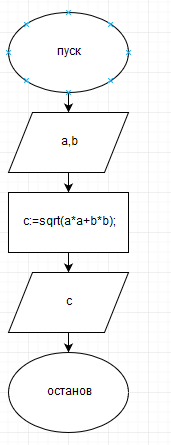
1. Линейные вычислительные процессы (кто сделает блок-схему будет лапочкой) (я ушла спать, гуд найт)
2. Пользователь вводит два числа. Одно присваивается одной переменной, а второе - другой. Необходимо поменять значения переменных так, чтобы значение первой оказалось во второй, а второй - в первой.
3. Вычислить площадь и периметр

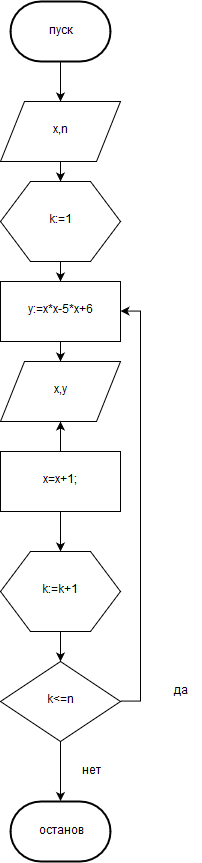
* треугольника по данным трем сторонам,
* прямоугольника по данным ширине и высоте,
* круга по заданному радиусу.

Длины сторон и радиус вводятся пользователем. (ЭТО ТРИ РАЗНЫЕ ЗАДАЧИ ЕСЛИ ЧТО)

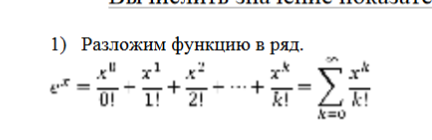
1. По двум введенным пользователем катетам вычислить длину гипотенузы.

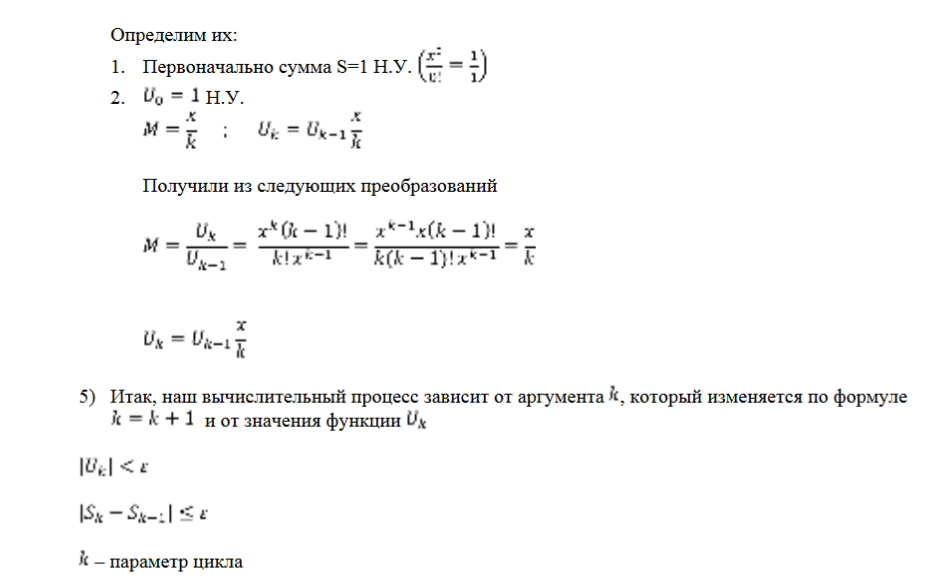
на большее я не способна сегодня

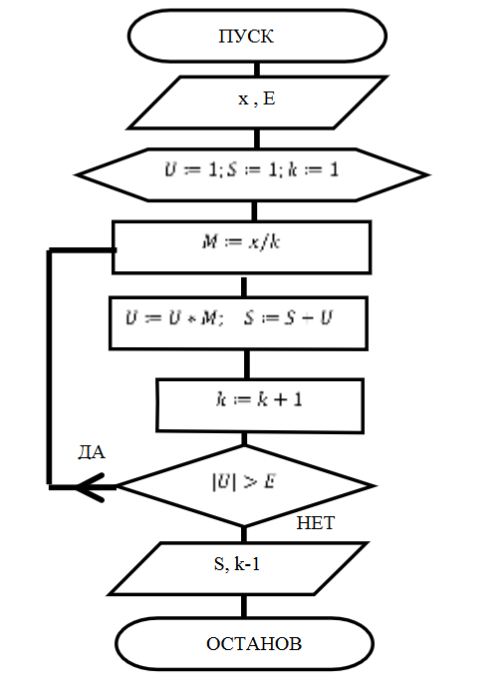
1. Комбинированные вычислительные процессы-
2. Детерминированные циклические вычислительные процессы с Управлением по аргументу Построить график в диапозоне xнач=1 , xконеч= 30 .



1. Итерационные циклические вычислительные процессы с Управлением по индексу и функции. (на примере вычисления элементарных функций)







(на примере построения переходной характеристики заряда конденсатора в схеме RC цепочки с заданной точностью ε).

Разработать структурную схему вычислительного

алгоритма для построения переходной характеристики

заряда конденсатора в схеме RC цепочки с заданной

точностью ε:

Uвых = Uвх ( 1 - e (-t/RC) )

В блоке 2 (просмотреть) параметру цикла t присваивается

его начальное значение, равное нулю.

В этой задаче выходное напряжение Uвых будет

приближаться к Uвх при t → ∞.

Рекуррентная формула:

Для аргумента функции t выберем рекуррентную

зависимость

ti = ti-1 + ht

На что обращаем внимание при организации вычисления :

Вычислительный процесс остановим при выполнении

условия

| Uвых - Uвх | ≤ ε

Если блоки 5 и 6 поменять местами, то проверку условия

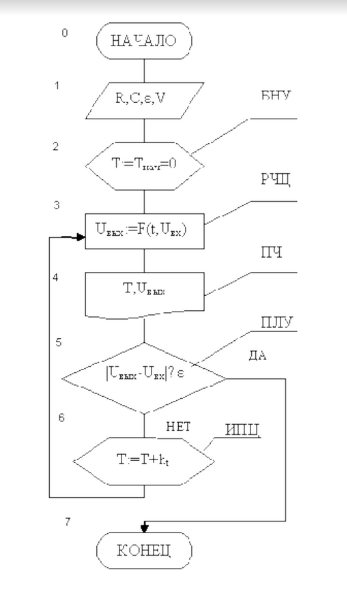
целесообразно записать:

| Uвых - Uвх | > ε

При выполнении этого условия следует продолжать

вычисления и нужно идти на блок 3, а при невыполнении -

на конец.



1. Детерминированные циклические вычислительные процессы с управлением по индексу
2. Сдвинуть элементы массива в указанном направлении (влево или вправо) и на указанное число шагов. Освободившиеся ячейки заполнить нулями. Выводить массив после каждого шага.  
   **сделайте блок-схему, прошу**  
   const N = 9;  
   var   
    arr: array[1..N] of integer;  
    qty: integer;  
    i,j: byte;  
   begin  
    readln(qty);  
    for i:=1 to N do begin  
    arr[i] := i\*100 + i\*10 + i;  
    write(arr[i]:4);  
    end; writeln;  
    for j:=1 to abs(qty) do begin  
    if qty > 0 then begin  
    for i:=N downto 2 do   
    arr[i] := arr[i-1];  
    arr[1] := 0;  
    end  
    else begin  
    for i:=1 to N-1 do   
    arr[i] := arr[i+1];  
    arr[N] := 0;  
    end;  
    for i:=1 to N do   
    write(arr[i]:4);  
    writeln;  
    end;  
   end.
3. Две матрицы заполняются вводом с клавиатуры. В ячейки третьей матрицы такой же размерности записывать бОльшие элементы из соответствующих ячеек первых двух матриц.  
   const  
    M = 4;  
    N = 3;  
   var   
    a, b, c: array[1..N,1..M] of integer;  
    i,j: byte;  
   begin  
    writeln('Первая матрица: ');  
    for i:=1 to N do   
    for j:=1 to M do read(a[i,j]);  
      
    writeln('Вторая матрица: ');  
    for i:=1 to N do   
    for j:=1 to M do read(b[i,j]);  
      
    writeln('Результирующая матрица: ');  
    for i:=1 to N do begin  
    for j:=1 to M do begin  
    if a[i,j] > b[i,j] then  
    c[i,j] := a[i,j]  
    else  
    c[i,j] := b[i,j];  
    write(c[i,j], ' ');  
    end;  
    writeln;  
    end;  
   end.
4. Разветвляющиеся вычислительные процессы.
5. Определить четверть координатной плоскости, которой принадлежит точка. Координаты точки ввести с клавиатуры.  
   **Пояснение к задаче и алгоритм решения**

* Если у точки обе координаты (x и y) положительны, то она принадлежит первой четверти.
* Если координата x отрицательна, а y положительна, то точка находится во второй четверти.
* Если обе координаты отрицательны, то точка принадлежит третьей координатной четверти.
* Если x положительна, а y отрицательна, то точка находится в IV четверти.

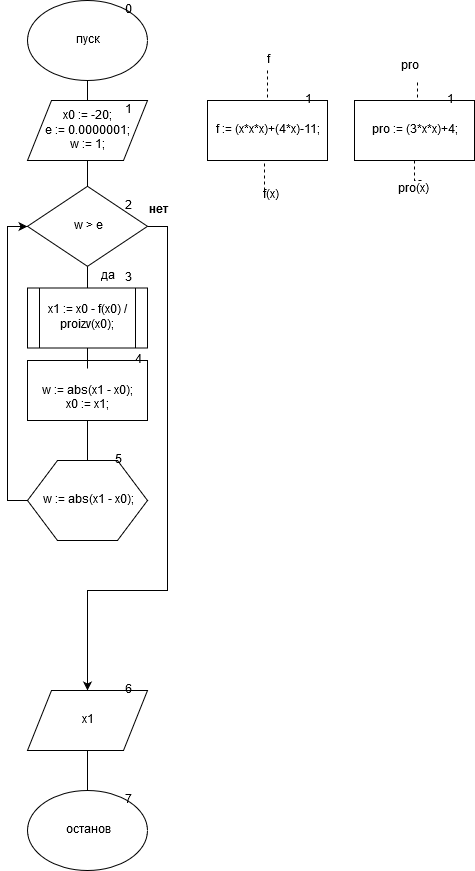
1. Найти корни квадратного уравнения и вывести их на экран, если они есть. Если корней нет, то вывести сообщение об этом. Конкретное квадратное уравнение определяется коэффициентами a, b, c, которые вводит пользователь
2. Даны положительные целые числа A, B, C. Выяснить существует ли треугольник с длинами сторон A, B, C.

USES Crt;   
VAR A, B, C: INTEGER;   
BEGIN   
ReadLn(A,B,C);   
IF (A+B>C) AND (A+C>B) AND (B+C>A) Then WriteLn('Треугольник построить можно') Else WriteLn('Треугольник построить нельзя');   
END.

1. Итерационные циклические вычислительные процессы с управлением по функции.

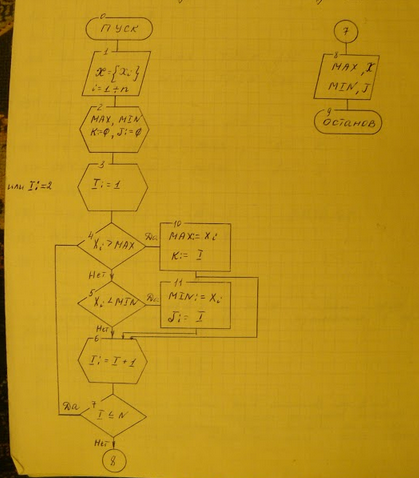
x^3+4x-11=0 решить нелинейное уравнение методом Ньютона

на отрезке от 20 до -20 с точностью 10e-7



1. Итерационные циклические вычислительные процессы с управлением по индексу и функции.

# Дан массив от 1 до n. Найти максимальный элемент массива



# a[1..n]. Сколько чисел со значениями >50? (за эту блок-схему не ручаюсь, кто проверит и исправит получит нихуя)

1. Многоступенчатые циклические вычислительные процессы

Доказать гипотезу Сиракуз на диапазоне чисел. Гипотеза Сиракуз утверждает, что любое натуральное число сводится к единице в результате повторения следующих действий над самим числом и результатами этих действий.

* Если число четное следует разделить его на 2.
* Если нечетное, то умножить его на 3, прибавить 1 и разделить на 2

пишу код, мб кто блок-схему ебнет:   
**var**

n, i, a, b: **integer**;

**begin**

readln(a,b);

**for** i:=a **to** b **do** **begin**

n := i;

write('-> ', n, ' ');

**while** n <> 1 **do** **begin**

**if** (n **mod** 2) = 0 **then**

n := n **div** 2

**else**

n := (3\*n + 1) **div** 2;

write(n,' ');

**end**;

writeln;

**end**;

**end**.