**ВВЕДЕНИЕ**

**Python**— высокоуровневый язык программирования общего назначения, ориентированный на повышение производительности разработчика и читаемости кода. Синтаксис ядра Python минималистичен. В то же время стандартная библиотека включает большой объём полезных функций.

Python поддерживает структурное, объектно-ориентированное, функциональное, императивное и аспектно-ориентированное программирование. Основные архитектурные черты — динамическая типизация, автоматическое управление памятью, полная интроспекция, механизм обработки исключений, поддержка многопоточных вычислений, высокоуровневые структуры данных. Поддерживается разбиение программ на модули, которые, в свою очередь, могут объединяться в пакеты.

Эталонной реализацией Python является интерпретатор CPython, поддерживающий большинство активно используемых платформ. Он распространяется под свободной лицензией Python Software Foundation License, позволяющей использовать его без ограничений в любых приложениях, включая проприетарные. Есть реализация интерпретатора для JVM с возможностью компиляции, CLR, LLVM, другие независимые реализации. Проект PyPy использует JIT-компиляцию, которая значительно увеличивает скорость выполнения Python-программ.

Python — активно развивающийся язык программирования, новые версии с добавлением/изменением языковых свойств выходят примерно раз в два с половиной года. Язык не подвергался официальной стандартизации, роль стандарта де-факто выполняет CPython, разрабатываемый под контролем автора языка.

Название языка произошло вовсе не от вида пресмыкающихся. Автор назвал язык в честь популярного британского комедийного телешоу 1970-х «Летающий цирк Монти Пайтона». Впрочем, всё равно название языка чаще связывают именно со змеёй, нежели с передачей — пиктограммы файлов в KDE или в Microsoft Windows и даже эмблема на сайте python.org (до выхода версии 2.5) изображают змеиные головы. Важная цель разработчиков Python — создавать его забавным для использования. Это отражено в его названии, которое пришло из Монти Пайтона. Также это отражено в иногда игривом подходе к обучающим программам и справочным материалам, таким как примеры использования, которые используют понятия ветчины (spam) и яиц вместо стандартных foo и bar.

**Задание 1. Основные правила работы в среде PyCharm.**

**Цель работы**: Получение базовых навыков работы со средой PyCharm; знакомство с простейшей программой на языке Python и изучение ее содержания.

**Задание**

1. Запустите среду PyCharm.

2. Создайте новый проект в вашей папке.

3. Создайте новый файл Python в проекте.

4. Скопируйте код примера (расположен ниже) в

созданный файл и запустите программу.

5. Запустите некоторые команды из примера в

терминале Python (построчный режим ввода) и

проследите за результатом.

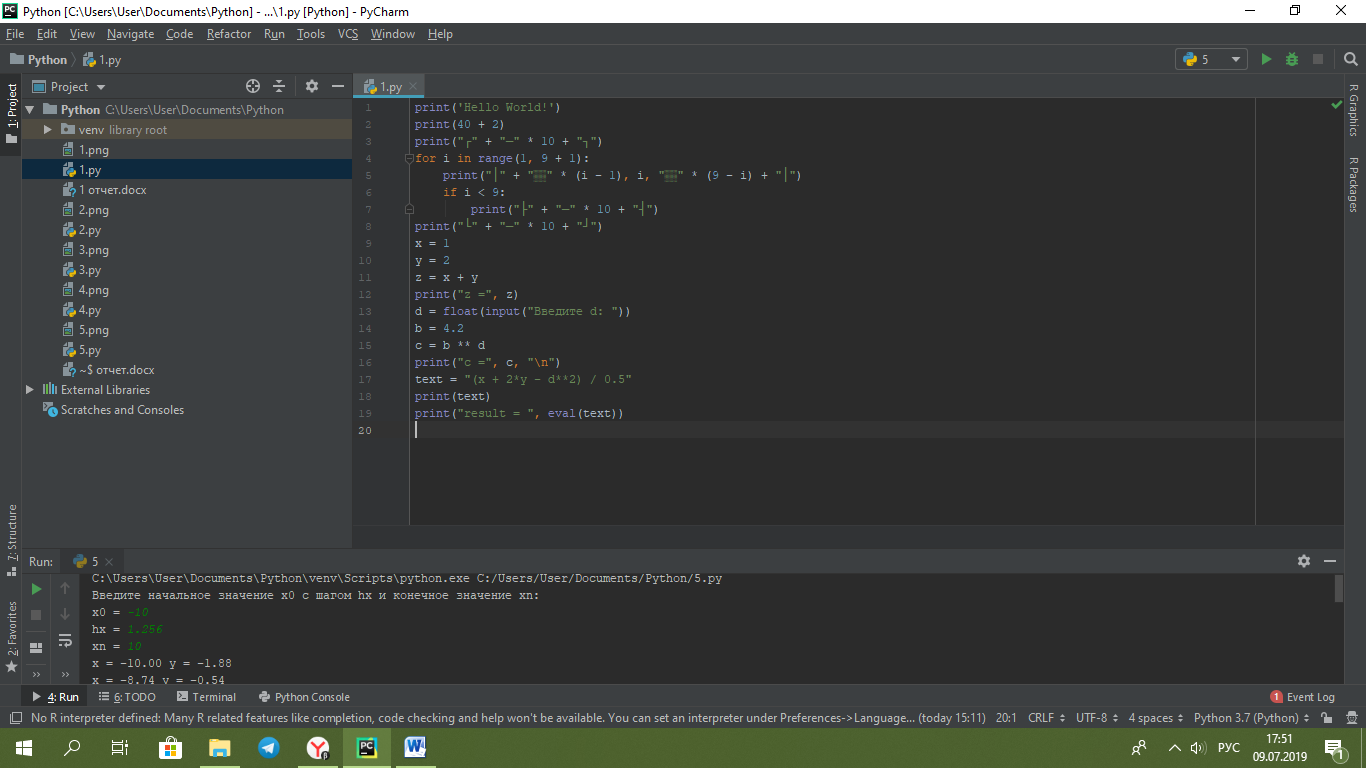
6. Попробуйте изменить код примера и понять

предназначение каждой из команд программы.

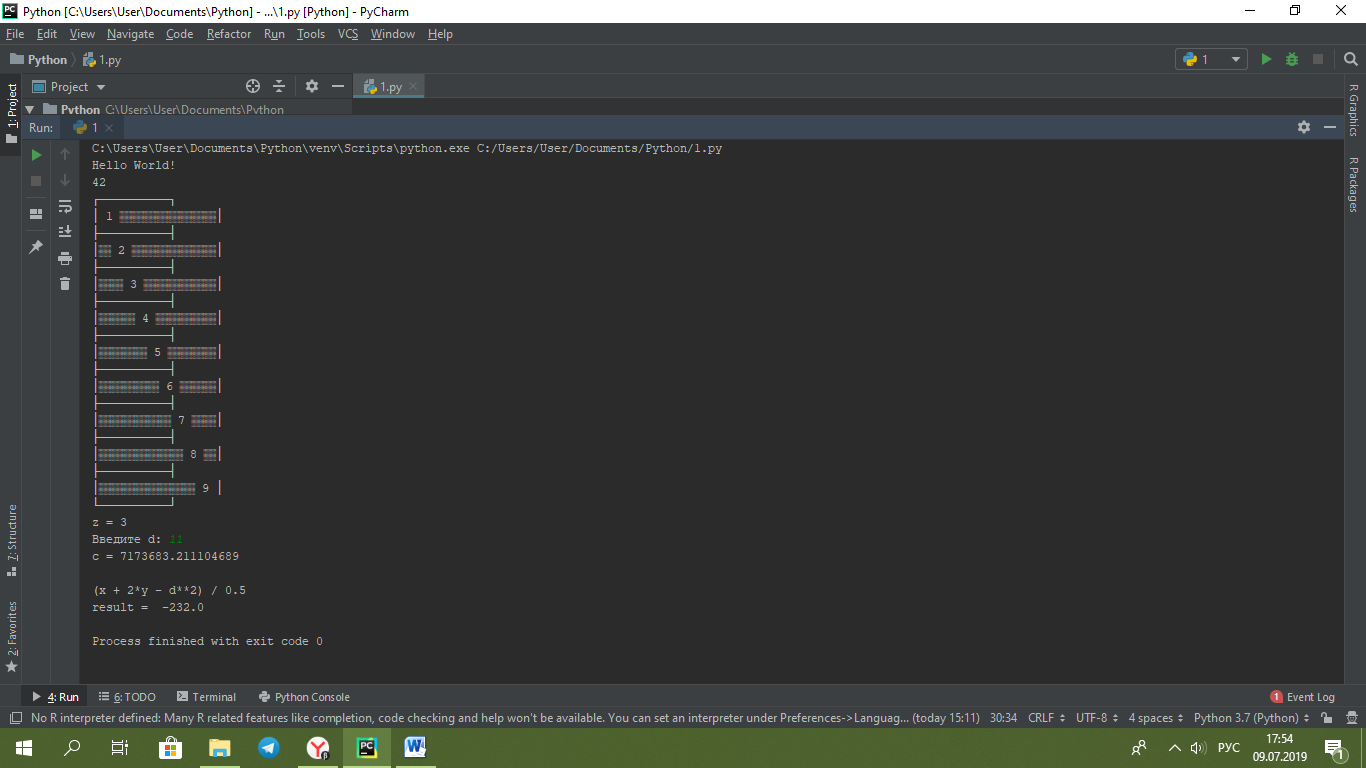
7. Переименуйте одну из переменных в «renamedVar»,

используя встроенную функцию среды.

Код программы:



Результаты выполнения:



**Задание 2. Программы с линейной структурой.**

**Цель работы:**

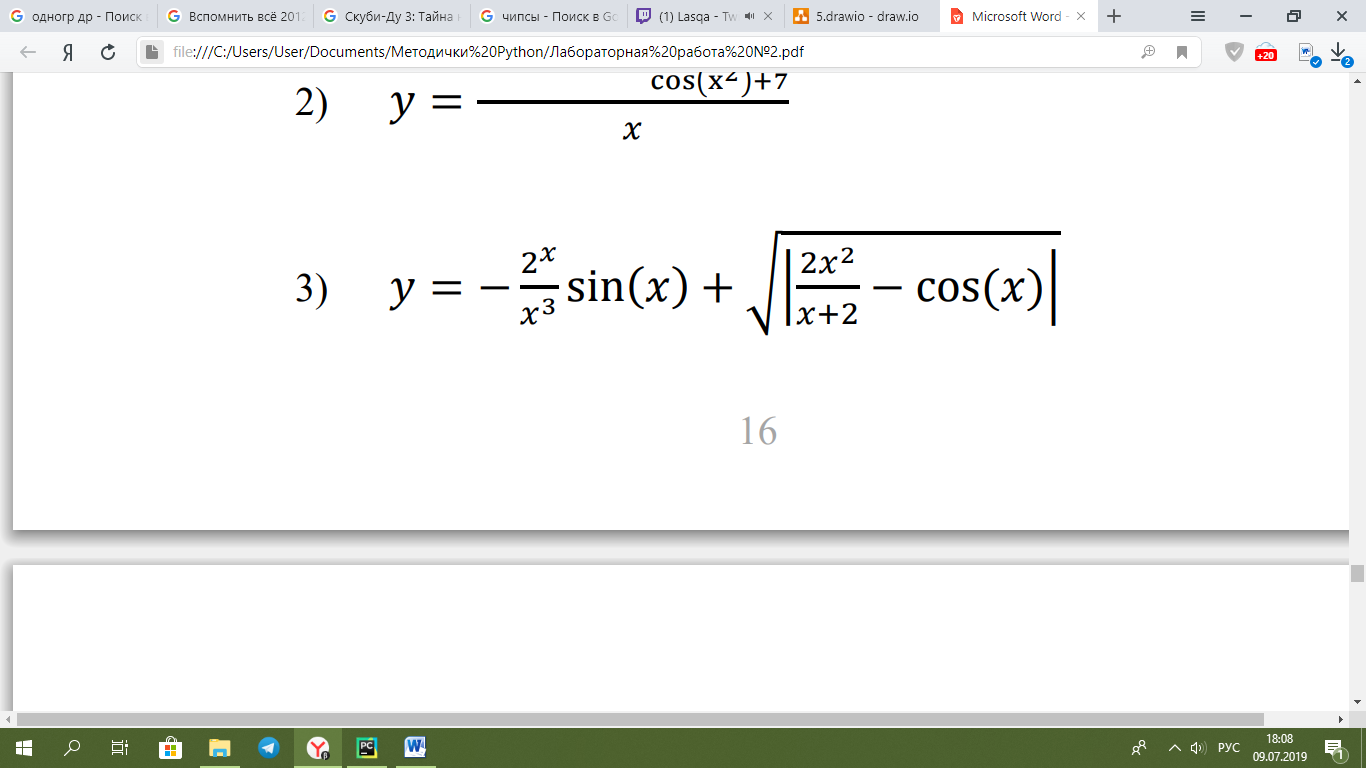
Получение базовых знаний об объектах программы на языке

Python, а также получение практических навыков написания программ

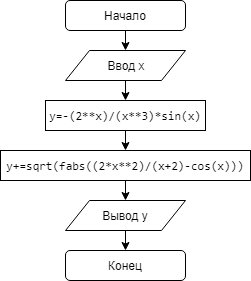
с линейной структурой.

**Задание**

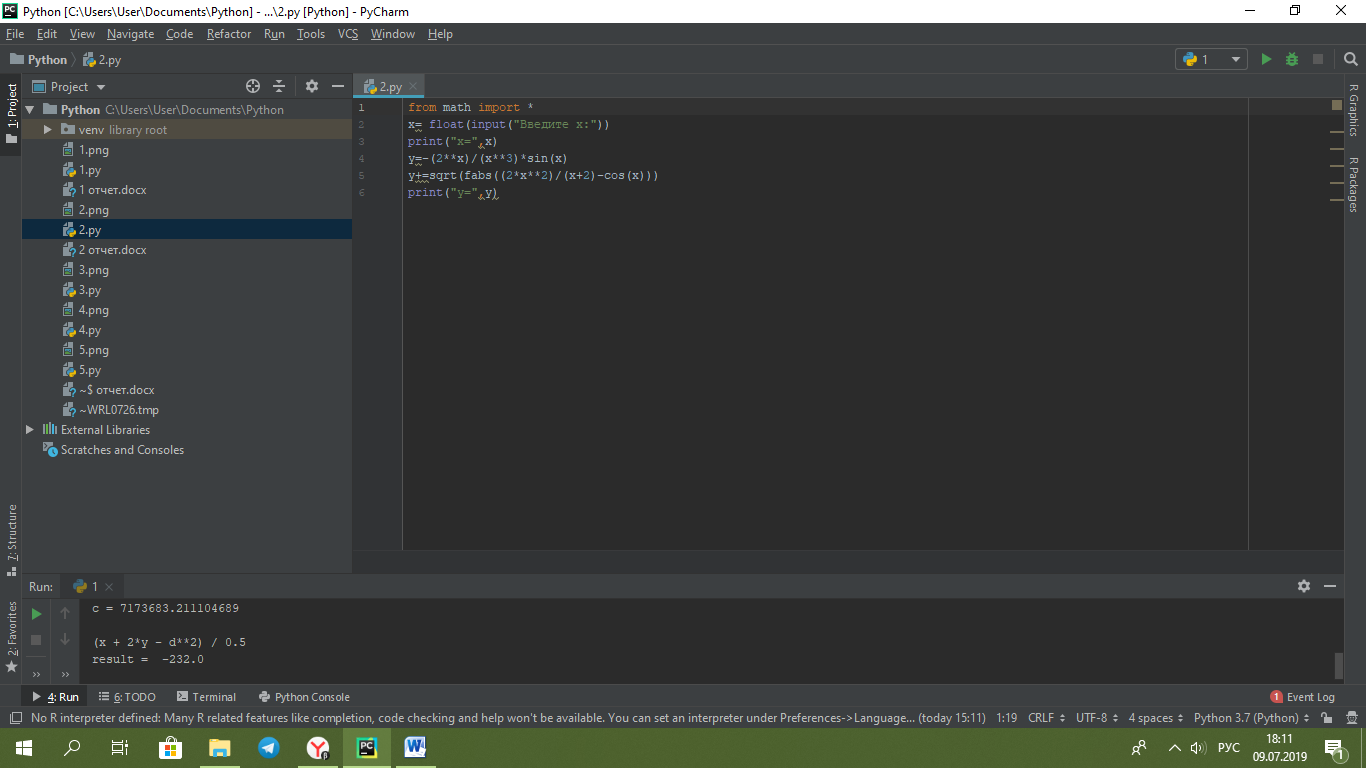
Составить алгоритм и написать программу, реализующую ввод исходных данных, вычисление значения уравнения и вывод результата



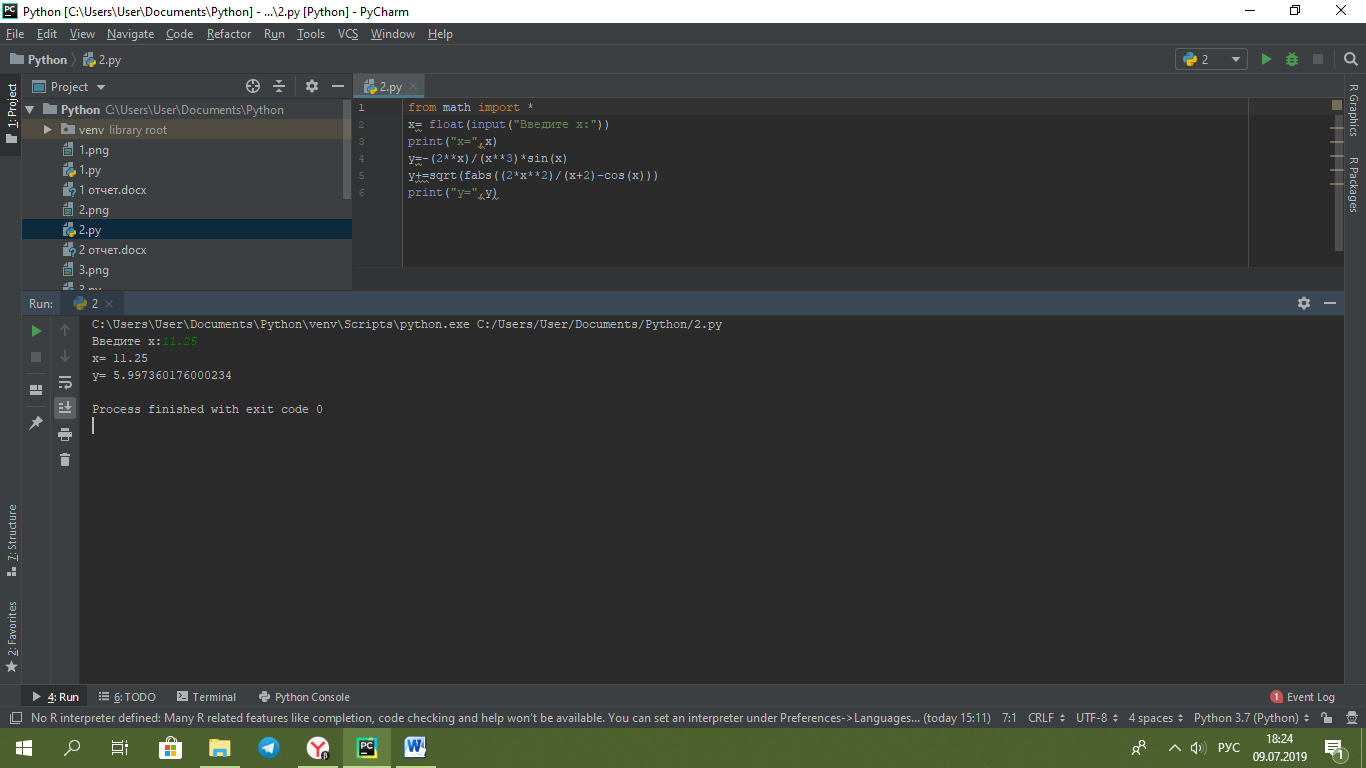
Блок-схема:



Код программы:



Результаты выполнения:



**Задание 3. Оператор условного перехода.**

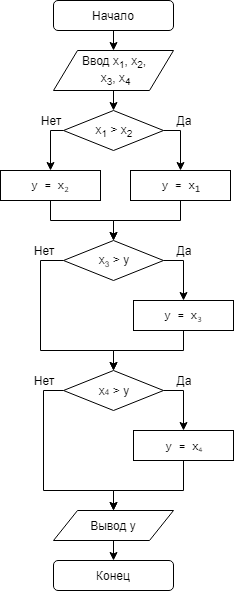
**Цель работы**

Освоение технологии проектирования алгоритма и программы разветвляющейся структуры с использованием оператора условного перехода на языке Python.

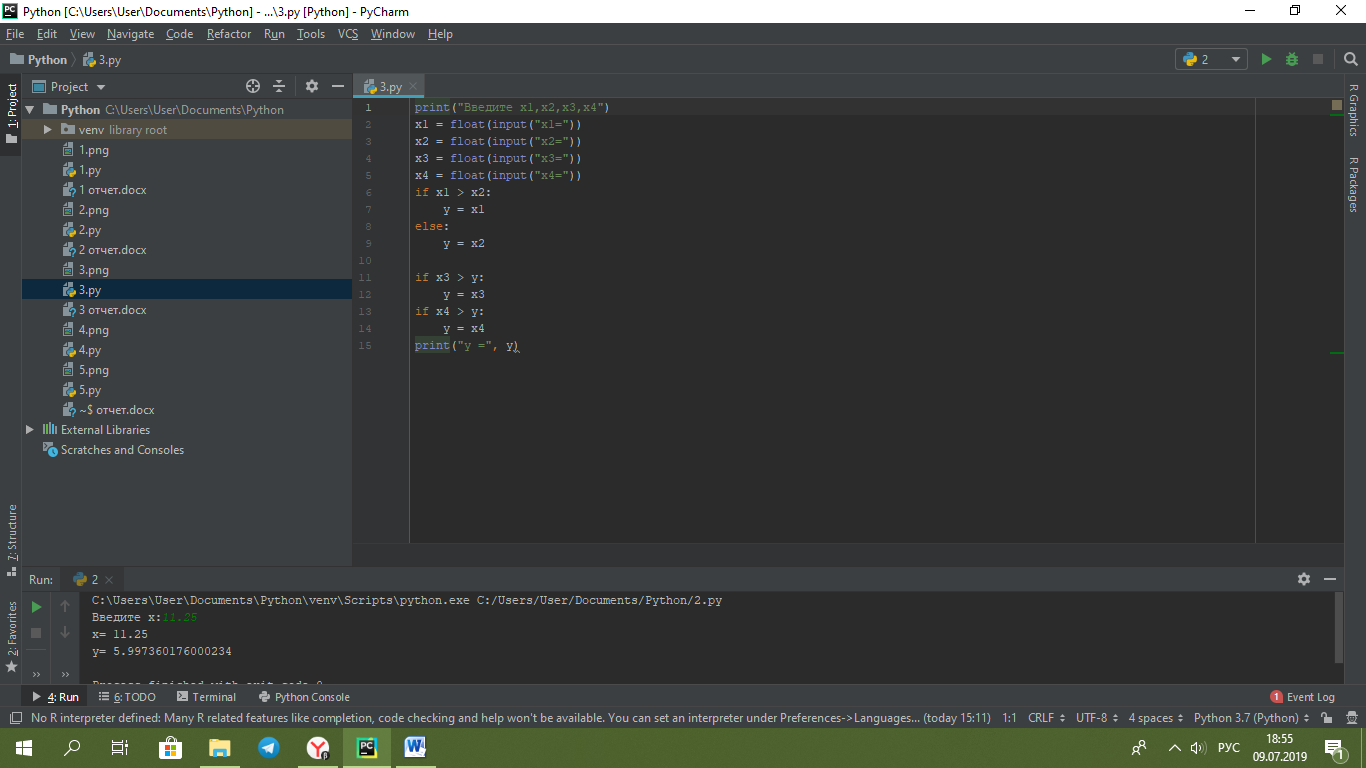
**Задание**

Определить максимальное значение из заданных x1, x2, x3, x4.

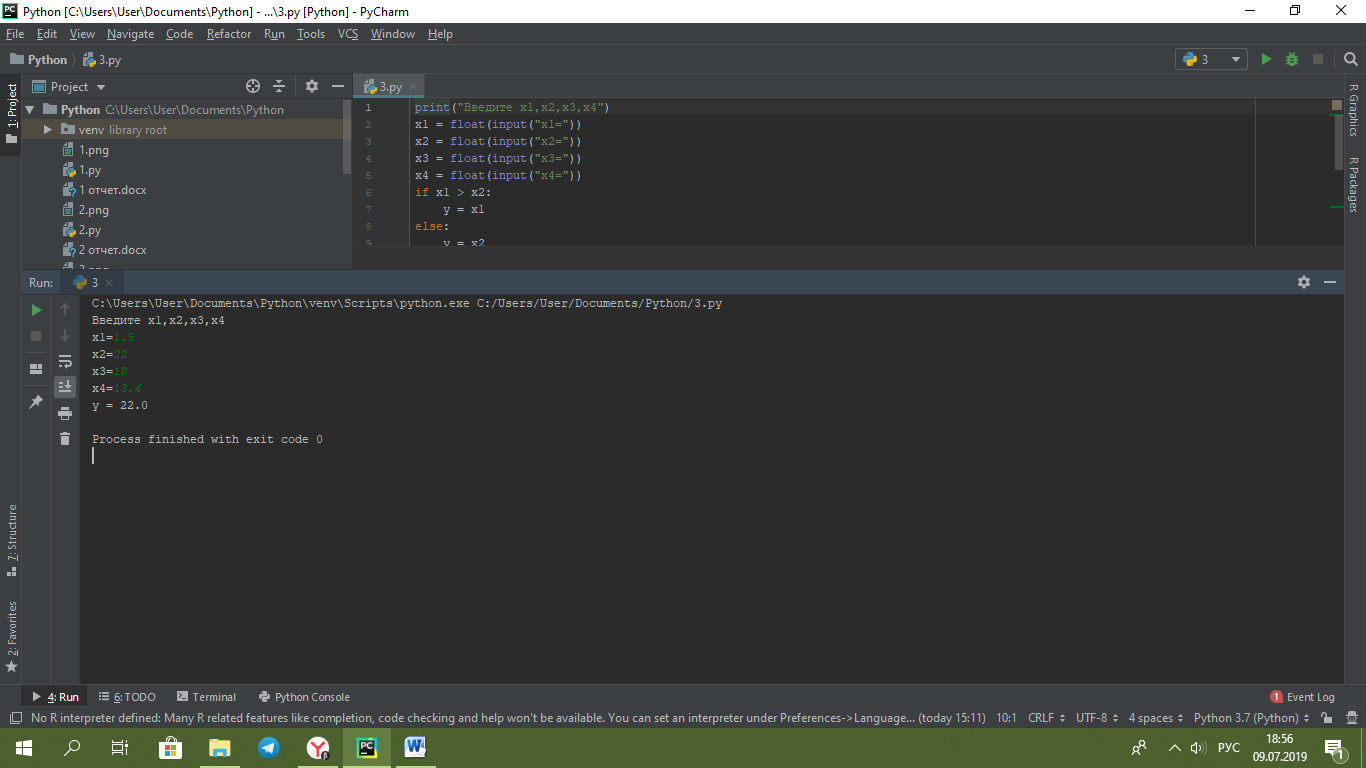
Блок-схема:



Код программы:



Результаты выполнения:



**Задание 4. Многоальтернативное ветвление.**

**Цель работы**

Изучение оператора if-elif-else для организации многоальтернативного ветвления.

**Задание**

Вывести в современных единицах измерения значение выбранной старой русской меры длины:

1) миля = 7,4676 км;

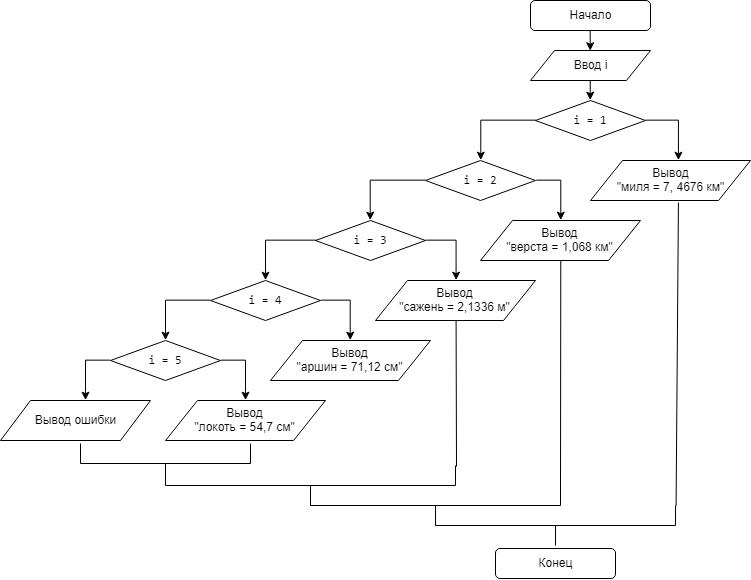
2) верста = 1,068 км;

3) сажень = 2,1336 м;

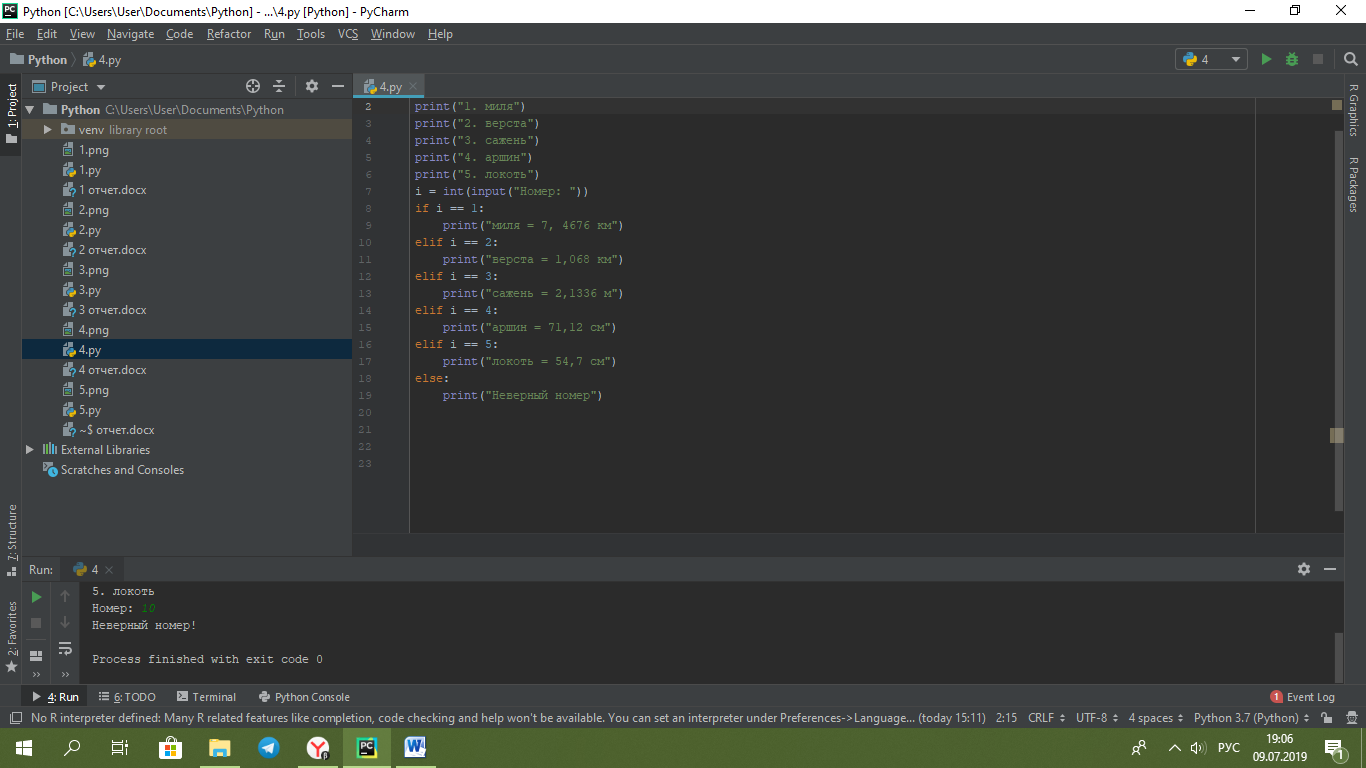
4) аршин = 71,12 см;

5) локоть = 54,7 см.

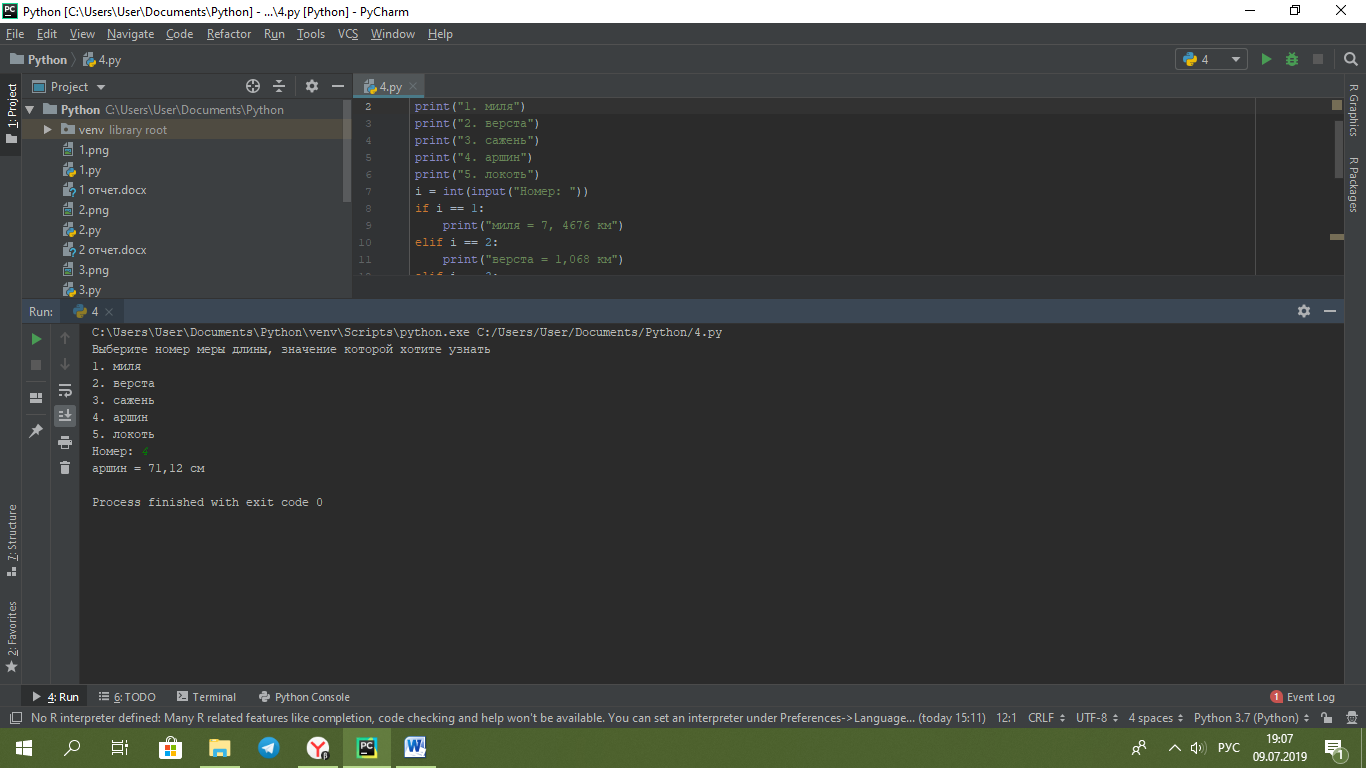
Блок-схема:



Код программы:



Результат выполнения:



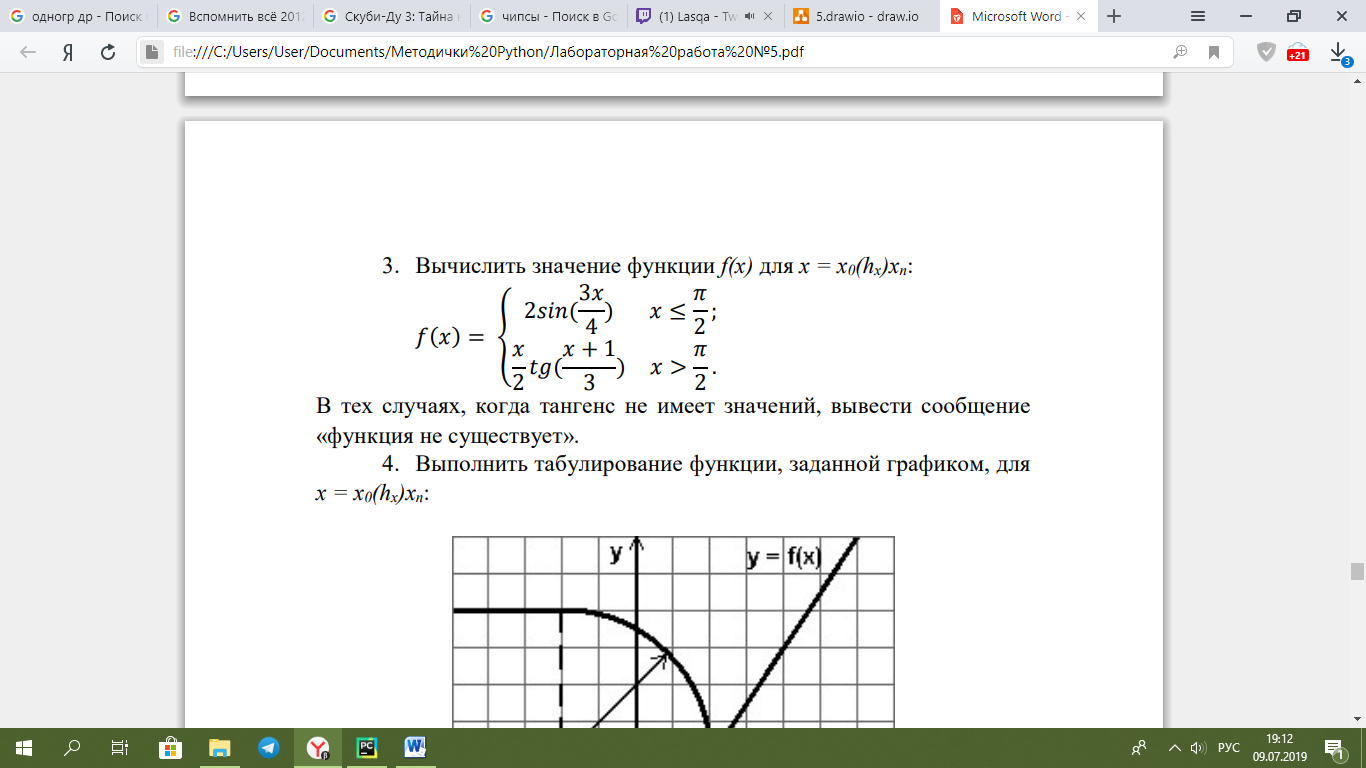
**Задание 5. Простейшие циклические программы. Оператор цикла с предусловием.**

**Цель работы**

Знакомство с простейшими приемами программирования циклических программ.

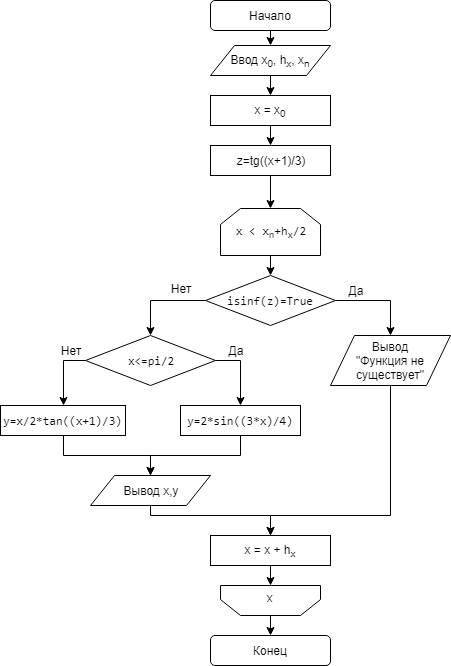
**Задание**

Вычислить значение функции f(x) для x = x0(hx)xn:

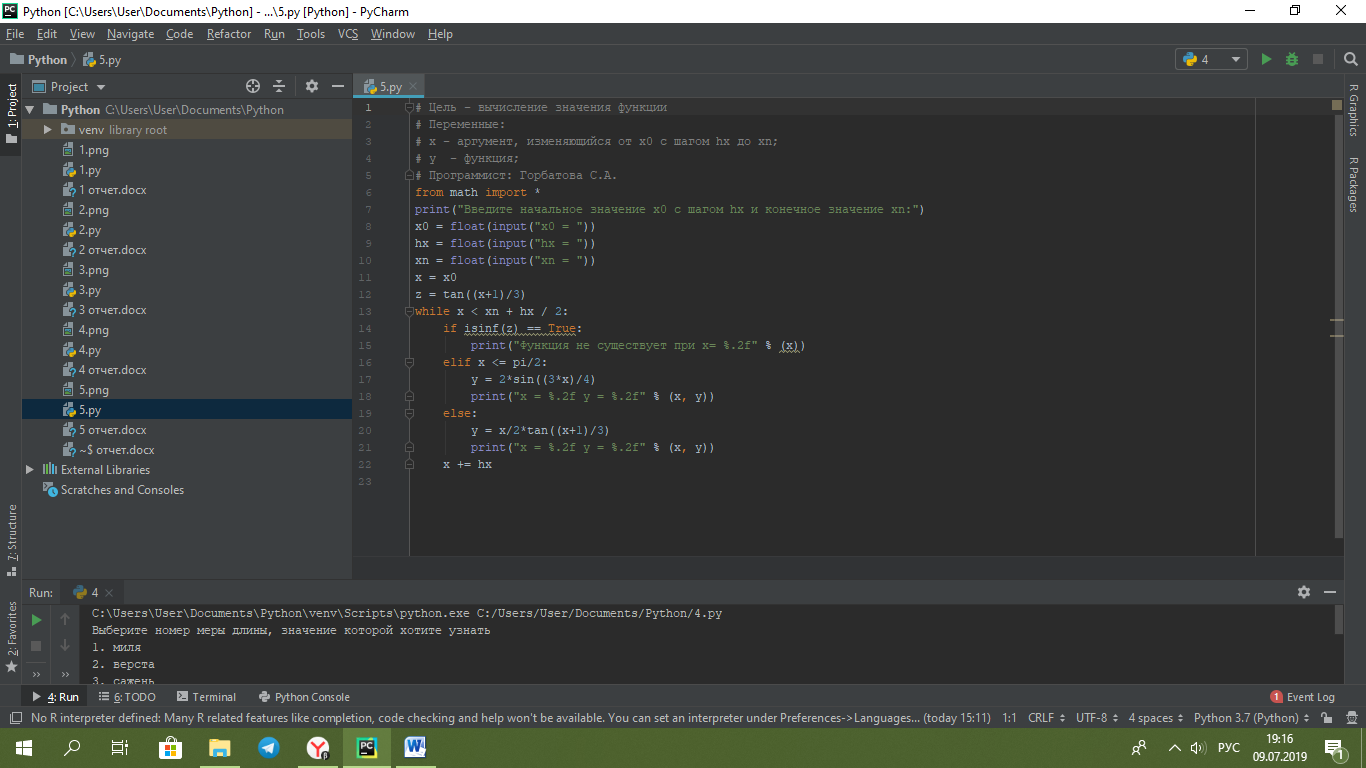


В тех случаях, когда тангенс не имеет значений, вывести сообщение «функция не существует».

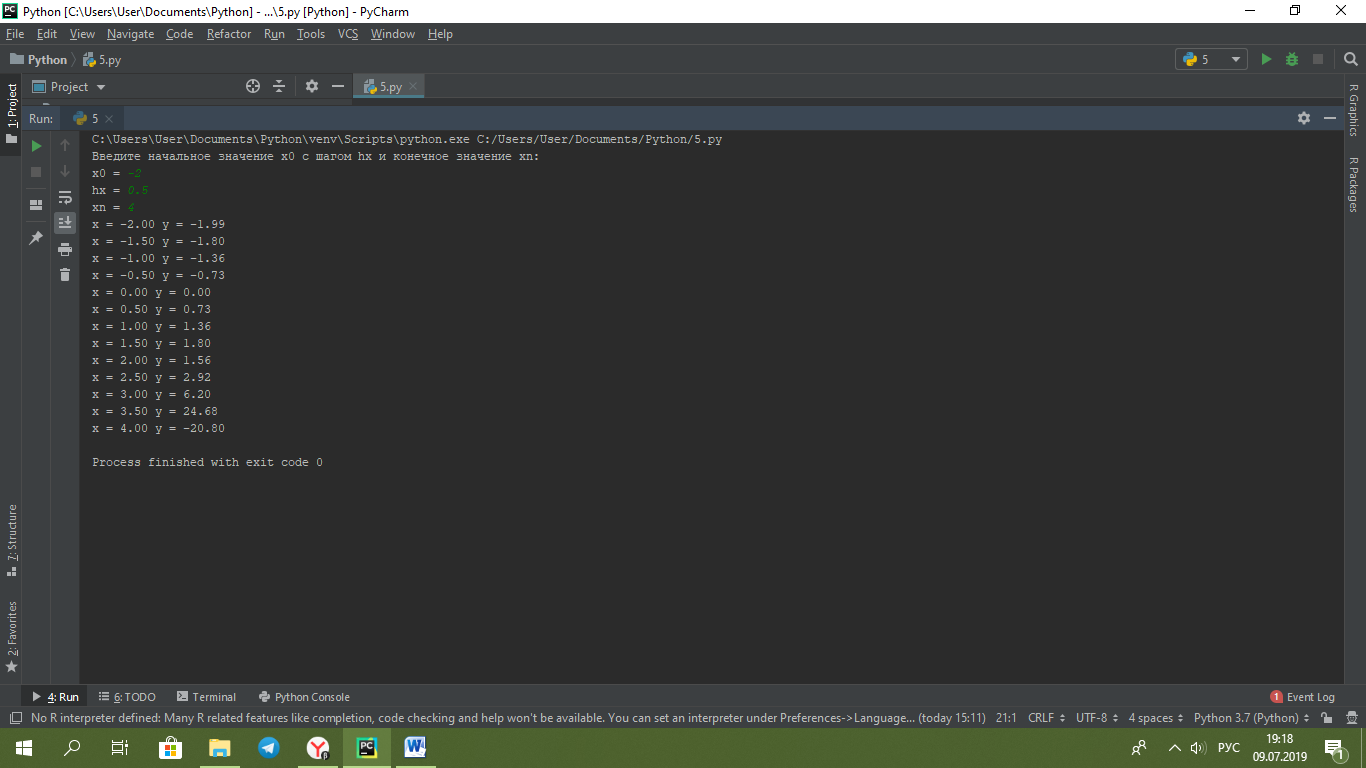
Блок-схема:



Код программы:



Результаты выполнения:



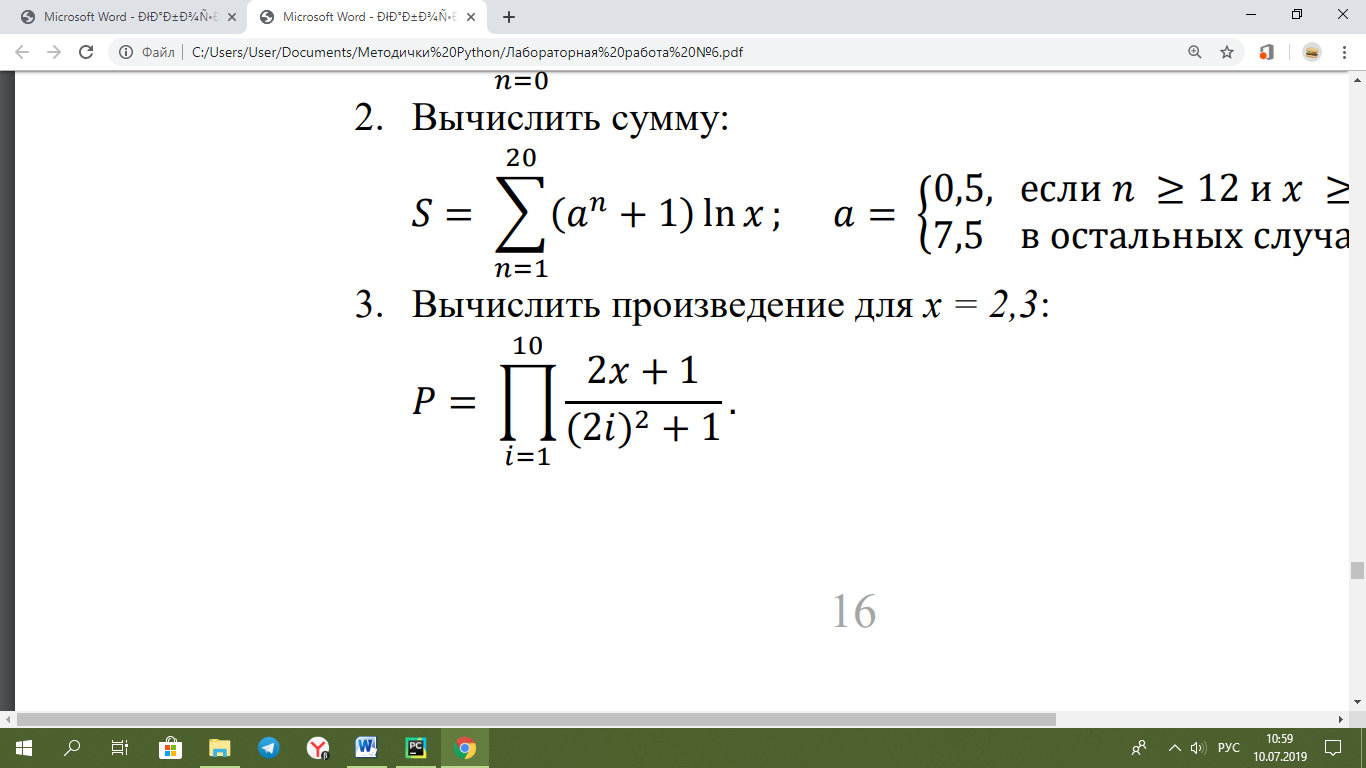
**Задание 6. Оператор цикла с заголовком. Вычисление конечных сумм и произведений.**

**Цель работы**

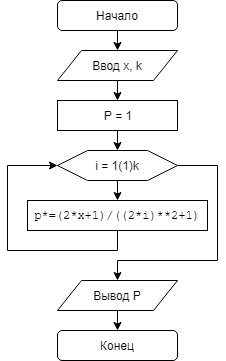
Получение практических навыков реализации цикла с заголовком с использованием операторов for, continue и break, функции range и конструкции else.

**Задание**

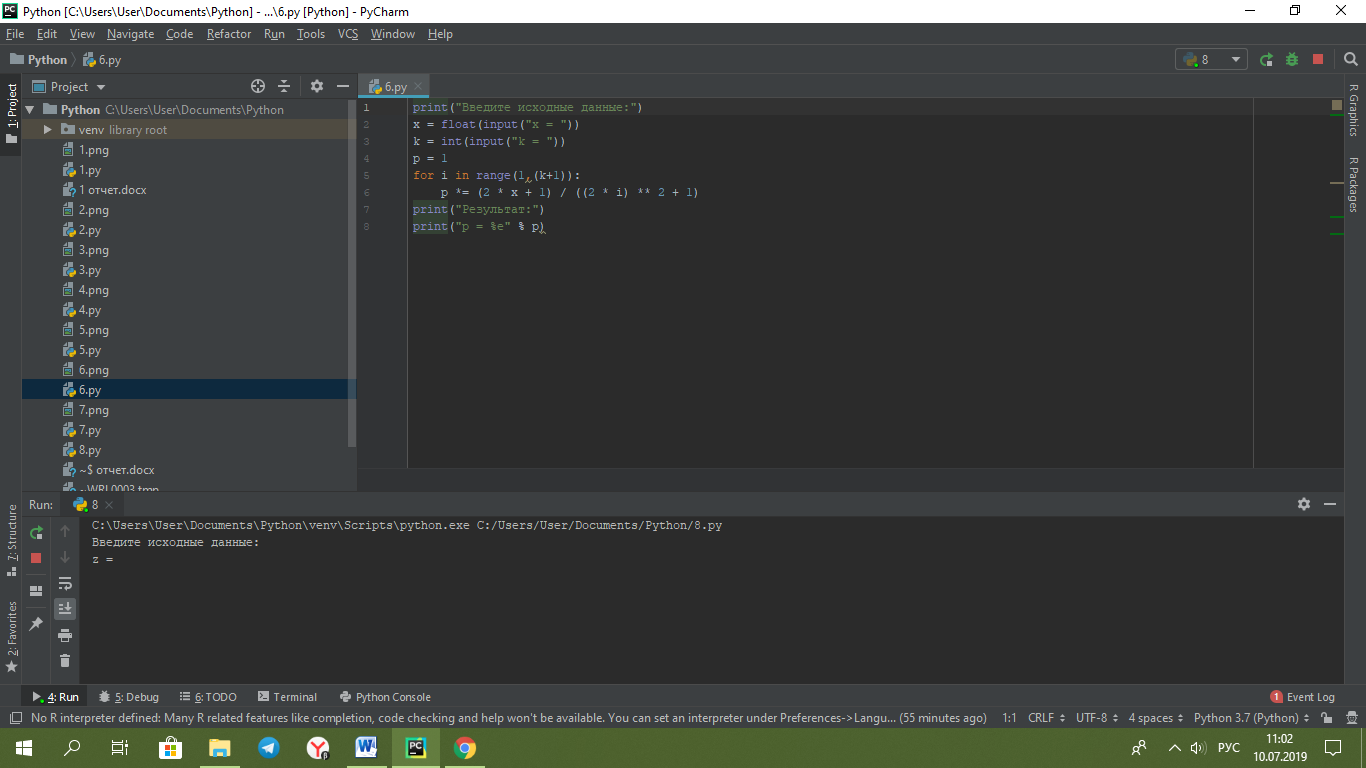
Вычислить произведение для x = 2,3:



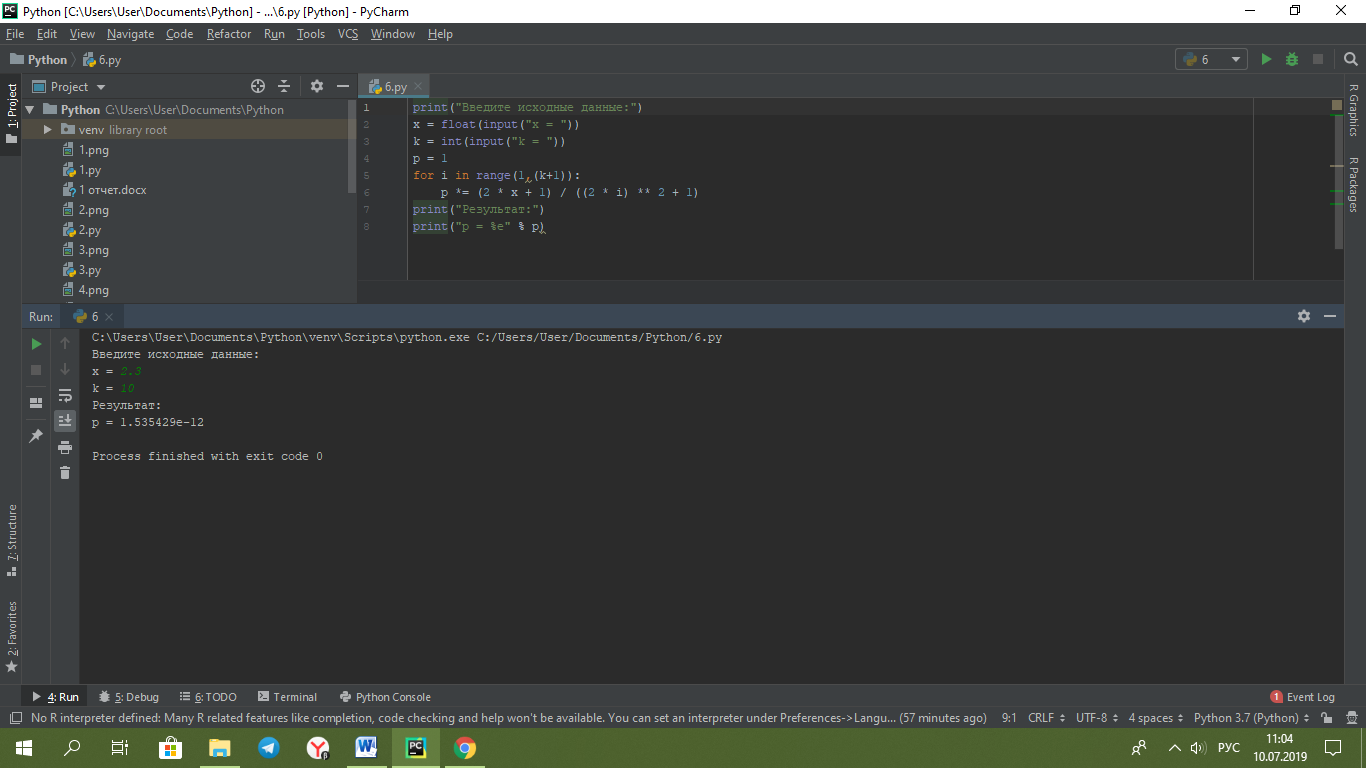
Блок-схема:



Код программы:



Результат выполнения:



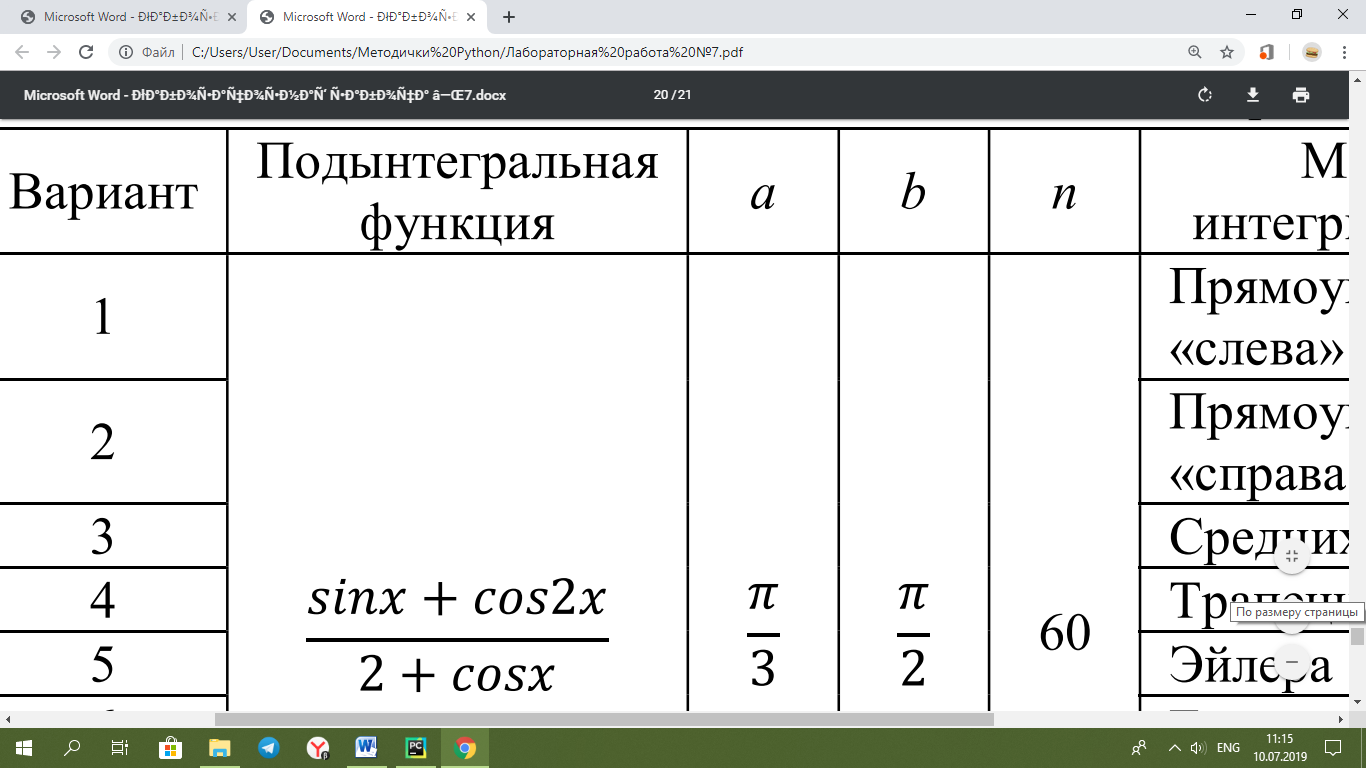
**Задание 7. Алгоритмы численного интегрирования.**

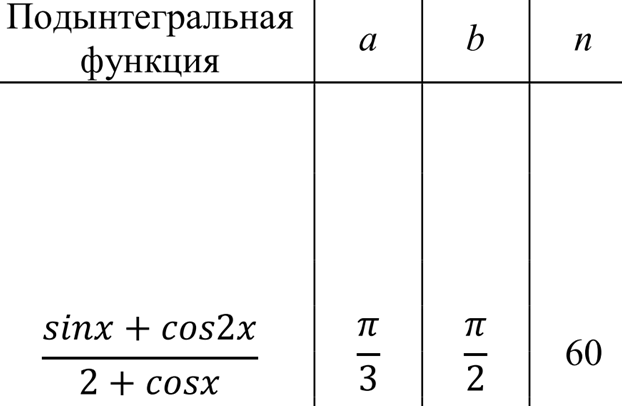
**Цель работы**

Знакомство с методами численного вычисления определенных интегралов, а также закрепление практических навыков использования оператора цикла с предусловием.

**Задание**

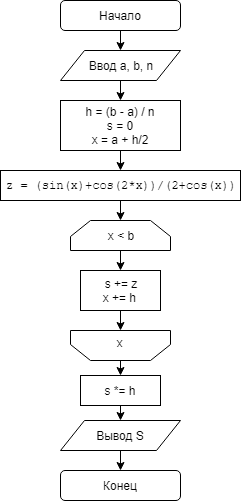
Составить алгоритм и написать программу, реализующие численное интегрирование. Сравнить полученные результаты при числе разбиений анализируемого отрезка, равному n и 2n.



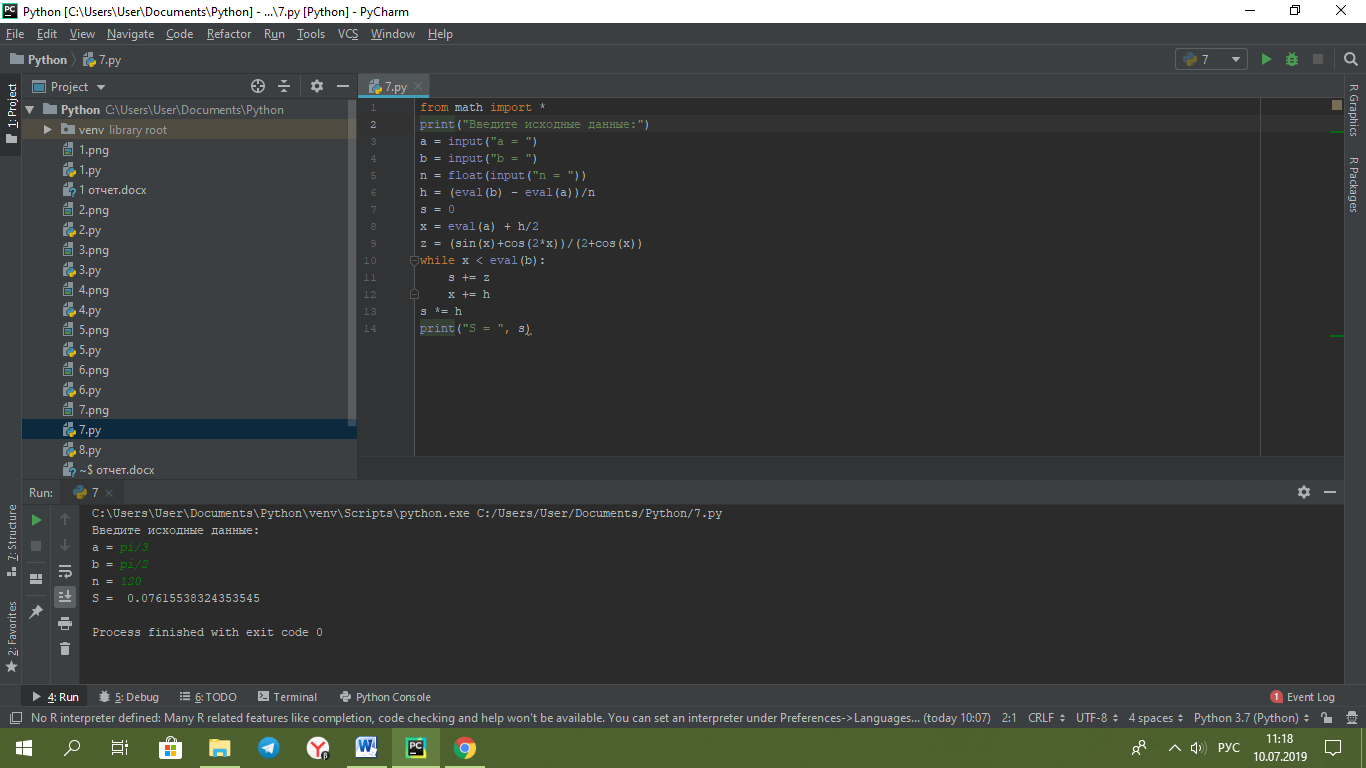


Метод интегрирования: средних

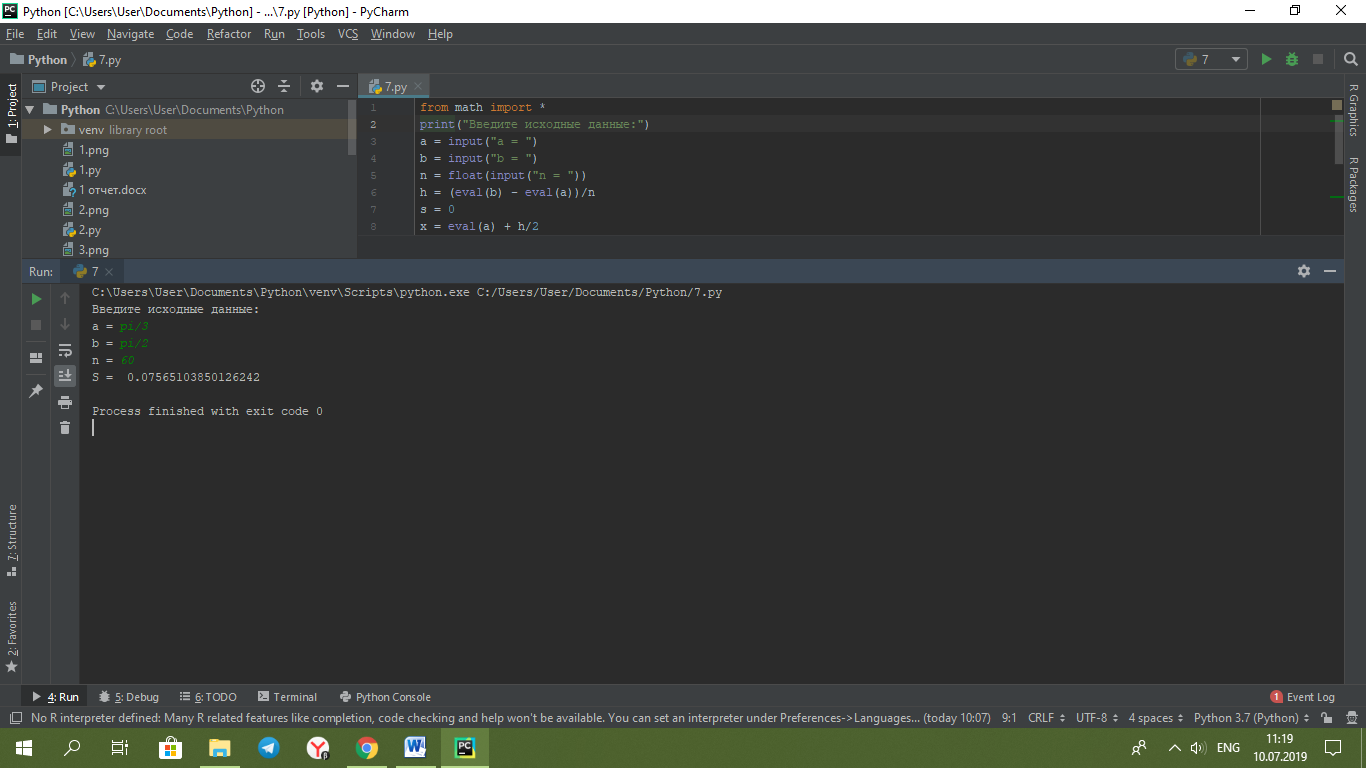
Блок-схема:

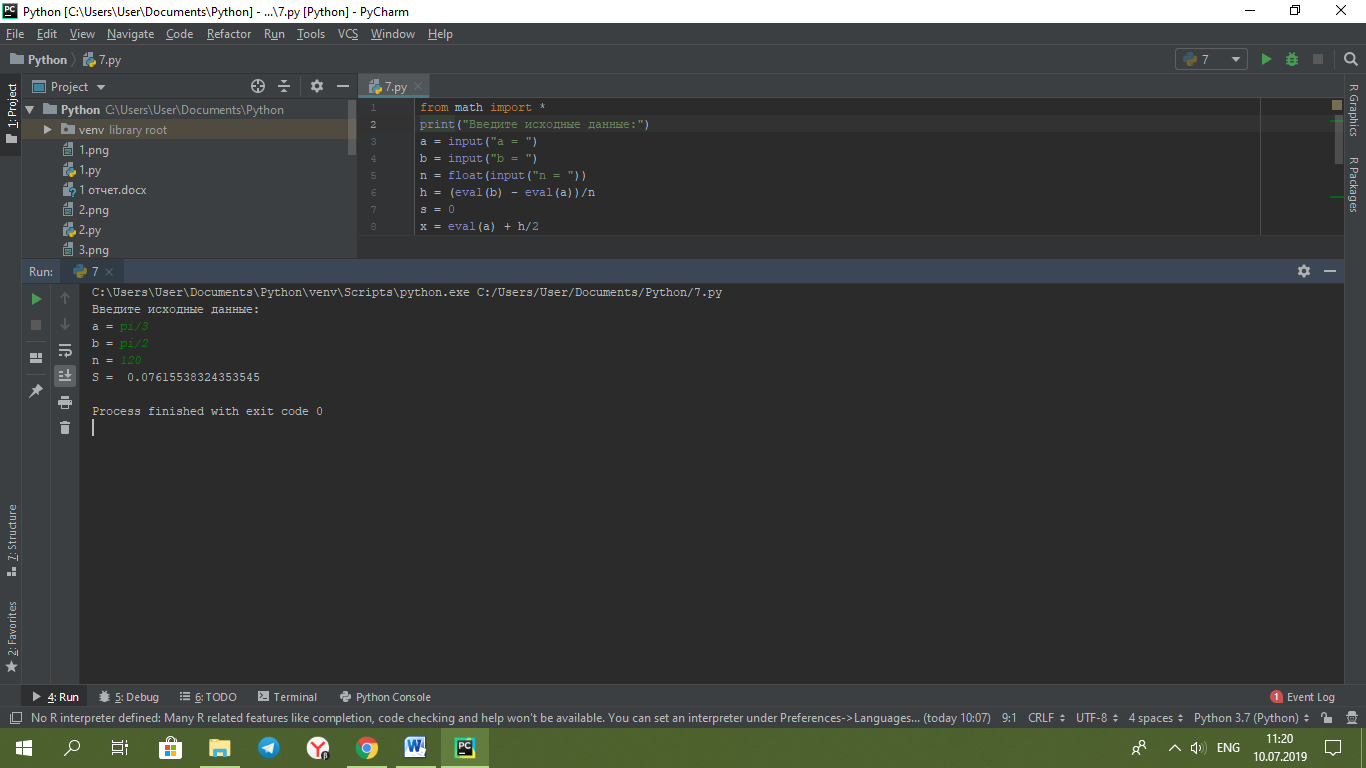


Код программы:



Результат выполнения:





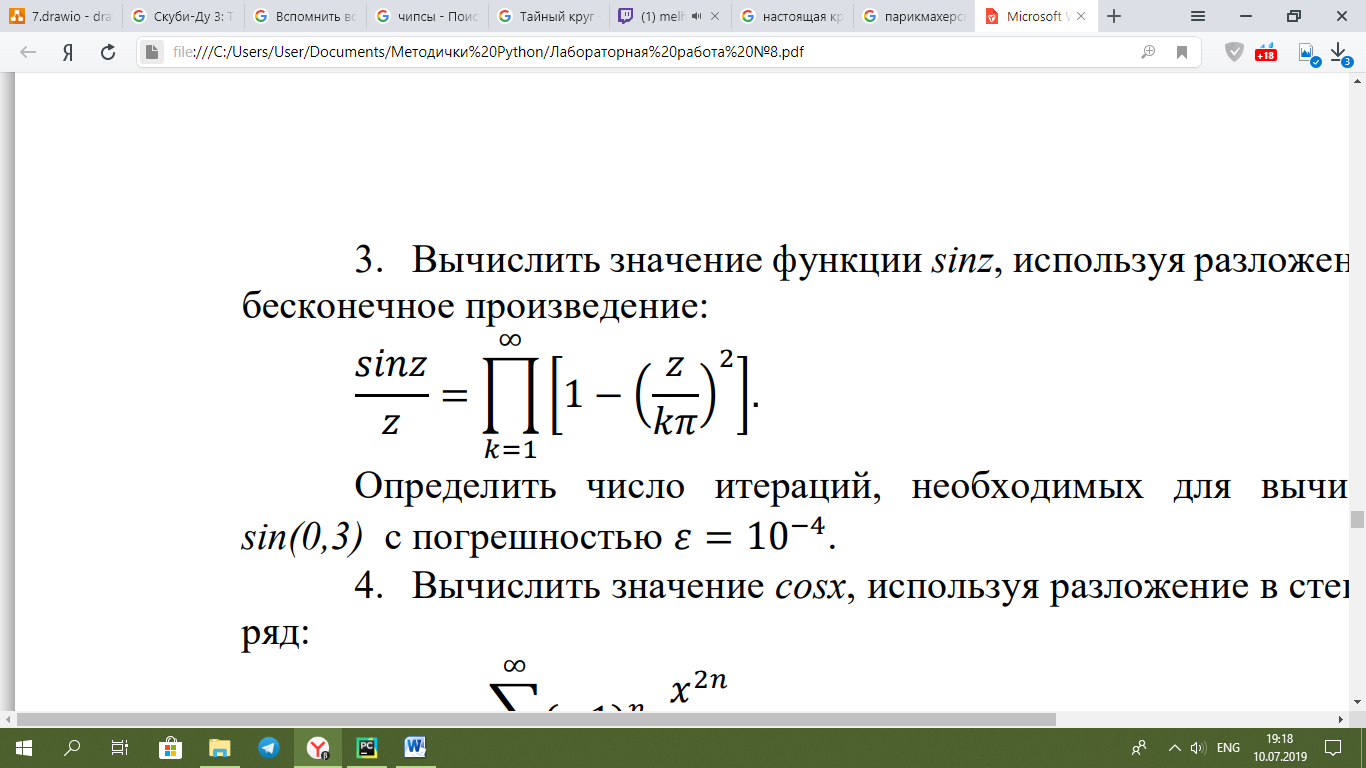
**Задание 8. Итерационный цикл. Вычисление суммы бесконечного ряда.**

**Цель работы**

Получение практических навыков программирования итерационных циклов на примере определения суммы бесконечных рядов.

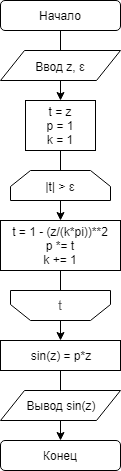
**Задание**

Вычислить значение функции sinz, используя разложение ее в бесконечное произведение:

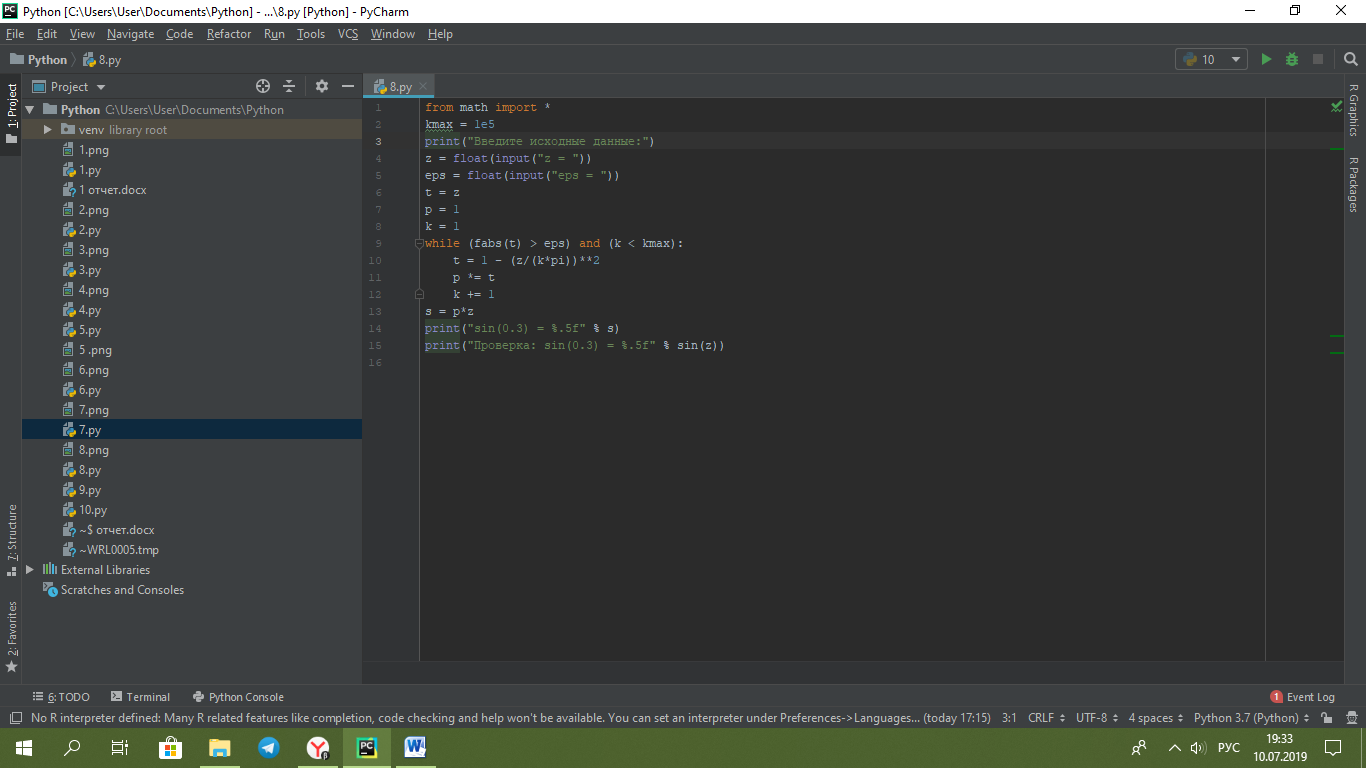


Определить число итераций, необходимых для вычисления sin(0,3) с погрешностью ε = 10-4 .

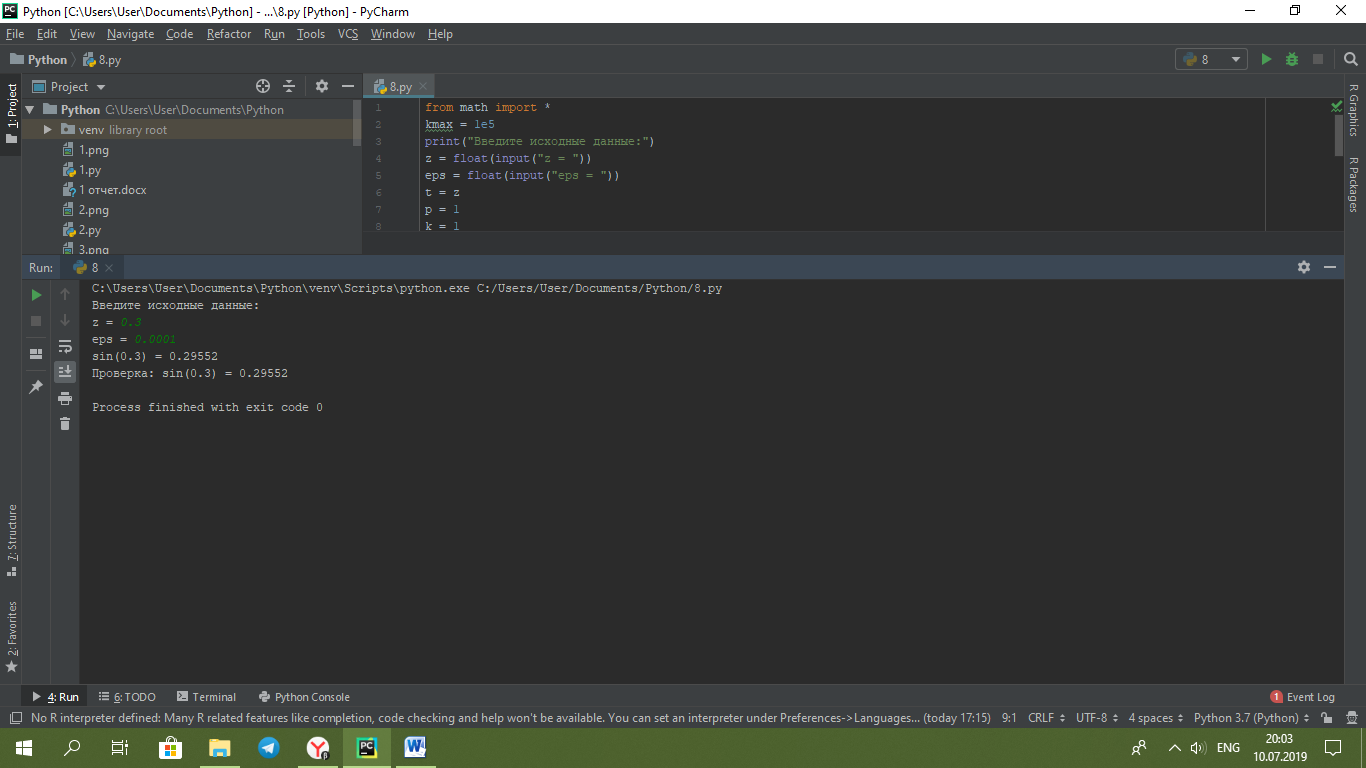
Блок-схема:



Код программы:



Результат выполнения:



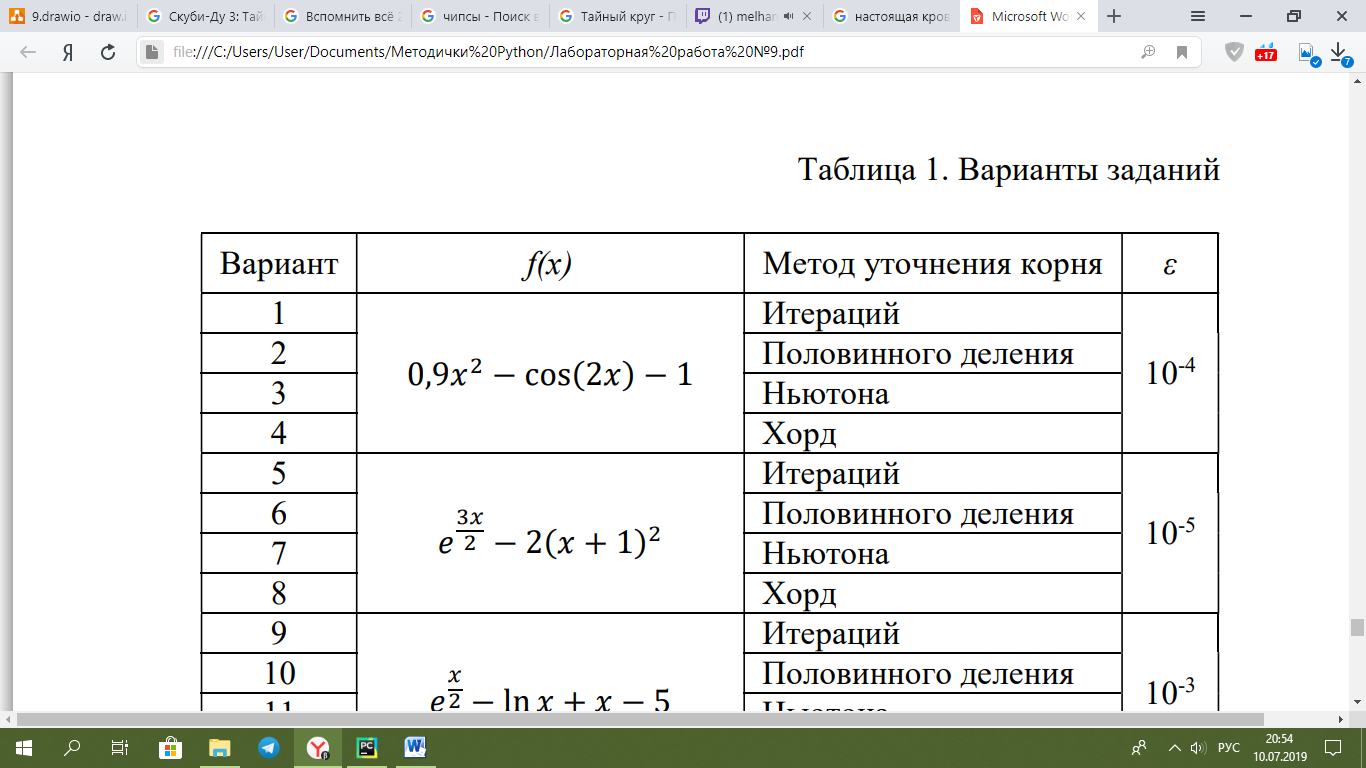
**Задание 9. Итерационный цикл. Численные алгоритмы уточнения корней трансцендентных и нелинейных алгебраических уравнений.**

**Цель работы**

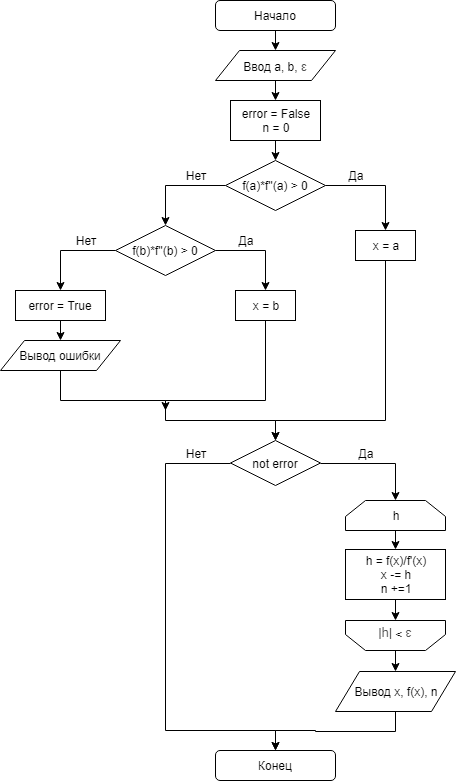
Закрепление навыков проектирования алгоритмов и программ со структурой итерационных циклов.

**Задание**

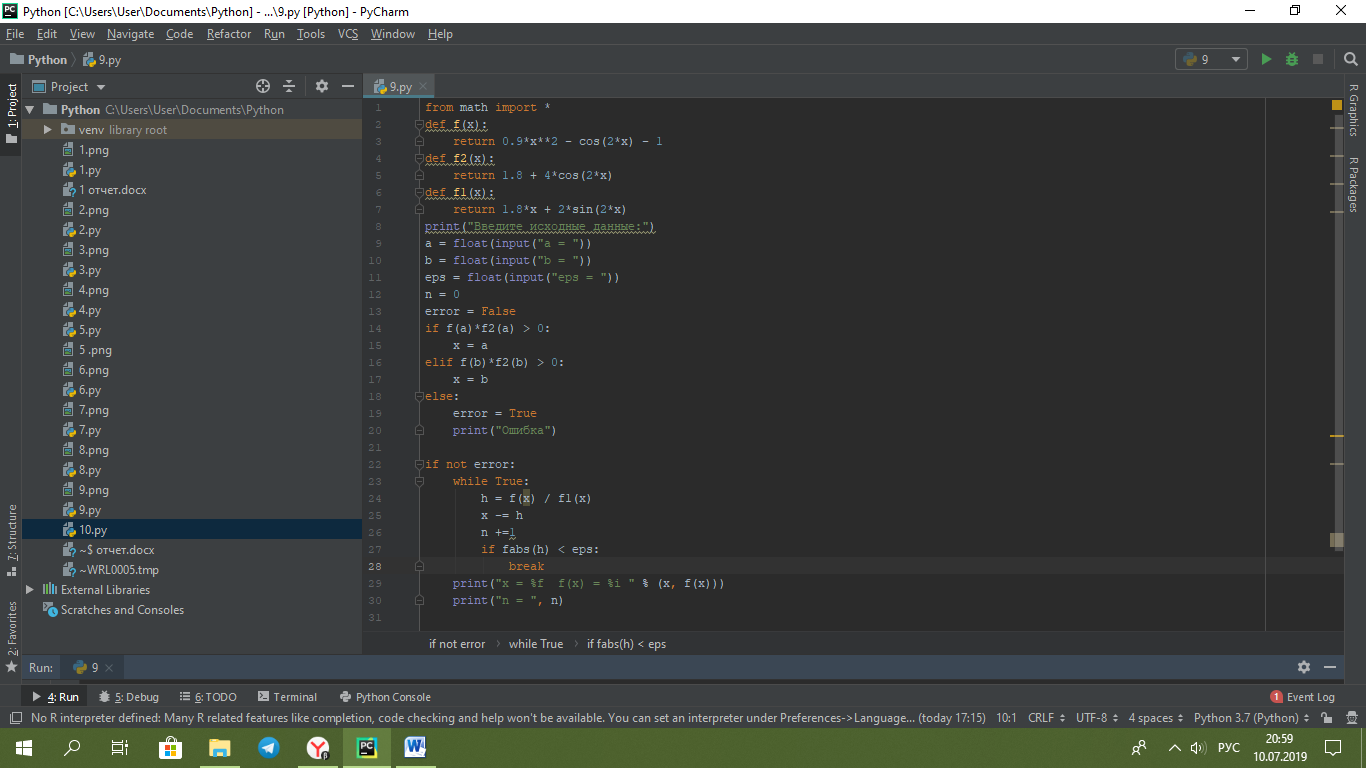
Cоставить программу отделения одного из корней уравнения f(x) = 0. После этого необходимо составить программу уточнения корня в пределах найденного отрезка [a, b] одним из заданных методов (см. табл. 1). Предусмотреть защиту от зацикливания путем ограничения числа повторений итерационного цикла. Вывести значение числа итераций, за которое удается найти значение корня с заданной погрешностью. Для контроля правильности результата включить в программу вычисление и вывод значения функции при найденном значении корня.



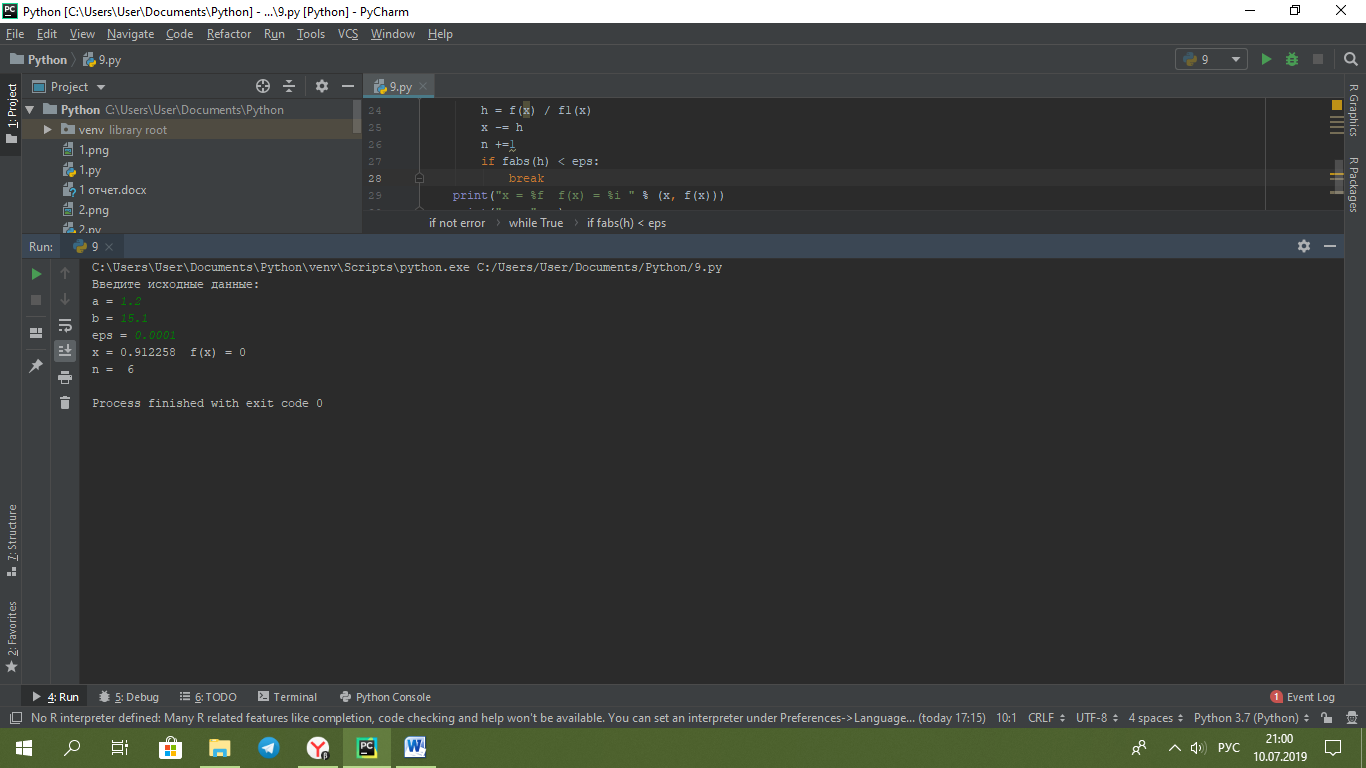
Блок-схема:



Код программы:



Результат выполнения:



**Задание 10. Проектирование алгоритмов и программ со структурой вложенных циклов.**

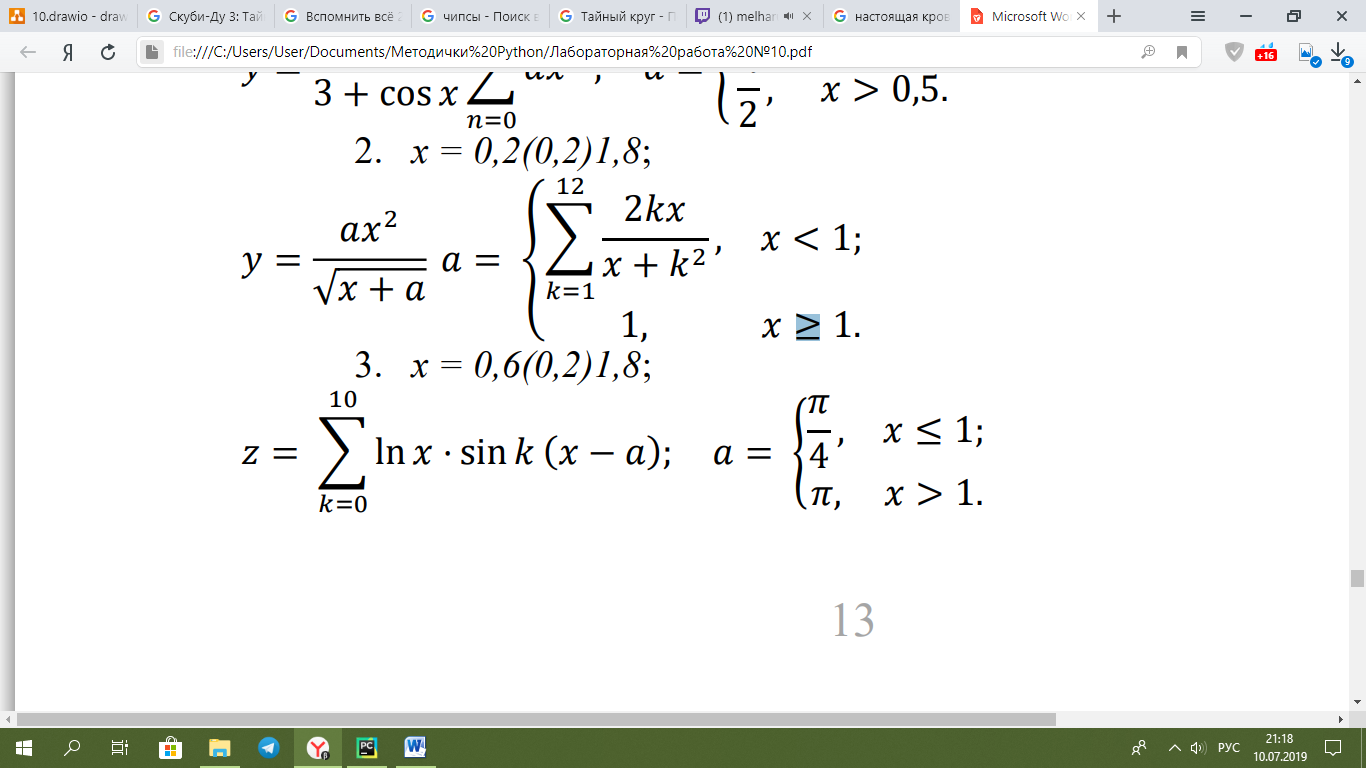
**Цель работы**

Получение практических навыков проектирования алгоритмов и программ со структурой вложенных циклов.

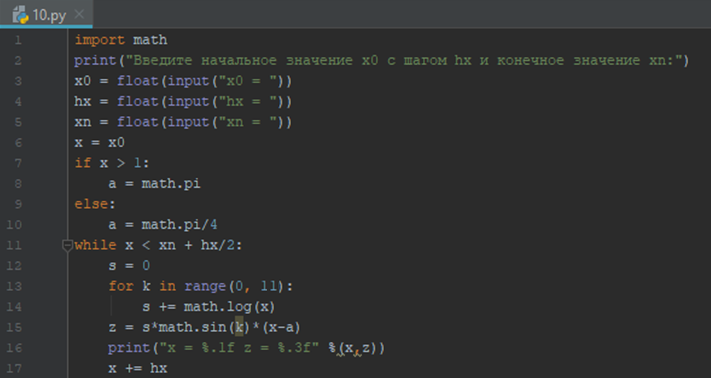
**Задание**

Используя метод нисходящего пошагового проектирования, разработать схему алгоритма и составить программу для вычисления функции при заданных значениях аргументов для своего варианта задания. Вывести результаты вычислений.

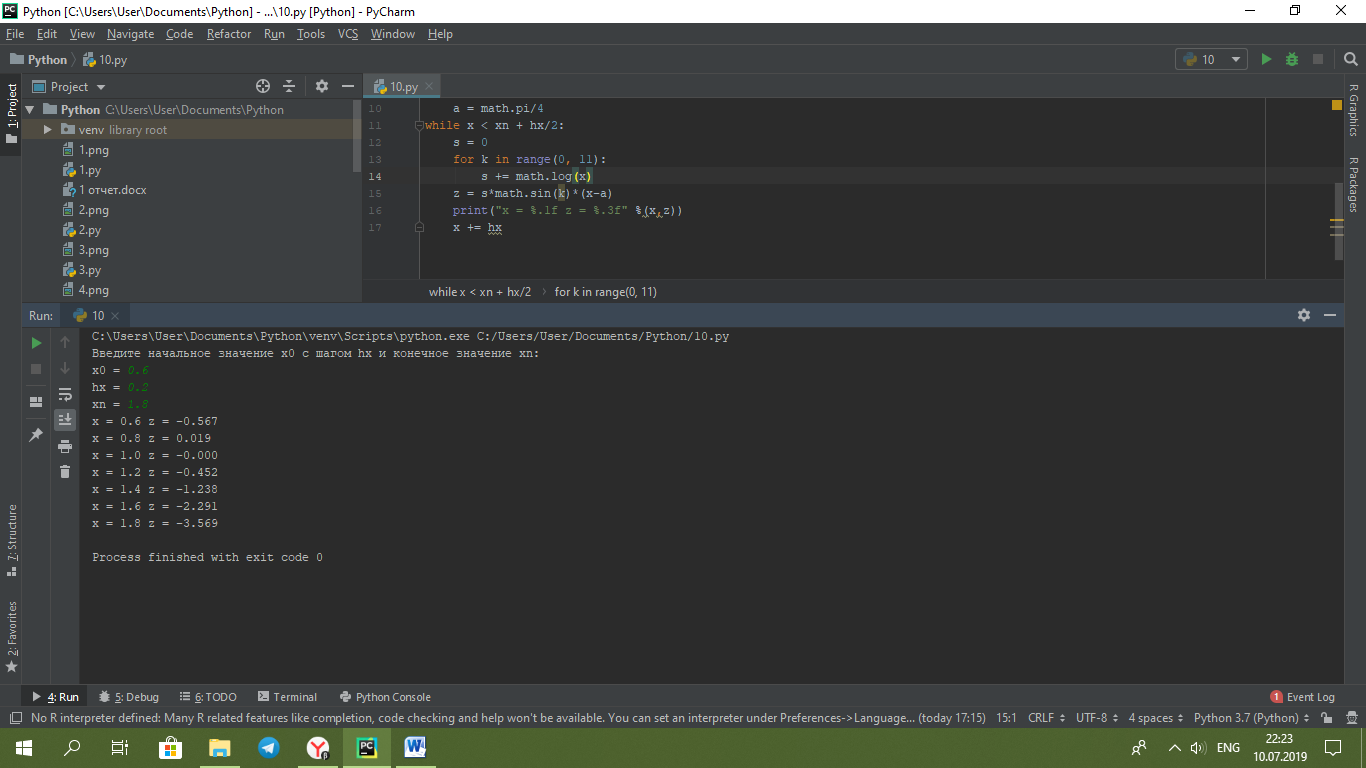
x = 0,6(0,2)1,8;



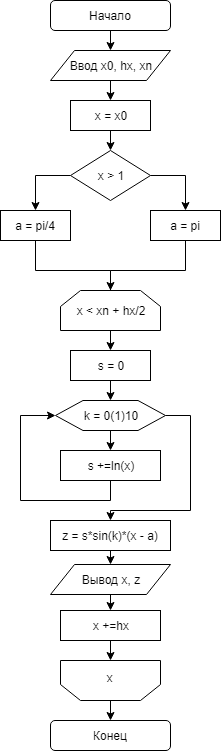
Код программы:



Результат выполнения:



Блок-схема:



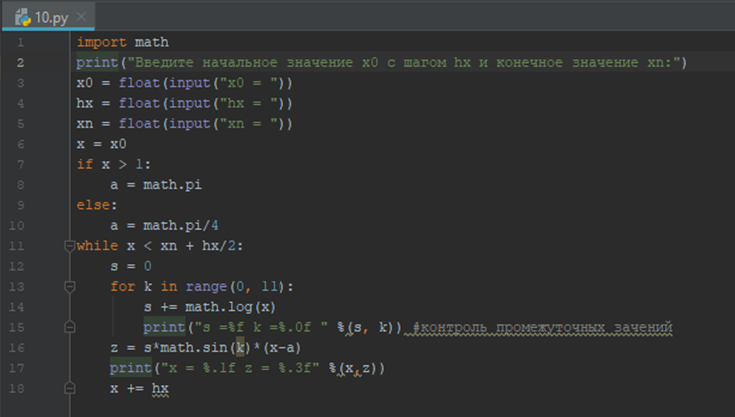
**Задание 11. Отладка программ в среде PyCharm.**

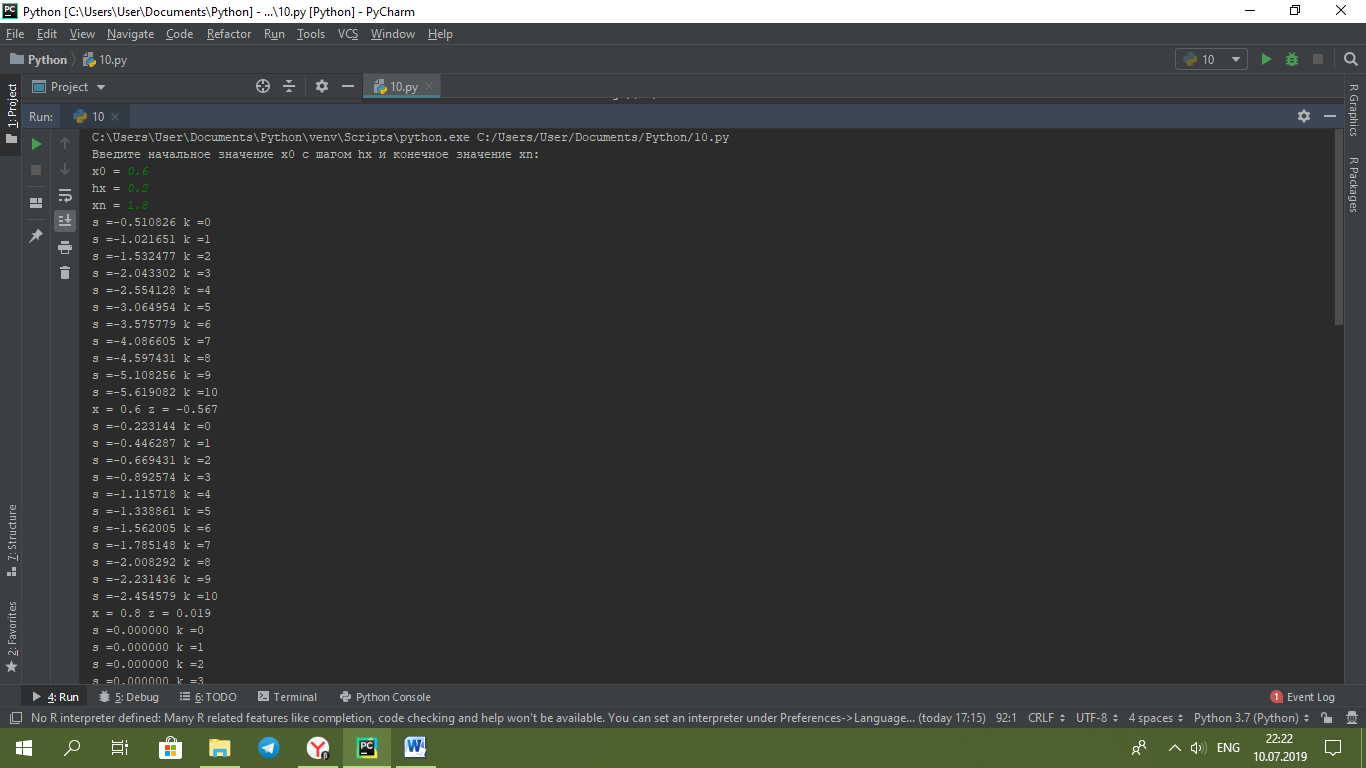
**Цель работы**

Получение практических навыков отладки программ на языке Python с использованием средств среды PyCharm.

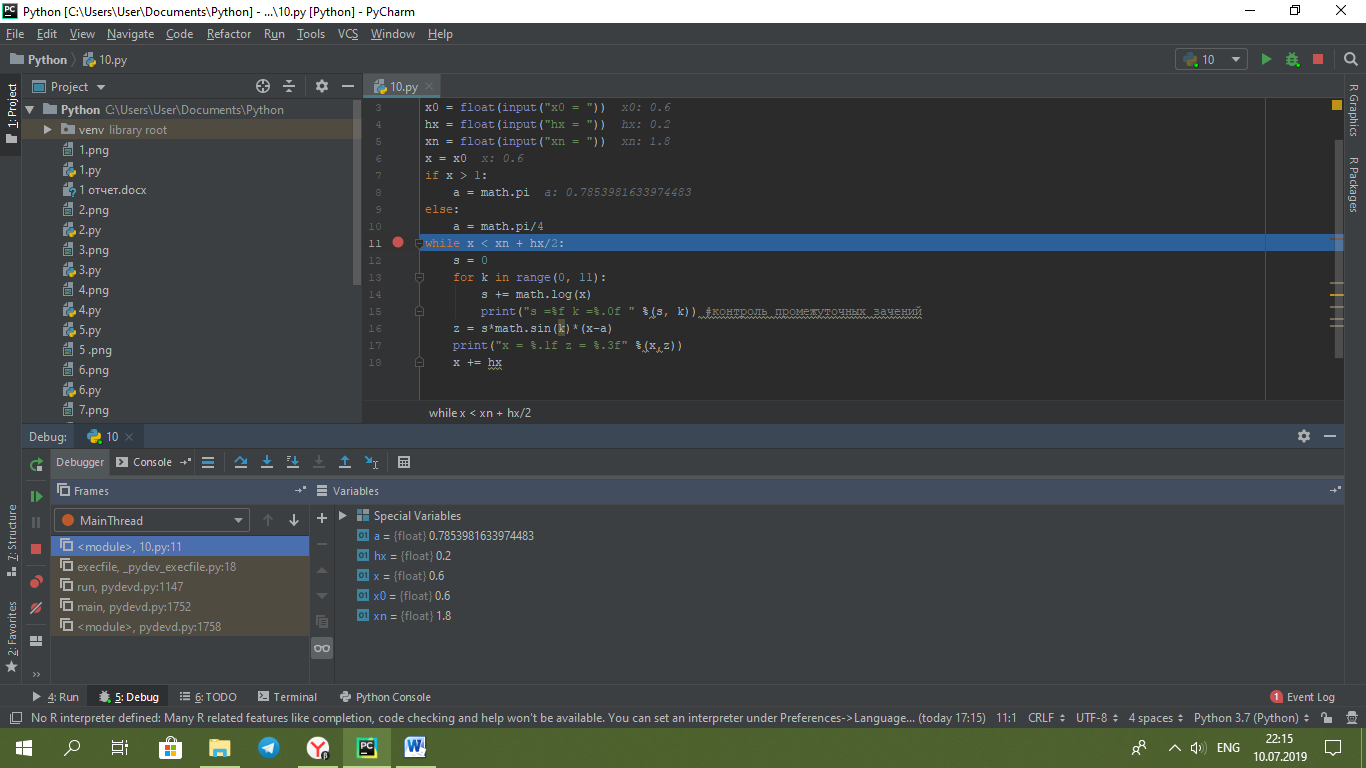
**Задания**

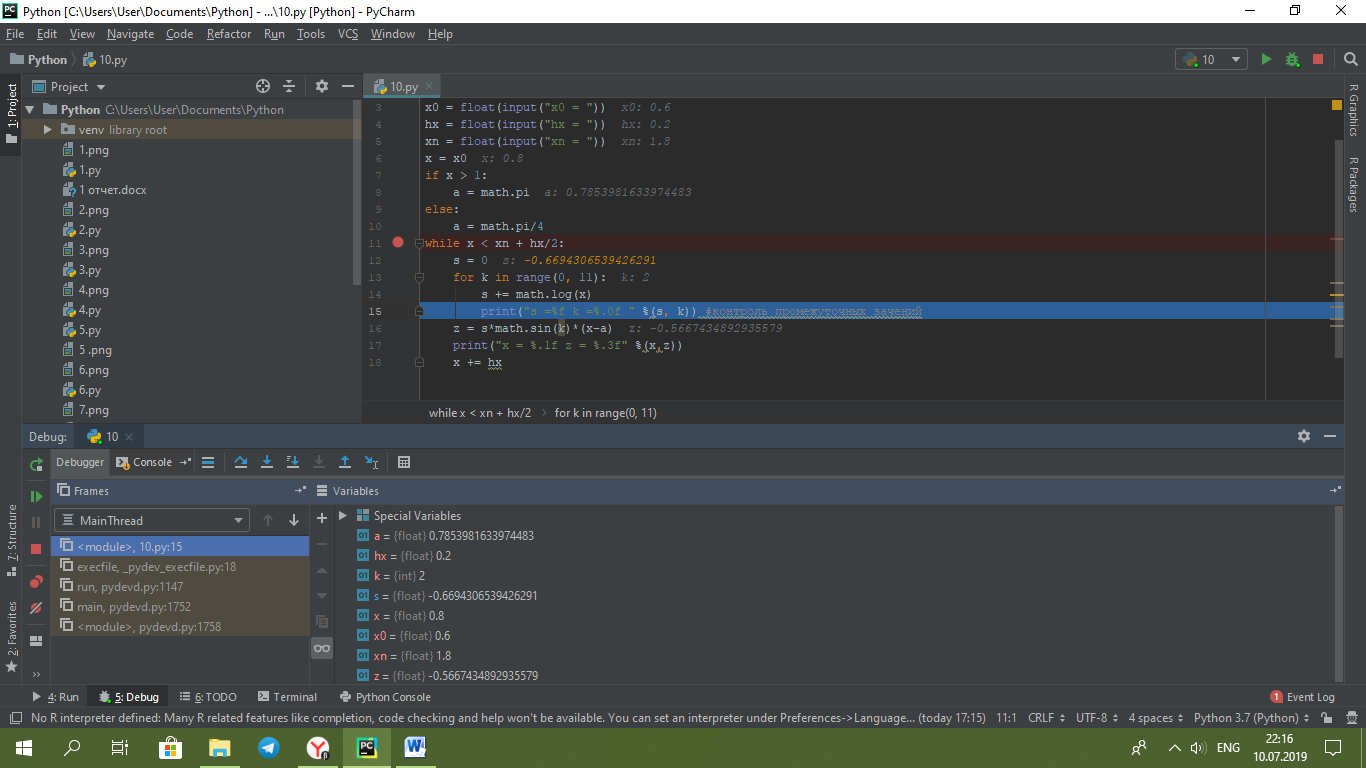
*Задание 1*. Проведите отладку своей программы, составленной при выполнении лабораторной работы №10, с помощью отладочных печатей и на специально подобранных тестовых наборах параметров программы. Проверьте выполнение всех ветвей и циклов.





*Задание 2.* Для той же программы проведите отладку, используя встроенные средства среды PyCharm. Выполните на экране трассировку программы, установку и использование точек останова, работу с окном отладки. Обоснуйте введение точек останова и контролируемых в окне отладки параметров программы, а также проведите сравнение результатов работы программы с результатами контрольных вычислений.





**Заключение**

В ходе учебной практики были изучены основы языка программирования Python и были выполнены задания.