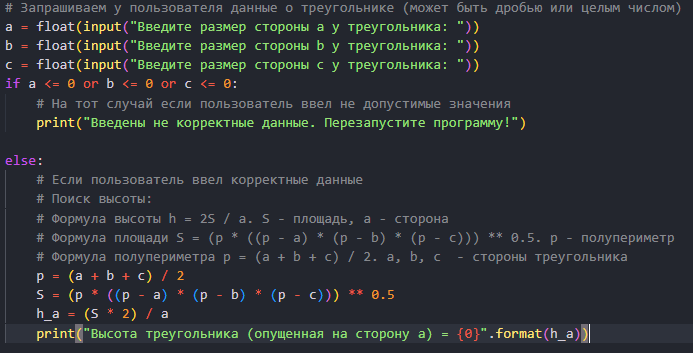
**1 Задание.**

Вычислить высоту треугольника, опущенную на сторону *a*, по известным значениям длин его сторон *a, b, c*.



Листинг рабочей программы «main.py»

Таблица 1

|  |  |
| --- | --- |
| Данные: | |
| Входные (ввод) | Выходные (вывод) |
| 5  5  6 | Высота треугольника (опущенная на сторону a) = 4.8 |
| 3  4  5 | Высота треугольника (опущенная на сторону a) = 4.0 |



Блок схема 1

**2 задание:**

Составить блок-схему и программу вычисления значения функции.

def fun(x):

    if x < -4.5:

        r1 = (9 \* (x \*\* 2)) + 5

        r2 = (3 \* x) + 12

        return r1 / r2

    elif x >= -4:

        return (4 \* (x \*\* 2)) - 7

    else:

        return "Введено число в диапозоне [-4.5; -4)"

def main():

    x = int(input("Введите значение x: "))

    print(f"Результат: {fun(x)}")

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

    main()

Листинг рабочей программы «main2.py»

Таблица 2

|  |  |
| --- | --- |
| Данные: | |
| Входные (ввод) | Выходные (вывод) |
| 3 | Результат: 29 |

Блок схема 2.1



Блок схема 2.2

**3 задание:**

Вычислить сумму ряда с точностью до 10-4.

def fun(n):

    r1 = (-1) \*\* n

    r2 = 2 \*\* n

    return r1 / r2

def main():

    n = 1

    e = 0.0001

    pos\_sum = 0

    while True:

        res = fun(n)

        if abs(res) < e:

            break

        pos\_sum += res

        n += 1

    print(f"Сумма ряда {pos\_sum}")

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

    main()

Листинг рабочей программы «main3.py»

Таблица 3

|  |  |
| --- | --- |
| Данные: | |
| Входные (ввод) | Выходные (вывод) |
| Константы | Сумма ряда -0.3333740234375 |



Блок схема 3.1

**4 Задание.**

Составить блок-схему и программу для запыления таблицы значений функции y = f(x) на отрезке с указанным шагом измерения аргумента. Значение функции выводить с точностью до тысячных долей.

import math

print("-" \* 60)

print("!{: ^28}!".format('X'), "{: ^28}!".format('y = f(x)'))

print("-" \* 60)

for i in range(-500, 1000, 55):

    i /= 100

    if i < -1:

        print(

            "!{: ^28}!".format(i),

            "{: ^29}!".format(

                "{: .3f}".format(i \*\* 2)

            )

        )

    if 0 <= i <= 12.5:

        print(

            "!{: ^28}!".format(i),

            "{: ^29}!".format(

                "{: .3f}".format((math.e \*\* i) + 5 + math.cos(0.001 \* i))

            )

        )

Листинг рабочей программы «main4.py»

Таблица 4

|  |  |
| --- | --- |
| Данные: | |
| Входные (ввод) | Выходные (вывод) |
| Константы |  |

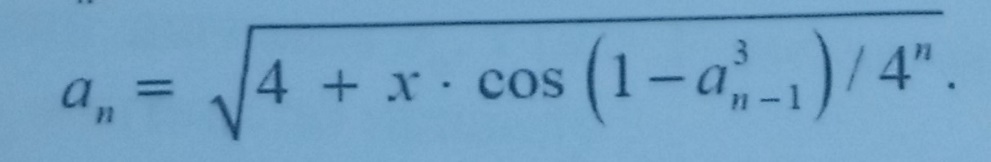


Блок схема 4

**5 Задание.**

Вычислить предел последовательности an с точностью 0.5 \* 10-6, если значения a0,x (x > 0) вводятся.

Значения an считать по правилу



import math

x = float(input('Введите x: '))

if x <= 1:

    print("Были введены не верные данные")

else:

    n = 0 # Количество итераций цикла

    # Точность предела последовательности

    e =  0.0000005

    # Прошлый элемент последовательности

    a\_past = float(input('Введите a0: '))

    while True:

        n += 1

        # Новый элемент последовательности

        a\_next = 1 + math.sin(1 + (a\_past \*\* 2)) / x \*\* n

        if abs(a\_past - a\_next) < e:

            print(f"Найденный предел последовательности: {a\_next}")

            break

        a\_past = a\_next

Листинг рабочей программы «main5.py»

Таблица 5

|  |  |
| --- | --- |
| Данные: | |
| Входные (ввод) | Выходные (вывод) |
| 2  5 | Найденный предел последовательности: 1.0000004335864567 |
| 4  9 | Найденный предел последовательности: 1.0000000541983394 |



Блок схема 5

**6 Задание.**

Дано натуральное число. Вычислить количество цифр, кратных 4.

number = input("Введите число: ")

flag = 0

for i in number:

    if int(i) % 4 == 0:

        flag += 1

print(f"Количество цифр кратных 4: {flag}")

Листинг рабочей программы «main6.py»

Таблица 6

|  |  |
| --- | --- |
| Данные: | |
| Входные (ввод) | Выходные (вывод) |
| 1234567890 | Количество цифр кратных 4: 3 |



Блок схема 6

**7 Задание.**

Вводится последовательность из N целых положительных элементов. Посчитать количество чисел палиндромов. Для определения паллиндрома создать функцию

def found\_pall(number:str) -> bool:

    return number == number[::-1]

def main():

    k = 0

    for i in range(int(input("Введите количество элементов в последовательности: "))):

        number = input("Введите значение элемента: ")

        if found\_pall(number):

            k += 1

    print(f"Количество палиндромов: {k}")

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

    main()

Листинг рабочей программы «main7.py»

Таблица 7

|  |  |
| --- | --- |
| Данные: | |
| Входные (ввод) | Выходные (вывод) |
| 3  101  111  121 | Количество палиндромов: 3 |
| 3  265  22  131 | Количество палиндромов: 2 |



Блок схема 7

**8 Задание.**

Составить программу, которая считывает сначала количество оценок, потом по очереди сами эти оценки, затем выводит их же в том же порядке (используем список). Найдите среднюю оценку за урок.

def main():

    count\_number = int(input("Введите количество цифр: "))

    sp = [] # список для оценок

    for i in range(count\_number):

        sp.append(int(input("Введите значение оценки: ")))

    print(f"Среднее арифметическое {round(sum(sp) / len(sp), 3)}")

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

    main()

Листинг рабочей программы «main8.py»

Таблица 8

|  |  |
| --- | --- |
| Данные: | |
| Входные (ввод) | Выходные (вывод) |
| 3  2  2  3 | Среднее арифметическое 2.333 |



Блок схема 8

**9 Задание.**

Для списка X, состоящего из 15 элементов, найти наибольший элемент списка и его порядковый номер

def main():

    sp = []

    for i in range(15):

        sp.append(float(input("Введите значение числа: ")))

    print(f"Максимальный элемент: {max(sp)}")

    print(f"Номер максимального элемента: {sp.index(max(sp)) + 1}")

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

    main()

Листинг рабочей программы «main9.py»

Таблица 9

|  |  |
| --- | --- |
| Данные: | |
| Входные (ввод) | Выходные (вывод) |
| 1  1  1  1  1  1  1  1  2  3  3  4  4  5  66 | Максимальный элемент: 66.0  Номер максимального элемента: 15 |



Блок схема 9

**10 Задание.**

В списке из 12 вещественных чисел найти наименьший элемент и поменять его местами с первым элементом.

def main():

    sp = []

    for i in range(12):

        sp.append(float(input("Введите значение числа: ")))

    print(f"Первоначальный список: {sp}")

    ind = sp.index(min(sp))

    sp[0], sp[ind] = sp[ind], sp[0]

    print(f"Измененый список: {sp}")

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

    main()

Листинг рабочей программы «main10.py»

Таблица 10

|  |  |
| --- | --- |
| Данные: | |
| Входные (ввод) | Выходные (вывод) |
| 3456  34  5623  456  3456  23456  0  3456  3245  234  3  43 | Первоначальный список: [3456.0, 34.0, 5623.0, 456.0, 3456.0, 23456.0, 0.0, 3456.0, 3245.0, 234.0, 3.0, 43.0]  Измененый список: [0.0, 34.0, 5623.0, 456.0, 3456.0, 23456.0, 3456.0, 3456.0, 3245.0, 234.0, 3.0, 43.0] |



Блок схема 10

**11 Задание.**

Записать элементы массива X, удовлетворяющие условию 1<= x <= 2, подряд в массив Y. Поменять местами максимальный и минимальный элементы в массиве.

def main():

    '''Функция main'''

    x = []

    while True:

        number = input("Введите число (break - остановка): ").rstrip('\n')

        if number == "break":

            break

        x.append(float(number))

    print(f"Полученный список {x}")

    Y = []

    for i in x:

        if 1 <= i <= 2:

            Y.append(i)

    if len(Y) <= 1:

        print(f"Полученный список c элементами с из диапазона [1, 2]: {Y}")

        return

    in\_max = Y.index(max(Y))

    in\_min = Y.index(min(Y))

    Y[in\_min], Y[in\_max] = max(Y), min(Y)

    print(f"Полученный список c элементами с из диапазона [1, 2]: {Y}")

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

    main()

Листинг рабочей программы «main11.py»

Таблица 11

|  |  |
| --- | --- |
| Данные: | |
| Входные (ввод) | Выходные (вывод) |
| 1  2  3  1.2  1.4  5  6  7  1.5  1.9  1.0  1.7  break | Полученный список [1.0, 2.0, 3.0, 1.2, 1.4, 5.0, 6.0, 7.0, 1.5, 1.9, 1.0, 1.7]  Полученный список c элементами с из диапазона [1, 2]: [2.0, 1.0, 1.2, 1.4, 1.5, 1.9, 1.0, 1.7] |



Блок схема 11

**12 Задание.**

Удалить из строки все запятые

def main():

    '''Функция main'''

    while True:

        st = input("Введите строку (break для завершения ввода): ")

        if st == "break":

            return

        print(f"Строка до преобразований: {st}")

        st = st.replace(",", " ")

        print(f"Строка после преобразований: {st}")

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

    main()

Листинг рабочей программы «main12.py»

Таблица 12

|  |  |
| --- | --- |
| Данные: | |
| Входные (ввод) | Выходные (вывод) |
| ав, ав, ыв  break | Строка до преобразований: ав, ав, ыв  Строка после преобразований: ав ав ыв |



Блок схема 12

**13 задание:**

Задана матрица A(n, m)б в каждом столбце которой максимальный элемент необходимо заменить произведением отрицательных элементов этого же столбца.

def made\_mtr(st, col):

    A = []

    for i in range(st):

        sp = []

        for ii in range(col):

            sp.append(int(input(f"Введите значение элемента с номером {i + 1, ii + 1}: ")))

        A.append(sp)

    return A

def mtr\_command(A):

    for i in range(len(A[0])):

        sp = []

        for ii in range(len(A)):

            sp.append(A[ii][i])

        ind\_max = sp.index(max(sp))

        res = 1

        for g in sp:

            if g < 0:  res \*= g

        A[ind\_max][i] = res

def \_\_str(A):

    arr = []

    for i in range(len(A)):

        arr.append(" ".join([str(ii) for ii in A[i]]))

    arr = '\n'.join(arr)

    return arr

def main():

    st = int(input("Введите количество строк матрицы: "))

    col = int(input("Введите количество колонок матрицы: "))

    A = made\_mtr(st, col)

    print(f"Введеная матрица:\n{\_\_str(A)}")

    mtr\_command(A)

    print(f"Обработанная матрица:\n{\_\_str(A)}")

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

    main()

Листинг рабочей программы «main13.py»

Таблица 13

|  |  |
| --- | --- |
| Данные: | |
| Входные (ввод) | Выходные (вывод) |
| 3  2  1  2  -4  -3  -7  -8 | Введеная матрица:  1 2  -4 -3  -7 -8  Обработанная матрица:  28 24  -4 -3  -7 -8 |



Блок схема 13.1



Блок схема 13.2



Блок схема 13.3

**14 Задание.**

Создать типизированный файл, куда записать n целых чисел. Из компонентов исходного файла сформировать массивы, из чисел, больших 10 и меньших двух. Вычислить количество нулевых компонентов файла.

import sys, os

def wright\_file() -> None:

    file = open("test.txt", "w")

    sp = []

    while True:

        number = input("Введите число (break - остановка ввода): ")

        if number == "break":

            break

        sp.append(number)

    file.write("\n".join(sp))

    file.close()

def read\_file() -> None:

    file = open("test.txt", "r")

    sp = [int(i) for i in file.readlines()]

    file.close()

    big\_numbers, low\_number = [], []

    cout\_sero\_number = 0

    for i in sp:

        if i > 10:

            big\_numbers.append(i)

        elif i == 0:

            cout\_sero\_number += 1

        elif i < 2:

            low\_number.append(i)

    print(f"Числа больше 10: {big\_numbers}")

    print(f"Числа меньше 2: {low\_number}")

    print(f"Числа равные 0: {cout\_sero\_number}")

# Особенноесть среды разработки

PROGRAM\_DIR = "\\".join(sys.argv[0].split("\\")[:-1]) # Папка программы

os.chdir(PROGRAM\_DIR) # Переход в папку программы

def main():

    wright\_file()

    read\_file()

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

    main()

Листинг рабочей программы «main14.py»

Таблица 14

|  |  |
| --- | --- |
| Данные: | |
| Входные (ввод) | Выходные (вывод) |
| 1  2  3  4  5  6  7  8  9  21  34  23  0  0  234  54  12  666  4  break | Числа больше 10: [21, 34, 23, 234, 54, 12, 666]  Числа меньше 2: [1]  Числа равные 0: 2 |



Блок схема 14.1



Блок схема 14.2



Блок схема 14.3