Министерство науки и высшего образования РФ

ФГАОУ ВО Пермский национальный исследовательский  
политехнический университет

Кафедра «Вычислительная математика, механика и биомеханика»

Отчёт по лабораторной работе № 2  
тема «Линейные алгоритмы»  
по дисциплине «Информатика»

Выполнил: студент группы ПМ-23-1б Кузнецов Д.В.

Проверил: ст. пр. каф. ВММБ Ильиных Г.В.

Пермь, 2023

**Задание 1**

* 1. ***Постановка задачи***

Написать функцию, которая определяет, оканчивается ли число на чётную цифру. Написать функцию которая определяет, является ли введённое число больше заданного. Создать программу, которая в бесконечном цикле получает от пользователя числа и проверяет их с помощью двух функций.

* 1. ***Алгоритм выполнения***

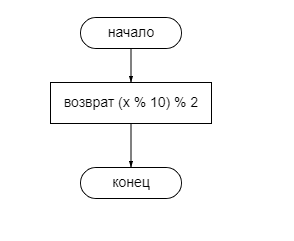
**

Рис.1 – Блок-схема *func1(x)*

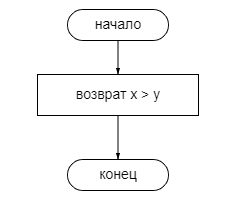


Рис.2 – Блок-схема *func2(x,y)*

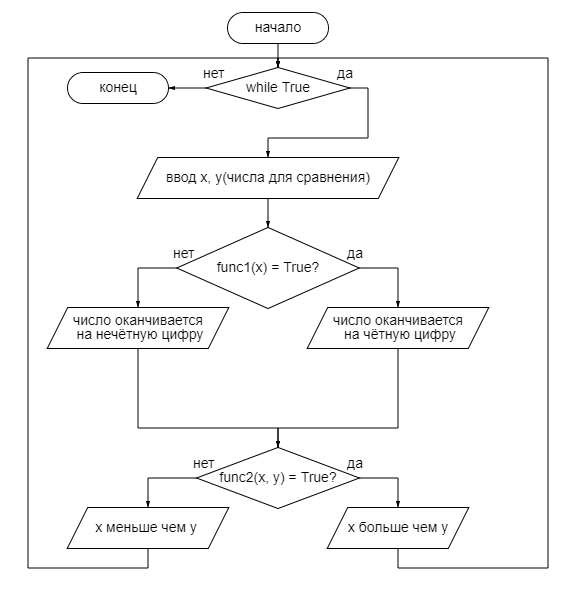


Рис.3 – Блок-схема программы №1

* 1. ***Код программы***

def task1():  
 def func1(x):  
 return (x % 10) % 2 == 0:  
  
 def func2(x, y):  
 return x > y  
  
 x = int(input("Введите число "))  
 y = int(input("Введите число для сравнения "))  
 if func1(x):  
 print("Число оканчивается на чётную цифру")  
 else:  
 print("Число оканчивается на нечётную цифру")  
 if func2(x, y):  
 print(f'{x} больше чем {y}')  
 else:  
 print(f'{x} меньше чем {y}'))

* 1. ***Тестирование работы программы с проверкой***

Для проверки решений в MS Excel была создана таблица:  
В ячейке A1 было записано заданное значение(x), в ячейке B1 было записано  
число для сравнения(y). В ячейку С1 была записана func1, в ячейку D1 func2  
Ячейки C1 и D1 были заданы формулами:  
C1 - =ЕСЛИ(ОСТАТ(ОСТАТ(A1;10);2) = 0;"правда"; "ложь")  
D1 - =ЕСЛИ(A1 > B1; "правда"; "ложь")

Таблица 1 – Тестирование работы и проверка решений программы №1

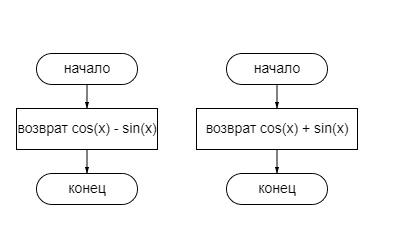
|  |  |  |
| --- | --- | --- |
| №п.п | Решение Python | Решение MS Excel |
| 1 | Введите число **1234**  Введите число для сравнения **4321**  Число оканчивается на чётную цифру 1234 меньше чем 4321 | |  |  |  |  | | --- | --- | --- | --- | | x | y | func1 | func2 | | 1234 | 4321 | правда | ложь | |
| 2 | Введите число **999**  Введите число для сравнения **51083**  Число оканчивается на нечётную цифру  999 меньше чем 51083 | |  |  |  |  | | --- | --- | --- | --- | | x | y | func1 | func2 | | 999 | 51083 | ложь | ложь | |
| 3 | Введите число **32132**  Введите число для сравнения **32123**  Число оканчивается на чётную цифру  32132 больше чем 32123 | |  |  |  |  | | --- | --- | --- | --- | | x | y | func1 | func2 | | 32132 | 32123 | правда | правда | |

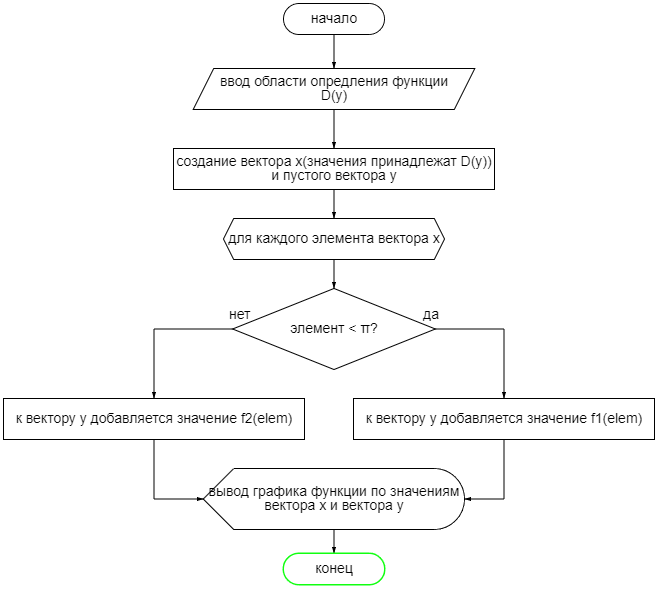
**Задание 2**

***2.1 Постановка задачи***

По формуле вычислить значения функции f(x) в диапазоне значений   
x ∈ [a;b] и представить в виде графика:

***2.2 Алгоритм решения***

Рис.4- Блок-схема *f1(x)* и *f2(x)*

  
Рис.5 – Блок-схема программы №2

***2.3 Код программы***

import matplotlib.pyplot as plt  
import numpy as np  
  
  
def task2():  
 def f1(x):  
 return np.cos(x) - np.sin(x)  
  
 def f2(x):  
 return np.cos(x) + np.sin(x)  
 d\_y = (int(input("введите область определения функции \n")), int(input()))  
 vector\_x = np.linspace(d\_y[0], d\_y[1], 200)  
 vector\_y = []  
 for elem in vector\_x:  
 if elem < np.pi:  
 vector\_y.append(f1(elem))  
 else:  
 vector\_y.append(f2(elem))  
 plt.plot(vector\_x, vector\_y, color='r')  
 plt.grid()  
 plt.show()

***2.4 Тестирование работы программы с проверкой***

Для проверки построения графика используется сайт Wolfram Alpha.

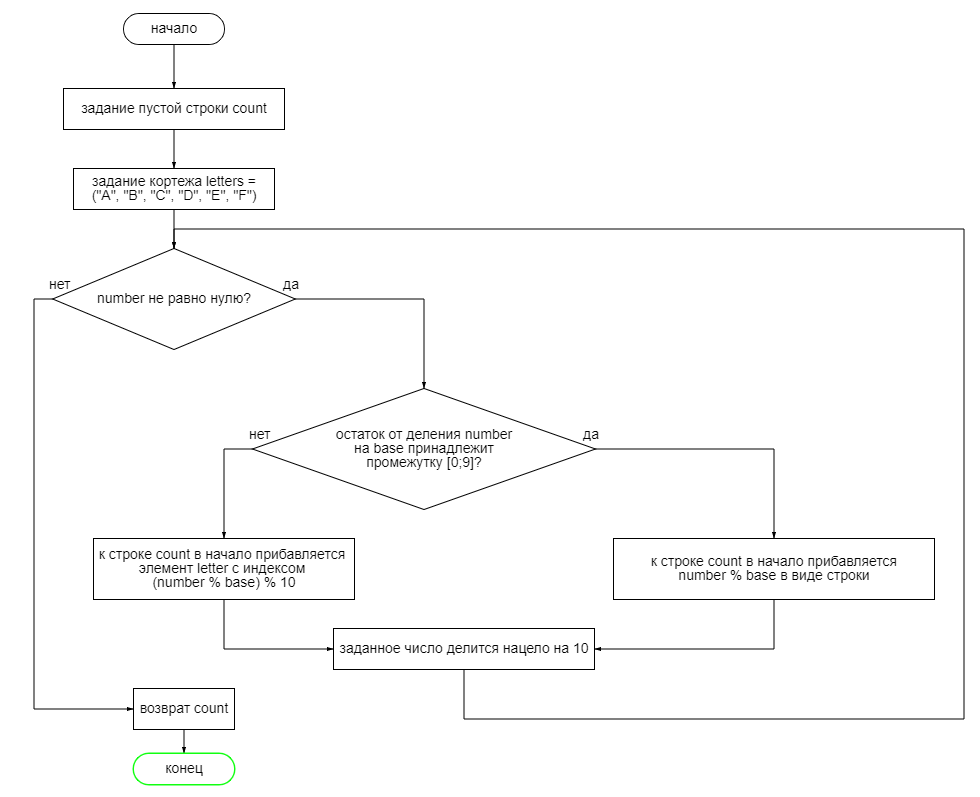
Таблица 2 - Тестирование работы и проверка решений программы №2

|  |  |  |
| --- | --- | --- |
| № | Решение Python | График Wolfram Alpha |
| 1 | введите область определения функции **1 10** |  |
| 2 | введите область определения функции  **-10 0** |  |
| 3 | введите область определения функции **200 210** |  |

**Задание 3  
*3.1 Постановка задачи***

Написать функцию *number\_in\_new\_numeral\_system(number, base)*, которая переводит целое десятичное число в СС с выбранным основанием. Написать программу, которая переводит заданное пользователем значение в заданную пользователем СС.

***3.2 Алгоритм решения***

Рис.6 – Блок-схема функции *number\_in\_new\_numeral\_system(number, base)*

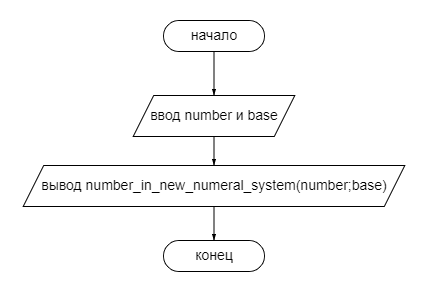


Рис.7 – Блок-схема программы №3

***3.3 Код программы***

def task3():  
  
 def number\_in\_new\_numeral\_system(number, base):  
 count = ''  
 letters = ["A", "B", "C", "D", "E", "F"]  
 while number != 0:  
 if 0 <= number % base <= 9:  
 count = str(number % base) + count  
 else:  
 count = letters[(number % base) % 10] + count  
  
 number //= base  
 return count  
  
 user\_count = int(input("Введите число в 10-чной СС "))  
 cs\_base = int(input("Введите основание новой СС "))  
  
 print(number\_in\_new\_numeral\_system(user\_count, cs\_base))

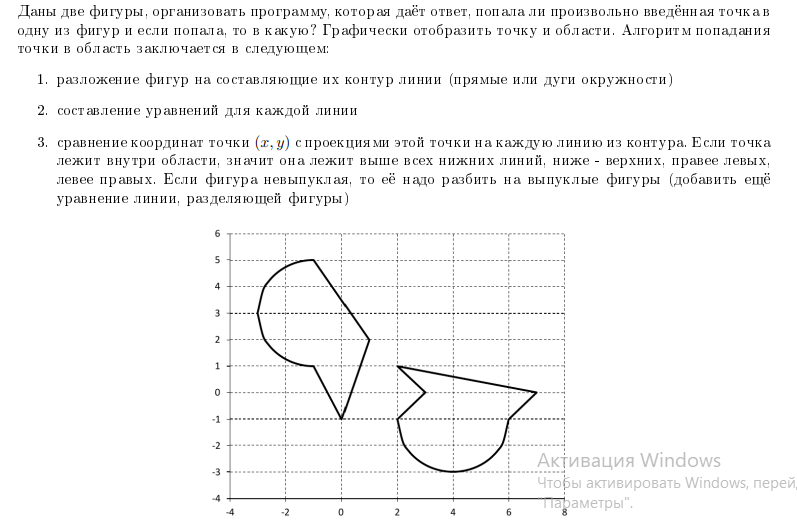
***3.4 Тестирование работы программы с проверкой***

Для проверки решений используется онлайн-калькулятор систем счисления.

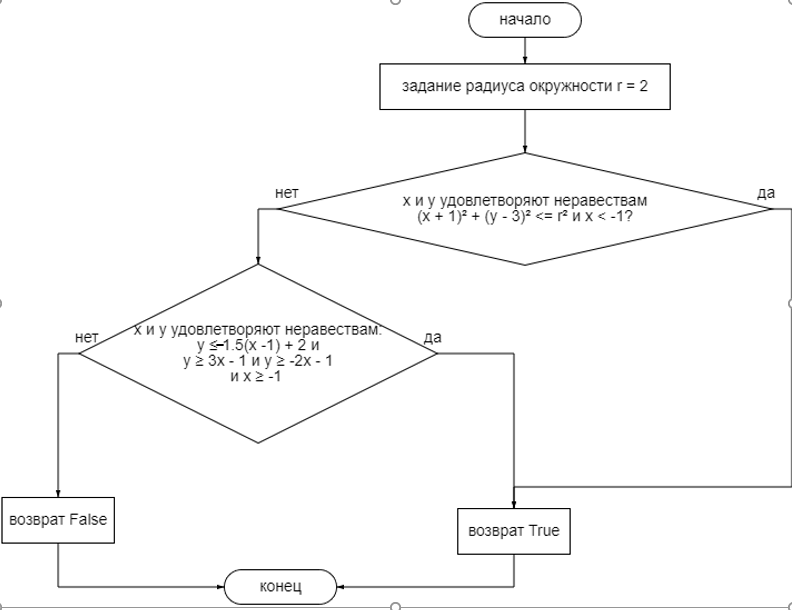
Таблица 3 - Тестирование работы и проверка решений программы №3

|  |  |  |
| --- | --- | --- |
| №п.п | Решение Python | Решение онлайн-калькулятора |
| 1 | Введите число в 10-чной СС **322**  Введите основание новой СС **13**  1BA |  |
| 2 | Введите число в 10-чной СС **5243**  Введите основание новой СС **6**  40135 |  |
| 3 | Введите число в 10-чной СС **729**  Введите основание новой СС **3**  1000000 |  |

**Задание 4**

***4.1 Постановка задачи***

***4.2 Алгоритм решения***

  
Рис.9 – Блок-схема функции *check1(x, y)*

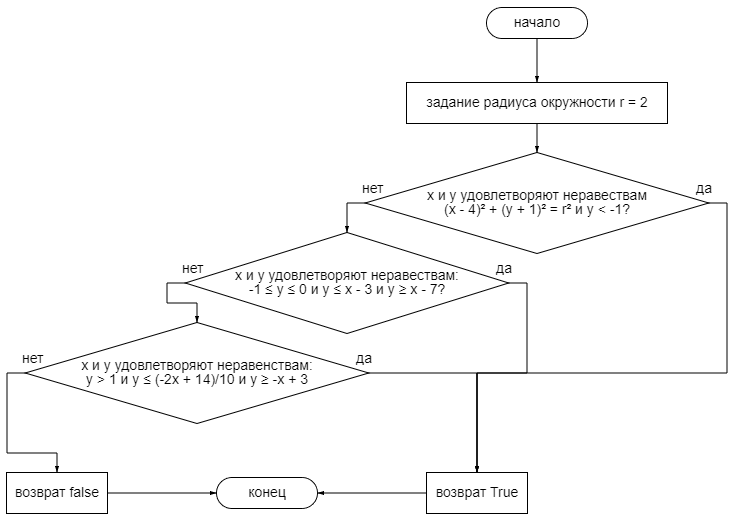


Рис.10 – Блок-схема *check2(x,y)*

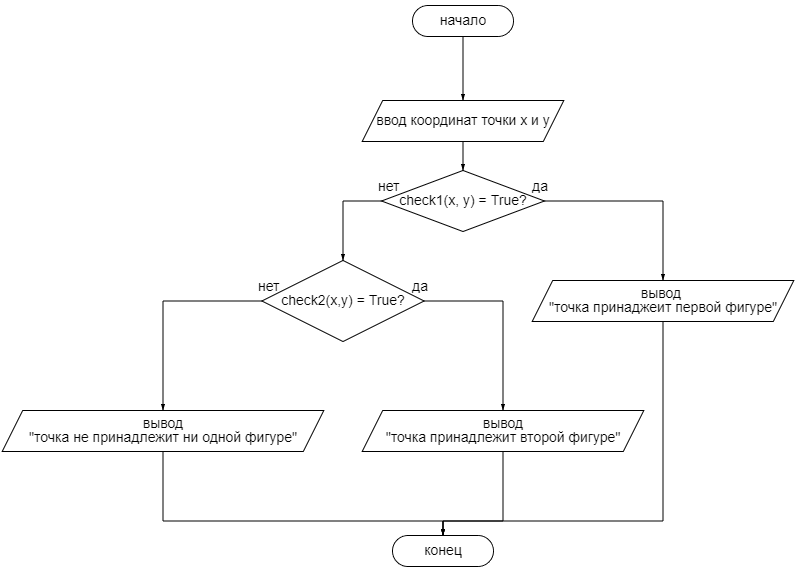


Рис.11 – Блок-схема программы №4

***4.3 Код программы***

def task4():  
 theta = np.linspace(np.pi / 2, 3 \* np.pi / 2, 150)  
 radius = 2  
 a1 = radius \* np.cos(theta) - 1  
 b1 = radius \* np.sin(theta) + 3  
 x1 = [-1, 1, 0, -1]  
 y1 = [5, 2, -1, 1]  
 axes = plt.subplot()  
 axes.plot(a1, b1, color='black')  
 axes.plot(x1, y1, color='black')  
 axes.set\_aspect(1)  
 theta2 = np.linspace(np.pi, 2 \* np.pi, 150)  
 a2 = radius \* np.cos(theta2) + 4  
 b2 = radius \* np.sin(theta2) - 1  
 x2 = [2, 3, 2, 7, 6]  
 y2 = [-1, 0, 1, 0, -1]  
 axes.plot(x2, y2, color='black')  
 axes.plot(a2, b2, color='black')  
 plt.grid()  
 plt.title('Task 4')  
  
 def check1(x, y):r = 2  
 if (x + 1) \*\* 2 + (y - 3) \*\* 2 <= r \*\* 2 and x < -1:  
 return True  
 elif x >= -1 and -1.5 \* (x - 1) + 2 >= y >= 3 \* x - 1 and y >= -2 \* x - 1:  
 return True  
 else:  
 return False  
  
 def check2(x, y):r = 2  
 if y < -1 and (x - 4) \*\* 2 + (y + 1) \*\* 2 == r \*\* 2:  
 return True  
 # упрощенное условие -1 <= y <= 0 and y <= x - 3 and y >= x - 7  
 elif -1 <= y <= x - 3 and x - 7 <= y <= 0:  
 return True  
 # упрощенное условие y >= 0 and y >= -x + 3 and y <= -0.2 \* x + 1.4  
 elif 0 < y <= (-2 \* x + 14) / 10 and y >= -x + 3:  
 return True  
 else:  
 return False  
  
 x = float(input("введите координату x точки "))  
 y = float(input("введите координату y точки "))  
 if check1(x, y):  
 print("Точка принадлежит первой фигуре")  
 elif check2(x, y):  
 print("Точка принадлежит второй фигуре")  
 else:  
 print("Точка не принадлежит ни одной из фигур")  
 axes.scatter(x, y)  
 plt.show()

***4.4 Тестирование работы программы с проверкой***

Для проверки решений используется MS Excel. В ячейки A3 и B3 записываются произвольные значения x и y, в ячейку A1 записан r = 2 = const. Ячейки C3, D3 проверяют принадлежность первой фигуре, а ячейки E3, F3, G3 принадлежность второй. Данные ячейки заданы формулами:

С3 =ЕСЛИ(И(СТЕПЕНЬ(A3 + 1;2) + СТЕПЕНЬ(B3 - 3;2) <= СТЕПЕНЬ($A$1;2);A3 < -1); "True";"False")  
D3 =ЕСЛИ(И(A3 >= -1;B3 <= -1,5 \* (A3 - 1) + 2; B3 >= 3 \* A3 - 1; B3 >= -2 \* A3 - 1); "True"; "False")  
E3 =ЕСЛИ(И(СТЕПЕНЬ(A3 - 4;2) + СТЕПЕНЬ(B3 + 1;2) <= СТЕПЕНЬ($A$1;2);B3 < -1); "True";"False")  
F3 =ЕСЛИ(И( -1 <= B4; B4 <= A4 - 3; B4 >= A4 - 7; B4 <= 0); "True";"False")  
G3 =ЕСЛИ(И(B4 > 0; B4 <= -0,2 \* A4 + 1,4; B4 >= -A4 + 3); "True";"False")

Ячейка H3 определяет какой фигуре принадлежит точка(0 – никакой, 1 – первой, 2 – второй) и задана формулой:

H3 =ЕСЛИ(ИЛИ(C3 = "True"; D3 = "True"); "1"; ЕСЛИ(ИЛИ(E3 = "True"; F3 = "True"; G3 = "True");"2";"0"))

Таблица 4 - Тестирование работы и проверка решений программы №4

|  |  |  |
| --- | --- | --- |
| №п.п | Решение Python | Решение MS Excel |
| 1 | введите координату x точки **-2** введите координату y точки **2** Точка принадлежит первой фигуре | |  |  |  | | --- | --- | --- | | x | y | принадлежность | | -2 | 2 | 1 | |
| 2 | введите координату x точки **4** введите координату y точки **5** Точка не принадлежит ни одной из фигур | |  |  |  | | --- | --- | --- | | x | y | принадлежность | | 4 | 5 | 0 | |
| 3 | введите координату x точки **5** введите координату y точки -**2** Точка принадлежит второй фигуре | |  |  |  | | --- | --- | --- | | x | y | принадлежность | | 5 | -2 | 2 | |

**Задание 5**

***5.1 Постановка задачи***

Написать программу, которая находит наименьшее и наибольшее абсолютное значение среди трёх заданных чисел.

***5.2 Алгоритм решения***

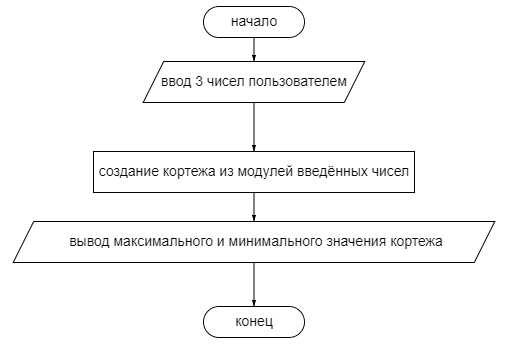


Рис.12 – Блок-схема программы №5

***5.3 Код программы***

def task5():  
 ans = (abs(int(input("Введите число ")) for x in range(3)))  
 print(f"Максимальное значение равно {max(ans)}", f"Минимальное значение равно {min(ans)}")

***5.4 Тестирование работы программы с проверкой***

Для проверки решения в MS Excel была создана таблица. A9, B9 и C9 используются для заданных чисел. Ячейки D9, E9, F9 являются модулями A9, B9 и C9 соответственно. Ячейки G9 и H9 заданы формулами:

G9 =МАКС(D9:F9)

H9 =МИН(D9:F9)

Таблица 5 - Тестирование работы и проверка решений программы №5

|  |  |  |
| --- | --- | --- |
|  | Решение Python | Решение Excel |
| 1 | Введите число -**48**  Введите число **7655**  Введите число -**213235**  Максимальное абсолютное значение равно 213235 Минимальное абсолютное значение равно 48 | |  |  |  |  |  | | --- | --- | --- | --- | --- | | a | b | c | Max.  abs | Min.  abs | | -48 | 7655 | -213245 | 213245 | 48 | |
| 2 | Введите число **321**  Введите число **123**  Введите число -**1245125**  Максимальное абсолютное значение равно 1245125 Минимальное абсолютное значение равно 123 | |  |  |  |  |  | | --- | --- | --- | --- | --- | | a | b | c | Max.  abs | Min.  abs | | 321 | 123 | -1245125 | 1245125 | 123 | |
| 3 | Введите число -**12412**  Введите число -**13214**  Введите число -**12341**  Максимальное абсолютное значение равно 13214 Минимальное значение равно 12341 | |  |  |  |  |  | | --- | --- | --- | --- | --- | | a | b | c | Max.  abs | Min.  abs | | -12412 | -13214 | -12341 | 13214 | 12341 | |

**Задание 6**

***6.1 Постановка задачи***

Известно, что каждую неделю количество особей увеличивается на 10% от общего количества. Причём, при достижении критического значения(*critical\_number* особей), количество особей уменьшается на треть. Написать программу определяющую количество особей через неделю. Изначально имеется *start\_number* особей. Написать программу, в которой с помощью *species\_number* определяется количество месяцев, через которое популяция превысит заданное значение *target\_number*.

***6.2 Алгоритм решения***

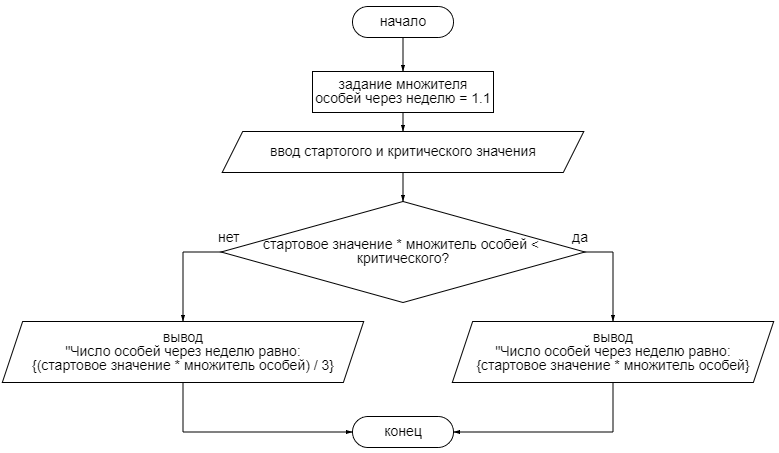


Рис.13 – Блок-схема первой части программы №6

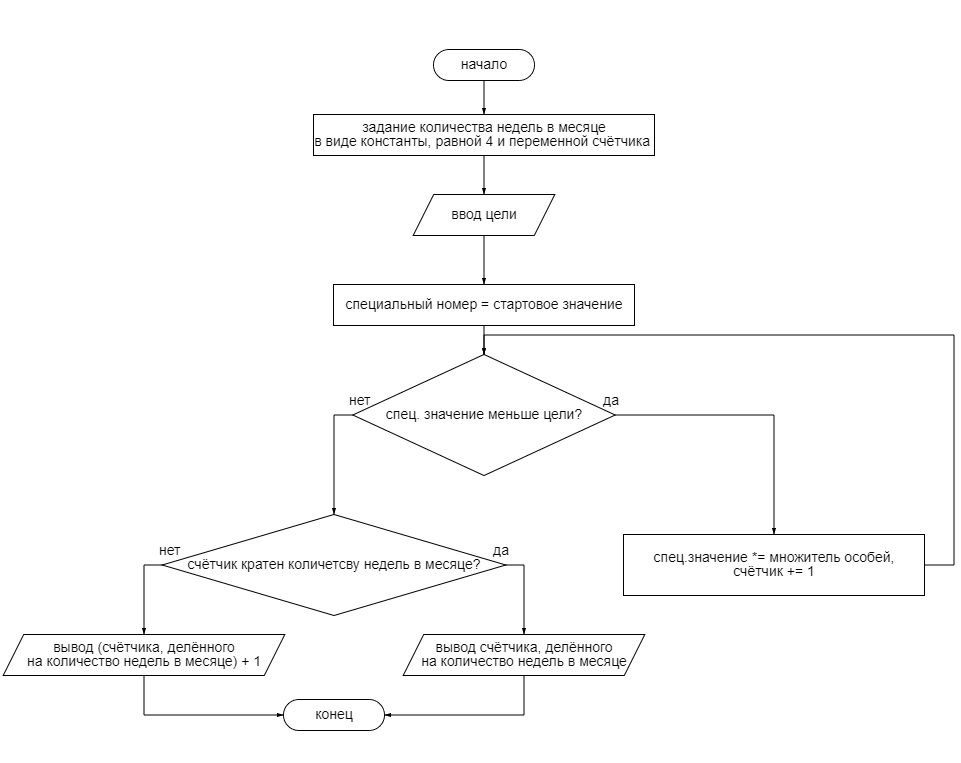


Рис.14 – Блок-схема второй части программы №6

***6.3 Код программы***

def task6():  
 INCREASE\_PER\_WEEK = 1.1  
 start\_number = float(input("Введите начальное количество особей "))  
 critical\_number = float(input("Введите критическое значение "))  
 if start\_number \* INCREASE\_PER\_WEEK < critical\_number:  
 print(f"Число особей через неделю равно {start\_number \* INCREASE\_PER\_WEEK}")  
 else:  
 print(f"Число особей через неделю равно", "{0:.4f}".format(start\_number \* INCREASE\_PER\_WEEK / 3))

WEEK\_PER\_MONTH = 4  
 count = 0  
 target\_number = float(input("Введите цель "))  
 species\_number = start\_number  
 while species\_number < target\_number:  
 species\_number \*= INCREASE\_PER\_WEEK  
 count += 1  
 if count % WEEK\_PER\_MONTH == 0:  
 print(f"Количество месяцев для достижения цели равно {count // WEEK\_PER\_MONTH}")  
 else:  
 print(f"Количество месяцев для достижения цели равно {count // WEEK\_PER\_MONTH + 1}")

***6.4 Тестирование работы программы с проверкой***

Для проверки задания в MS Excel создана таблица. В ячейку J3 и K3 записаны стартовое и критическое значения. Ячейка L3 определяет количество особей через неделю и задана формулой:  
L3 = ЕСЛИ(J3 \* 1,1 < K3; J3 \* 1,1;(J3 \* 1,1) / 3).

Далее в ячейку M3 записано специальное значение, равное стартовому (M3 = =J3). В ячейке N3 записана цель. В ячейке O3 высчитывается количество особей через месяц и она задана формулой: O3 =M3 \* СТЕПЕНЬ(1,1;4), а ячейка P3 является счётчиком. Ячейка M4 принимает значение O3(M4 = O3), а ячейка P4 задана формулой: P4 = =ЕСЛИ(M4 < $N$3;1;0). Столбцы M, O и P продлены вниз на 100 ячеек(формула столбца O взята из O3). Ячейка счётчика P3 задана формулой: P3 = 1 + сумм(P4:P104) Таким образом когда специальное значение станет больше цели, счётчик перестанет увеличиваться.

Таблица 6 - Тестирование работы и проверка решений программы №6

|  |  |  |
| --- | --- | --- |
|  | Решение Python | Решение MS Excel |
| 1 | Введите начальное количество особей **20**  Введите критическое значение **50**  Число особей через неделю равно 22.0 Введите цель **100** Количество месяцев для достижения цели равно 5 | |  |  |  |  |  | | --- | --- | --- | --- | --- | | старт | крит. | через неделю | цель | месяцы | | 20 | 50 | 22 | 100 | 5 | |
| 2 | Введите начальное количество особей **32** Введите критическое значение **32** Число особей через неделю равно 11.7333 Введите цель **500** Количество месяцев для достижения цели равно 8 | |  |  |  |  |  | | --- | --- | --- | --- | --- | | старт | крит. | через неделю | цель | месяцы | | 32 | 32 | 1.73333 | 500 | 8 | |
| 3 | Введите начальное количество особей **500**  Введите критическое значение **532**  Число особей через неделю равно 183.3333  Введите цель **100000**  Количество месяцев для достижения цели равно 14 | |  |  |  |  |  | | --- | --- | --- | --- | --- | | старт | крит. | через неделю | цель | месяцы | | 500 | 532 | 183.333 | 100000 | 14 | |

**Задание 7**

***7.1 Постановка задачи***

Дан ряд:

Посчитать сумму членов ряда с точностью до ε.

***7.2 Алгоритм решения***

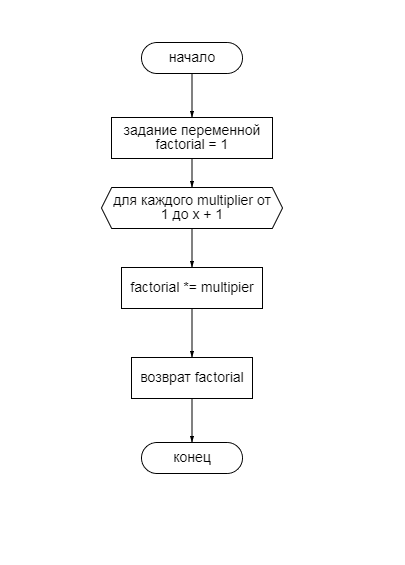
******

Рис.15 – Блок-схема функции *fact(x)*

**

Рис.16 – Блок-схема программы №7

***7.3 Код программы***

def task7():  
 def fact(x):  
 factorial = 1  
 for multiplier in range(1, x + 1):  
 factorial \*= multiplier  
 return factorial  
  
 line\_range = int(input("Введите длину последовательности "))  
 epsilon = "{0:." + str(input("Введите количество знаков после запятой ")) + "f}"  
 x = int(input("Введите x "))  
 summ = 0  
 for count in range(0, line\_range + 1):  
 summ += x\*\*count/fact(count)  
 print(f"Сумма последовательности при x = {x} и длине {line\_range} равна", float(epsilon.format(summ)))

***7.4 Тестирование работы программы с проверкой***

Для проверки решения используется сайт Wolfram Alpha.

Таблица 7 - Тестирование работы и проверка решений программы №7

|  |  |  |
| --- | --- | --- |
|  | Решение Python | Решение Wolfram Alpha |
| 1 | Введите длину последовательности **8** Введите количество знаков после запятой **5** Введите x **7** Сумма последовательности при x = 7 и длине 8 равна 656.56944 |  |
| 2 | Введите длину последовательности **5** Введите количество знаков после запятой **2** Введите x **30** Сумма последовательности при x = 30 и длине 5 равна 38731.0 |  |
| 3 | Введите длину последовательности 3 Введите количество знаков после запятой 4 Введите x 325 Сумма последовательности при x = 325 и длине 3 равна 53138.5 |  |

**Задание 8**

***8.1 Постановка задачи***

Посчитать значения выражений:

***8.2 Алгоритм решения***

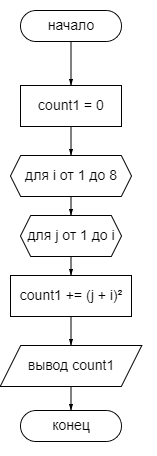


Рис.17 – Блок-схема вычисления выражения 1

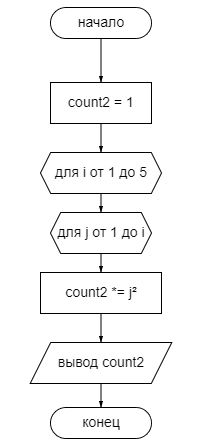


Рис.18 – Блок-схема вычисления выражения 2

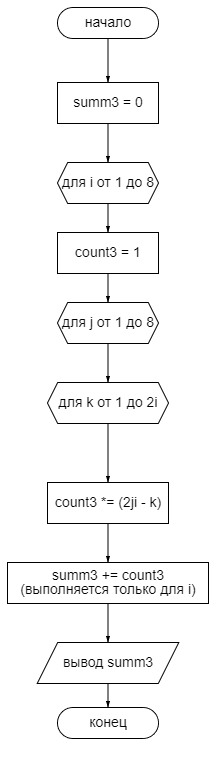


Рис. 19 - Блок-схема вычисления выражения 3

***8.3 Код программы***

def task8():  
 count1, count2 = 0, 1  
 for i in range(1, 9):  
 for j in range(1, i + 1):  
 count1 += (j + i)\*\*2  
 print(f"Значение первого выражения равно {count1}")  
 for i in range(1, 6):  
 for j in range(1, i + 1):  
 count2 \*= j  
 print(f"Значение второго выражения равно {count2}")  
 summ3 = 0  
 for i in range(1, 9):  
 count3 = 1  
 for j in range(1, 9):  
 for k in range(1, 2 \* i + 1):  
 count3 \*= (2 \* j \* i - k)  
 summ3 += count3  
 print(f"Значение третьего выражения равно {summ3}")

***8.4*** ***Тестирование работы программы с проверкой***

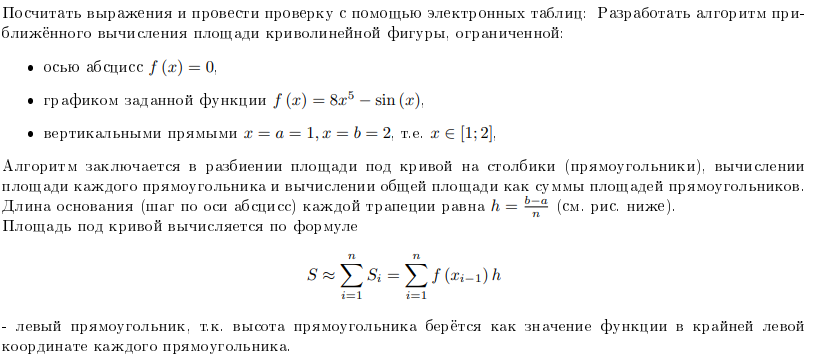
Для проверки решения используется сайт Wolfram Alpha

Таблица 8 - Тестирование работы и проверка решений программы №8

|  |  |  |
| --- | --- | --- |
|  | Решение Python | Решение Wolfram Alpha |
| 1 | Значение первого выражения равно 3336 |  |
| 2 | Значение второго выражения равно 34560 |  |
| 3 | Значение третьего выражения равно 0 |  |

**Задание 9**

***9.1 Постановка задачи***



***9.2 Алгоритм решения***

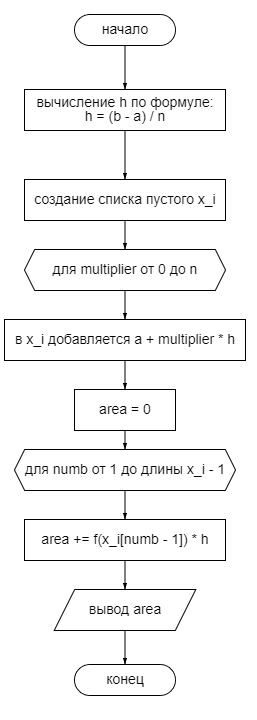


Рис.20 – Блок-схема функции rect\_area(a, b, n = 200000)

***9.3 Код программы***

def task9():  
 def f(x):  
 return 8 \* x\*\*5 - np.sin(x)  
  
 def visual\_function():  
 *"""  
 функция отображает график заданной функции  
 """* axes = plt.subplot()  
 x\_vector = np.linspace(1, 2, 30)  
 y\_vector = f(x\_vector)  
 axes.plot([1, 2], [0, 0], color='k')  
 axes.plot([1, 1], [0, f(1)], color='k', linestyle='--')  
 axes.plot([2, 2], [0, f(2)], color='k', linestyle='--')  
 axes.plot(x\_vector, y\_vector, color='red')  
 plt.show()  
  
 def rect\_area(a, b, n=200000):  
  
 h = (b - a) / n  
 x\_i = []  
 for multiplier in range(n + 1):  
 x\_i.append(a + (h \* multiplier))  
  
 area = 0  
 for numb in range(1, len(x\_i)):  
 area += f(x\_i[numb - 1]) \* h  
 print("Площадь фигуры равна", "{0:.2f}".format(area))  
  
 rect\_area(1, 2)  
 visual\_function()

***9.4 Тестирование работы программы с проверкой***

Для проверки используется Wolfram Alpha.

Таблица 9 - Тестирование работы и проверка решений программы №9

|  |  |  |
| --- | --- | --- |
|  | Решение Python | Решение Wolfram Alpha |
| 1 | *rect\_area(1, 2)*  Площадь фигуры равна 83.04 |  |
| 2 | *rect\_area(3, 5)*  Площадь фигуры равна 19862.49 |  |
| 3 | *rect\_area(-1, 5)*  Площадь фигуры равна 20831.37 |  |