ГУАП  
КАФЕДРА № 41

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| доцент |  |  |  | А.В. Аграновский |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ |
| ЦИКЛИЧЕСКИЕ КОНСТРУКЦИИ |
| по курсу: ТЕХНОЛОГИИ ПРОГРАММИРОВАНИЯ |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № | М023 |  |  |  | Д.А. Трегуб |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2022

ПОСТАНОВКА ЗАДАЧИ

Задание № 3.2.  
Вычислите приближенное значение функции 𝑓(𝑥) посредством вычисления соответствующего ряда. Вычисления следует остановить тогда, когда значение очередного члена ряда оказалось меньше заданного ɛ

|  |  |
| --- | --- |
| № вар | Функция |
| 2 |  |

Требования к программе:  
Цель программы: вывод значения функции в заданной точке или сообщения об ошибке. На вход подаётся аргумент x, по которому нужно вычислить функцию, а также точность вычисления. Выходными данными является число или сообщения по ошибке.

МАТЕМАТИЧЕСКАЯ МОДЕЛЬ

Для решения формулы подставим значения вместо x и произведем расчет.

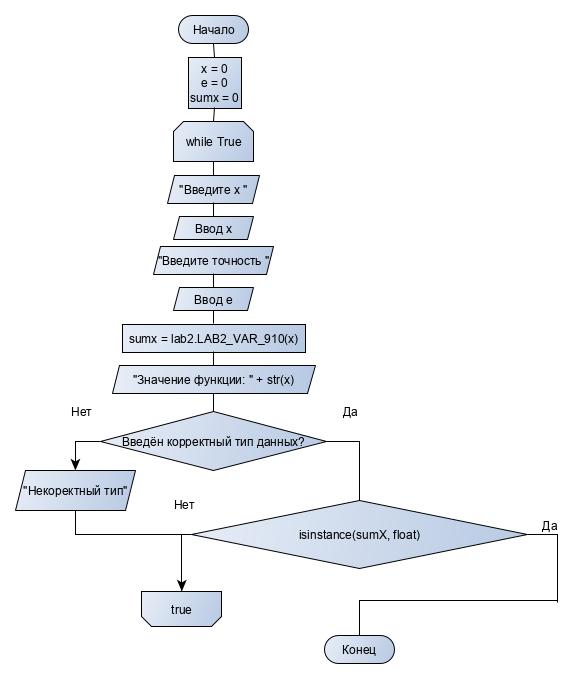
Пример ручного счёта, c точностью 0,002:

ОПИСАНИЕ РАЗРАБОТАННОЙ ПРОГРАММЫ

Таблица 1 - Описание переменных

|  |  |  |
| --- | --- | --- |
| Имя переменной | Тип переменной | Назначение |
| x | Int/Float | Запись начального значения x |
| e | Int/Float | Точность вычисления |
| sumx | Float | Конечное значение функции |
| n | Int | Номер очередного элемента ряда |
| i | Int | Копия n для вычисления факториала |
| q | Float | Хранение промежуточного члена ряда |

На вход подаётся 2 числа: x и e.

Рисунок 1 – функция main

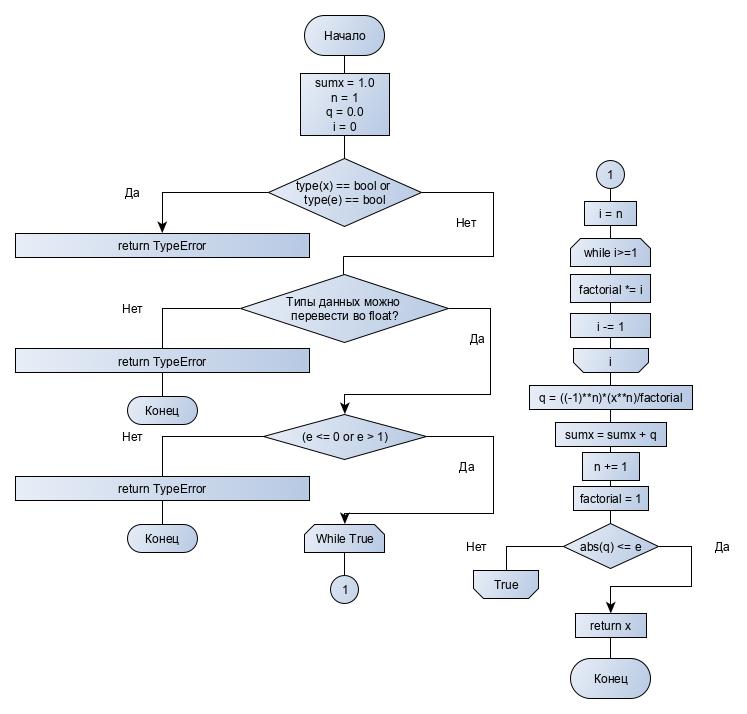


Рисунок 2 - функция LAB3\_VAR\_32

ОПИСАНИЕ ТЕСТОВОГО НАБОРА

Таблица 2 Тестовый набор

|  |  |  |  |
| --- | --- | --- | --- |
| Описание тестового случая | Входные данные | Ожидаемый результат | Результат теста |
| Передано целое число в x | x = 1, e = 0,00002 | 0.3678791887125221 | OK |
| Передано дробное число в x | x = 1.2, e = 0,00002 | 0.3011926747428572 | OK |
| Введен строковый тип данных в x | x = “var”, e = 0,00002 | TypeError | OK |
| Введен логический тип данных в x | x =True, e = 0,00002 | TypeError | OK |
| Введен массив в x | x= [1], e = 0,00002 | TypeError | OK |
| Введено дробное число e | x = 2, e = 0,00002 | 0.13533643311421098 | OK |
| Введено целочисленное число e | x = 5, e = 2 | "Недопустимое значение" | OK |
| Введена отрицательная e | x = 1, e = -0,00002 | "Недопустимое значение" | OK |
| Введена нулевая e | x = 1, e = 0 | "Недопустимое значение" | OK |
| Введен строковый тип данных в e | x = 1, e = “var” | TypeError | OK |
| Введен логический тип данных в e | x = 1, e = True | TypeError | OK |
| Введен массив в e | x = 1, e = [1] | TypeError | OK |

ПРИМЕР РАБОТЫ ПРОГРАММЫ

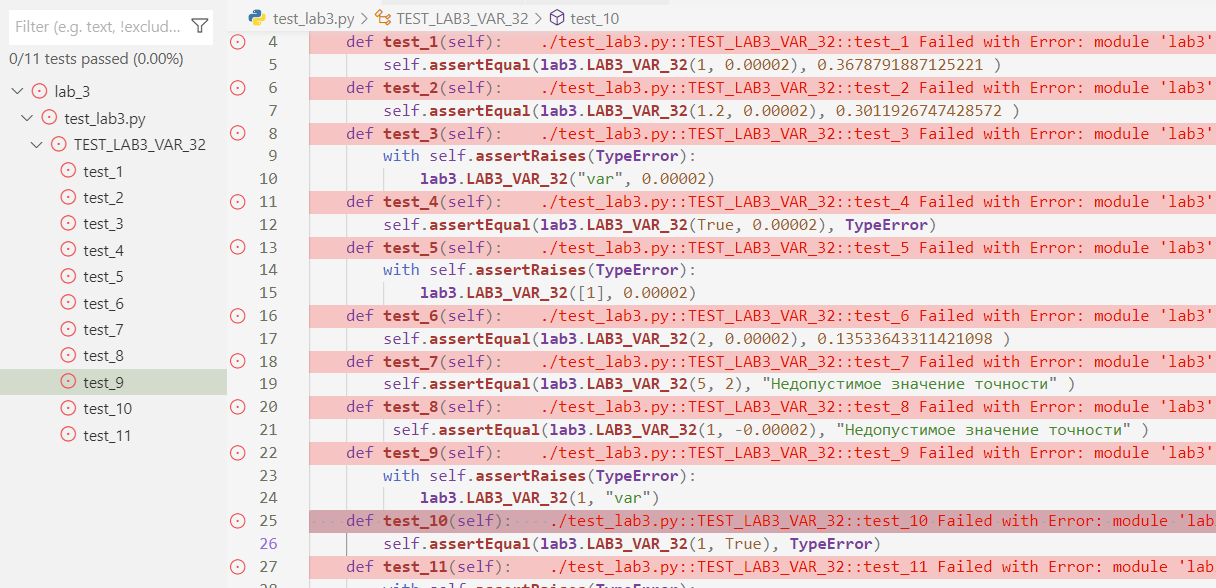


Рисунок 3 - Результат работы тестов до написания программы



Рисунок 4 - Результат работы тестов после написания программы

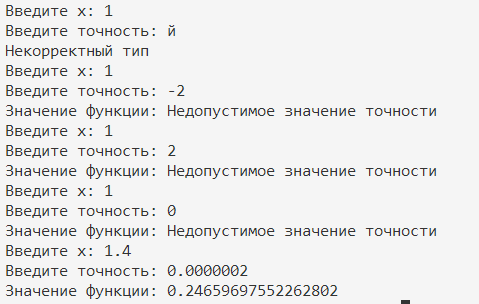


Рисунок 5 - Результат работы программы

ВЫВОД

Были улучшены навыки работы с python, изучены циклические конструкции, была выполнена поставленная задача. Была составлена программа успешно протестирована.

ПРИЛОЖЕНИЯ

Приложение 1

Код программы

Код тестов:

import **unittest**

import **lab3**

class **TEST\_LAB3\_VAR\_32**(**unittest**.**TestCase**):

    def **test\_1**(self):

        self.**assertEqual**(**lab3**.**LAB3\_VAR\_32**(1, 0.00002), 0.3678791887125221 )

    def **test\_2**(self):

        self.**assertEqual**(**lab3**.**LAB3\_VAR\_32**(1.2, 0.00002), 0.3011926747428572 )

    def **test\_3**(self):

        with self.**assertRaises**(**TypeError**):

**lab3**.**LAB3\_VAR\_32**("var", 0.00002)

    def **test\_4**(self):

        self.**assertEqual**(**lab3**.**LAB3\_VAR\_32**(True, 0.00002), **TypeError**)

    def **test\_5**(self):

        with self.**assertRaises**(**TypeError**):

**lab3**.**LAB3\_VAR\_32**([1], 0.00002)

    def **test\_6**(self):

        self.**assertEqual**(**lab3**.**LAB3\_VAR\_32**(2, 0.00002), 0.13533643311421098 )

    def **test\_7**(self):

        self.**assertEqual**(**lab3**.**LAB3\_VAR\_32**(5, 2), "Недопустимое значение точности" )

    def **test\_8**(self):

         self.**assertEqual**(**lab3**.**LAB3\_VAR\_32**(1, -0.00002), "Недопустимое значение точности" )

    def **test\_9**(self):

        with self.**assertRaises**(**TypeError**):

**lab3**.**LAB3\_VAR\_32**(1, "var")

    def **test\_10**(self):

        self.**assertEqual**(**lab3**.**LAB3\_VAR\_32**(1, True), **TypeError**)

    def **test\_11**(self):

        with self.**assertRaises**(**TypeError**):

**lab3**.**LAB3\_VAR\_32**(1, [1])

Код функции LAB3\_VAR\_32:

def **LAB3\_VAR\_32**(x, e):

*#иницилизация переменных*

    sumx = 1.0

    n = 1

    q = 0.0

    i = 0

    factorial = 1

*#Проверка, что введённые данные не являются логическим значением*

    if **type**(x) == **bool** or **type**(e) == **bool**:

        return **TypeError**

*#Попытка привестти введённые данные к типу float*

    try:

        x = **float**(x)

        e = **float**(e)

*#Вывод ошибки TypeError в случае неудачи*

    except:

        raise **TypeError**

*#Проверка на допустимые значения точности*

    if(e <= 0 or e > 1):

        return "Недопустимое значение точности"

*#Начало бесконечного цикла*

    while True:

*#Приравннивание i к n для последующего вычисления факториала*

        i = n

*#Цикл, в котром вычисляется факториал*

        while i >= 1: *#Условие продолжение цикла - i доложно быть больше либо равно 1*

            factorial \*= i *#Вычисление очередного значения факториала*

            i -= 1 *#Уменьшение i для перехода к следующему этапу*

*#Вычисление значения очередного члена редя*

        q = ((-1)\*\*n)\*(x\*\*n)/factorial

*#Вычисление значения функции*

        sumx = sumx + q

*#Переход к следующему члену ряда*

        n += 1

*#Сброс значения факториала для последующего вычисления*

        factorial = 1

*#Если абсолютне значение ряда меньше либо равно точности, то завершить цикл*

        if(**abs**(q) <= e):

            break

*#Вернуть значение функции*

    return sumx;

Код функции main:

import **lab3**

*#Иницилизация переменных*

x = 0

e = 0

sumx = 0

*#Начало бесконечного цикла*

while True:

*#Начало блока обработки исключений*

    try:

        x = **input**("Введите x: ")*#Ввод х*

        e = **input**("Введите точность: ")*#Ввод точности*

        sumx = **lab3**.**LAB3\_VAR\_32**(x, e)*#Вызов функции и запись результата её работы в переменную*

**print**("Значение функции: " + **str**(sumx))*#Вывод в консоль результата*

*#При вызове исключения*

    except **TypeError**:

**print**("Некорректный тип")*#Вывод в консоль сообщния об ошибке*

*#Выход в случае успешного выполнения*

    if **isinstance**(sumx, **float**):

        break

*#Конец бесконечного цикла*