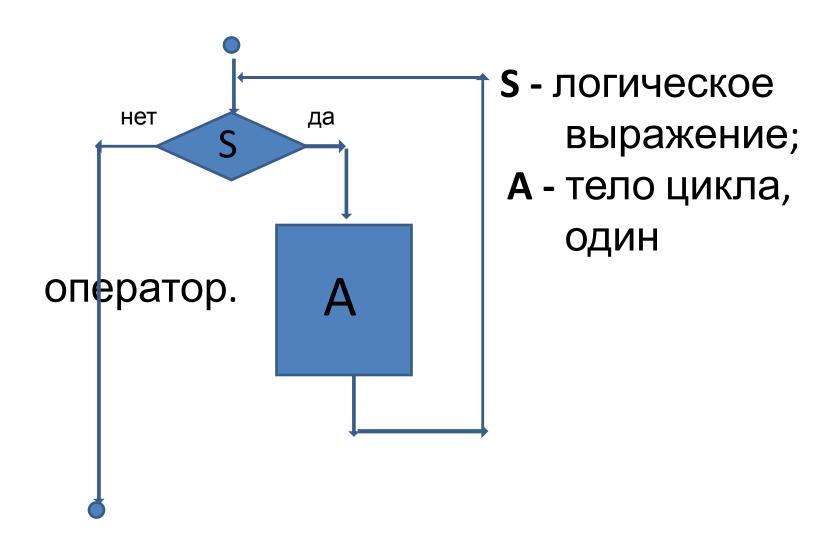
Оператор цикла с предусловием

While S Do A;



В этом операторе тело цикла будет выполняться до тех пор, пока значение выражения **S** истинно.

Если при входе в цикл значение **S** есть **False**, тело цикла не выполнится ни разу.

Пример.

Задан бесконечный ряд:

$$x - \frac{x^3}{3!} + \frac{x^5}{5!} - \dots + (-1)^n \frac{x^{2n+1}}{(2n+1)!} - \dots$$

Подсчитать сумму ряда с заданной точностью.

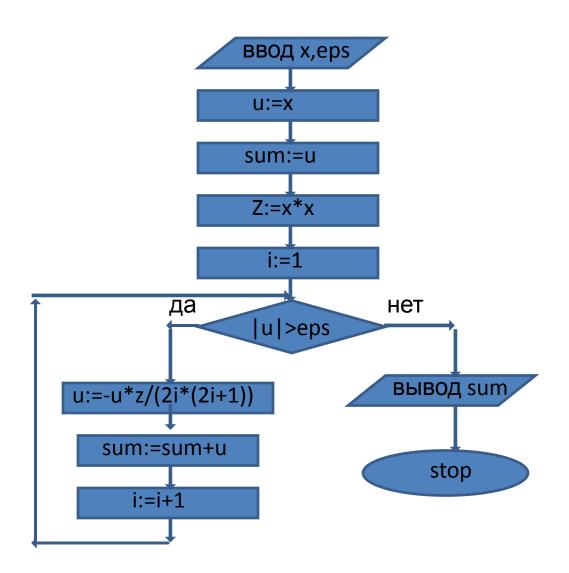
- Для решения этой задачи надо суммировать члены ряда до тех пор, пока абсолютная величина прибавляемого члена не станет меньше значения требуемой точности.
- Полученная при этом сумма есть сумма ряда с заданной точностью.
- Алгоритм решения этой задачи состоит из цикла, для которого заранее не известно число повторений.

Легко заметить, что, имея значение (i - 1) - го члена ряда, можно получить i - ый член, используя рекуррентную формулу:

$$u_i = -u_{i-1} * \frac{x^2}{2i * (2i+1)}$$

- В этой формуле і номер члена, u_і і-ый член ряда.
- Эта рекуррентная формула получена делением в общем виде следующего члена ряда на предыдущий.

- Индексы в этой формуле указывают на то, что в правой части стоит предыдущий член, а в левой получаемый из него следующий.
- Т.к. после получения нового значения члена ряда старое значение больше не нужно, новое значение можно записать на место старого, т.е. использовать для них одну и ту же переменную.
- В программе это выразится в том, что в рекуррентной формуле в левой и правой частях будет присутствовать одна и та же переменная, содержащая значение члена ряда.
- Естественно перед началом вычислений в цикле эта переменная должна получить значение первого члена ряда.



```
Var x,y,z,sum,u ,eps: real;
   i: integer;
Begin
 Write('x=');
  Readln(x); \{Bвод значения x\};
  Write('eps=');
  ReadIn(eps); {Ввод значения eps};
  u := x;
  i := 1;
  sum := u;
  z := x^*x;
        {цикл с предусловием для нахождения суммы ряда}
   While abs(u) > eps Do
     Begin
       u := -u * z / (2*i*(2*i+1));
       sum := sum + u;
       i := i + 1;
      End;
   Writeln('sum = ', sum)
 end.
```

Цикл, в котором заранее не известно число повторений, называется **итерационным циклом.**

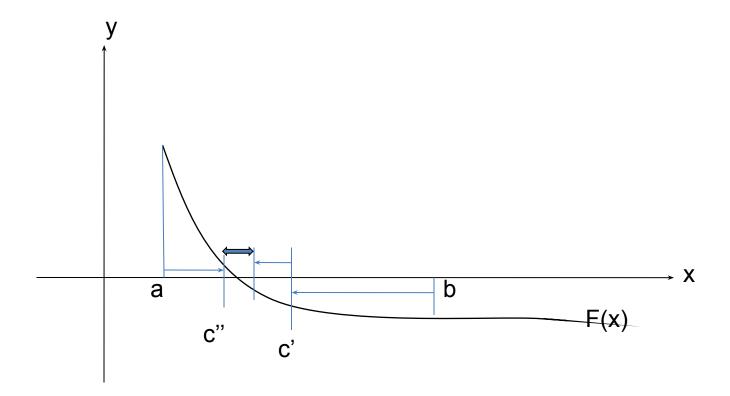
Пример.

<u>Приближённое решение нелинейного</u> <u>уравнения методом бисекции.</u>

Задано уравнение f(x) = 0.

Известно, что на отрезке [a,b] оно имеет один корень.

Требуется найти корень с точностью Е.



Решение уравнения методом бисекции заключается в следующем.

- 1. Исходный отрезок делится пополам.
- Выбирается та половина, на которой есть корень (отрезок уменьшился вдвое).
- 3. Процесс продолжается до тех пор, пока длина отрезка не станет меньше значения требуемой точности **E**.

Алгоритм Ввод а,в,Е да нет |b-a|>E c:=(a+b)/2c := (a+b)/2нет да Вывод с f(a)*f(c)>0stop b:=c a:=c

```
Program uravn;
VAR a,b,c,E:REAL;
BEGIN
 WRITE(' a='); {ввод границы отрезка}
 readIn(a);
 WRITE(' b=');
 readIn(b);
 WRITE(' E='); {ввод значения точности}
 readIn(E);
 while abs(b-a) >E do {цикл поиска корня}
   BEGIN
    c:=(a + b)/2;
    if (sqr(a)-2) * (sqr(c)-2) > 0 then a:=c
                             else b:=c;
   END;
 c:=(a+b)/2; {результат середина отрезка}
 writeln(' c=',c); {вывод на экран результата}
 readIn
end.
```