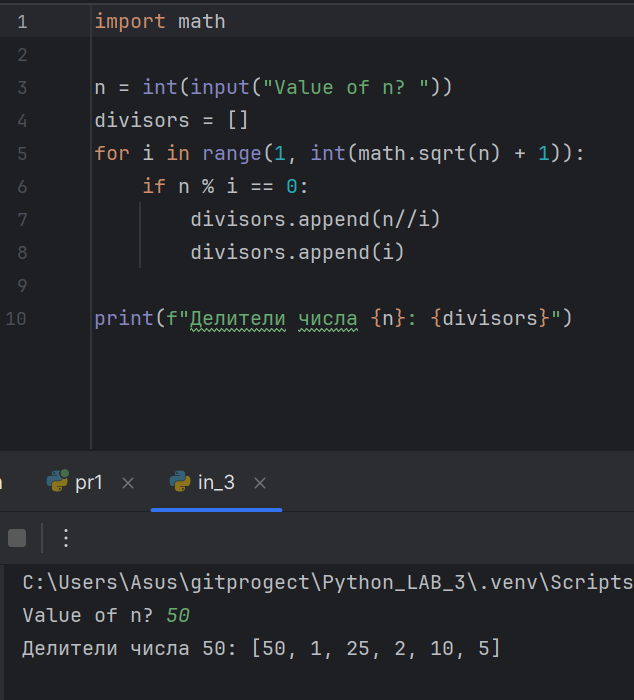


Рис. 12 — Индивидуальное задание №3

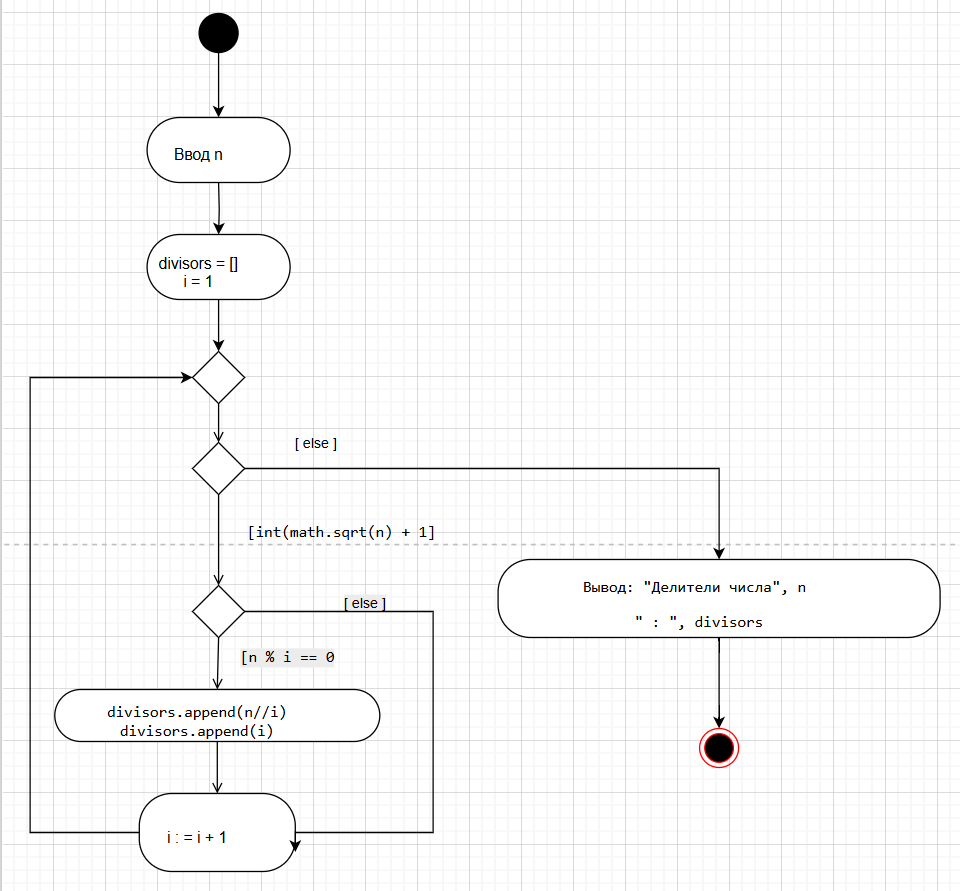


Рис. 13 — UML-диаграммы деятельности для индивидуальное задания №3

1. Выполнили индивидуальное задание повышенной сложности и составили UML-диаграмму деятельности.

Индивидуальное задание повышенной сложности: Интегральный косинус

Решение:

Выразим текущий член ряда:

Следующий член ряда:

Листинг кода индивидуального задания повышенной сложности:

import math

#!/usr/bin/env python3

# -\*- coding: utf-8 -\*-

if \_\_name\_\_ == "\_\_main\_\_":

x = float(input("Введите x: "))

eps = 1e-10

gamma = 0.57721566490153286060

result\_f = gamma + math.log(x)

n = 1

a = -(x\*\*2)/4

result = a

while abs(a) > eps:

a \*= ((-x\*\*2)\*(2\*n))/((2\*n+2)\*(2\*n+2)\*(2\*n+1))

result += a

n += 1

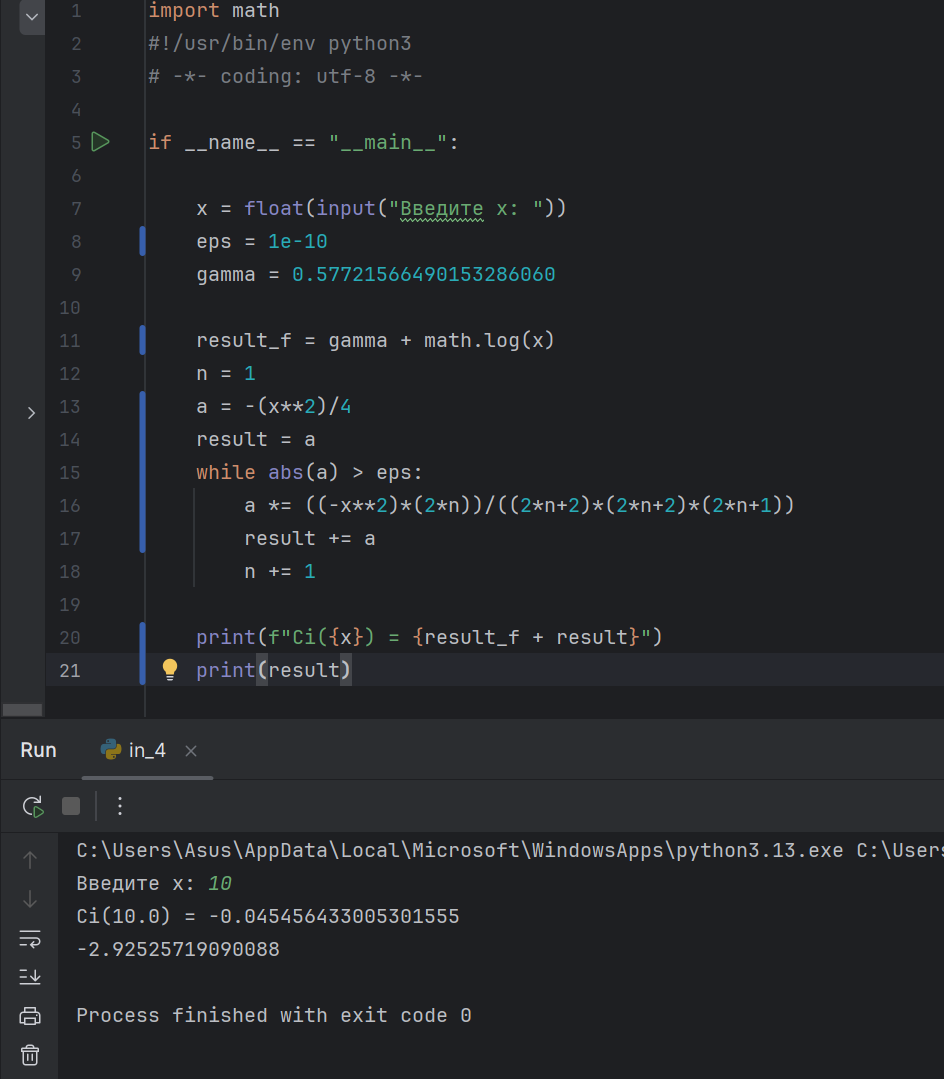
print(f"Ci({x}) = {result\_f + result}")

Рис. 14 — Индивидуальное задание №3

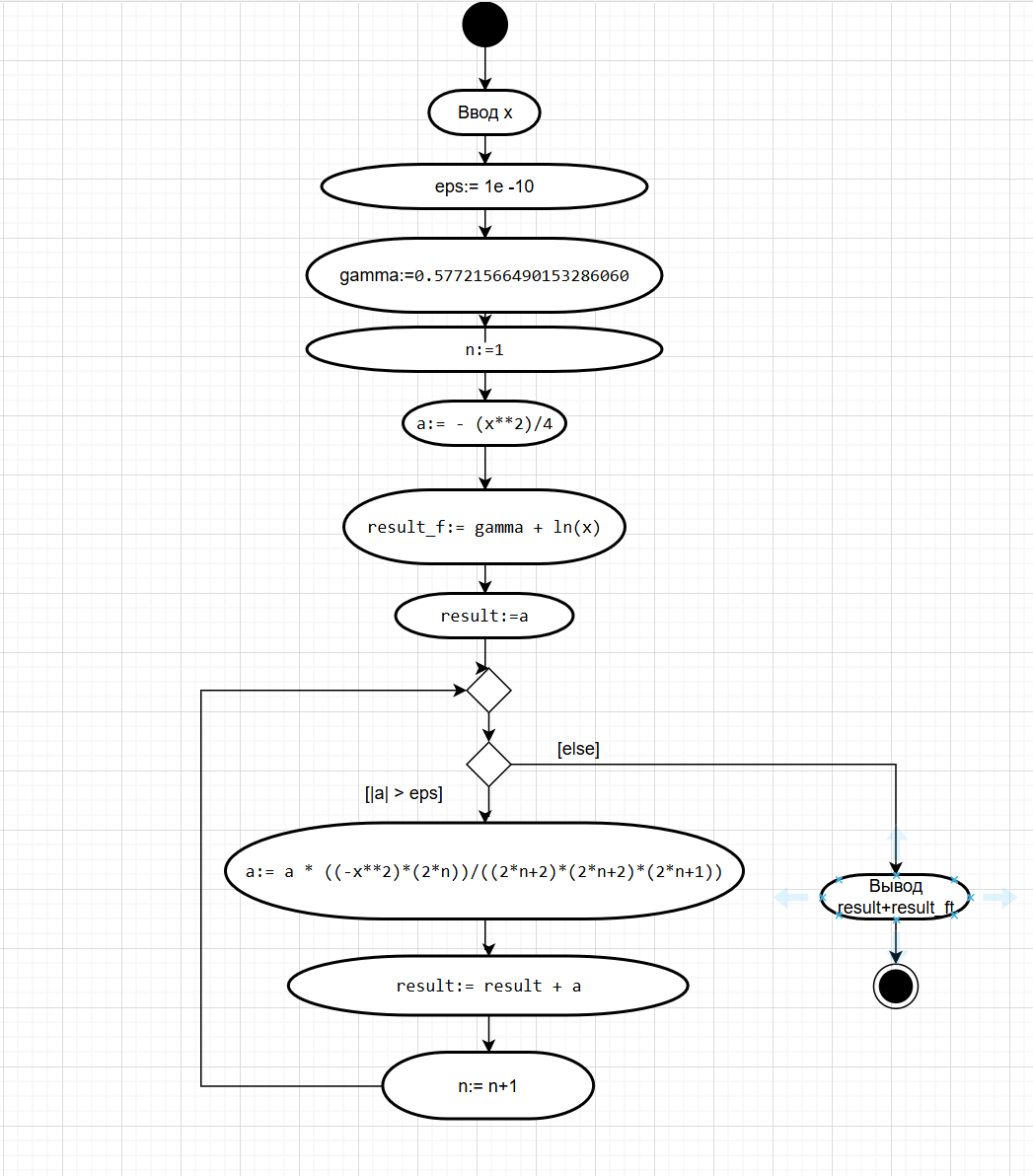


Рис. 15 — UML-диаграммы деятельности для индивидуальное задания №3

1. Зафиксировали все изменения в репозиторий.

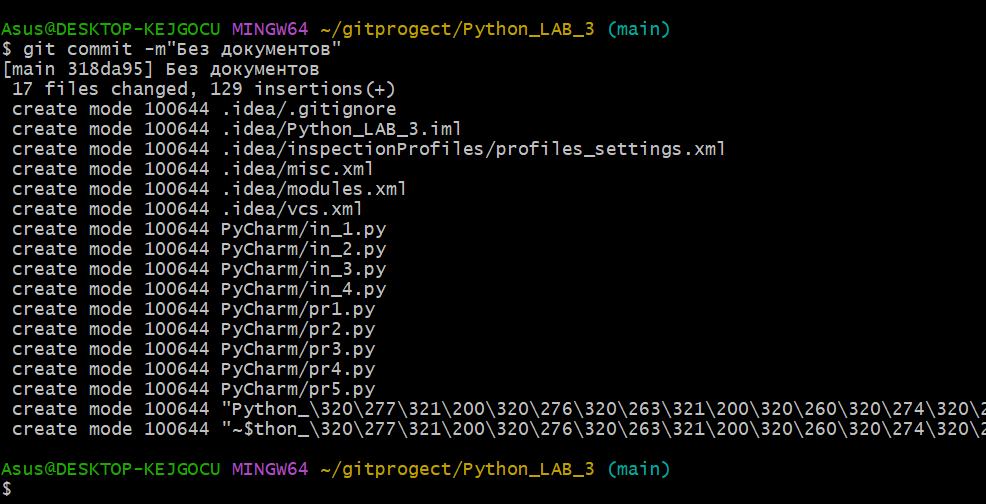


Рис. 16 — Фиксирование изменений

Контрольные вопросы:

1. Для чего нужны диаграммы деятельности UML?

Для визуализации, специфицирования, конструирования и документации артефактов программных систем.

2. Что такое состояние действия и состояние деятельности?

Состояние действия - атомарная операция, состояние системы, представляющее собой выполнение некоторого действия.

Состояние деятельности - составная операция, содержащая поддеятельности, и которую можно прервать.

3. Какие нотации существуют для обозначения переходов и ветвлений в диаграммах деятельности?

Для переходов используются стрелки, показывающие путь из одного состояния в другое.

Ромб является точкой ветвления, из которой алгоритм разветвляется.

4. Какой алгоритм является алгоритмом разветвляющейся структуры?

Это алгоритм, в котором вычислительный процесс осуществляется либо по одной, либо по другой ветви, в зависимости от условия.

5. Чем отличается разветвляющийся алгоритм от линейного?  
Наличием условий и альтернативных ветвей выполнения.

6. Что такое условный оператор? Какие существуют его формы?  
Оператор ветвления: if, if-else, if-elif-else.

7. Какие операторы сравнения используются в Python?  
==, !=, <, >, <=, >=.

8. Что называется простым условием? Приведите примеры.  
Условие с одной логической операцией: x > 5; name == "John".

9. Что такое составное условие? Приведите примеры.  
Условие с логическими операторами: x > 5 and y < 10, age >= 18 or parent\_consent.

10. Какие логические операторы допускаются при составлении сложных условий?

Логические операторы: and, or, not.

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

Да, это называется вложенные условия.

12. Какой алгоритм является алгоритмом циклической структуры?

Алгоритм с многократно повторяющимися участками программы.

13. Типы циклов в языке Python.

for, while, do while.

14. Назовите назначение и способы применения функции range.

Генерация последовательностей чисел: range(start, stop, step).

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

for i in range(15, -1, -2): …

16. Могут ли быть циклы вложенными?

Да, цикл внутри другого цикла.

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

while True: ….

break …

18. Для чего нужен оператор break?

Для досрочного выхода из цикла.

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

В циклах, для перехода к следующей итерации.

20. Для чего нужны стандартные потоки stdout и stderr?

stdout - стандартный вывод даных и информационных сообщений, stderr – вывод сообщений об ошибках.

21. Как в Python организовать вывод в стандартный поток stderr?

import sys;

….

print(“Error!”, file=sys.stderr)

22. Каково назначение функции exit?

Завершение программы с кодом возврата: exit(1)

Вывод: приобрели навыки программирования разветвляющихся алгоритмов и алгоритмов циклической структуры. Освоили операторы языка Python версии 3.x if, while, for, break u continue, позволяющих реализовывать разветвляющиеся алгоритмы и алгоритмы циклической структуры.