

**Оператор цикла repeat**

**Оператор цикла repeat** позволяет организовать цикл с неизвестным числом повторений, так как он зависит от вычислений в операторе.

Общий вид оператора

**repeat s until (b)**

b- логическое выражение;

s- тело цикла.

Если логическое выражение имеет значение **false** то выполняются операторы входящие в тело цикла. Как только логическое выражение принимает значение **true** выполнение операторов тела цикла прекращается.

Значения переменных входящих в условие должны изменяться в теле цикла иначе цикл никогда не завершится.

## Пример: 1;

Разработать алгоритм и написать программу вычисления суммы членов ниже приведенного ряда с неизвестным числом повторений, суммирование завершить при выполнении условия :  $UN < E$

Где  $UN$  – член ряда;

$E$  - точность с которой нужно завершить вычисления. Это может быть  $E = 0.1$ ;  $E = 0.001$  и

$$Y = x + \frac{x^2}{2} + \frac{x^3}{3} + \dots + \frac{x^n}{n}$$

Переменные:

X – тип real;

Y – тип real;

I – счетчик повторений цикла тип integer;

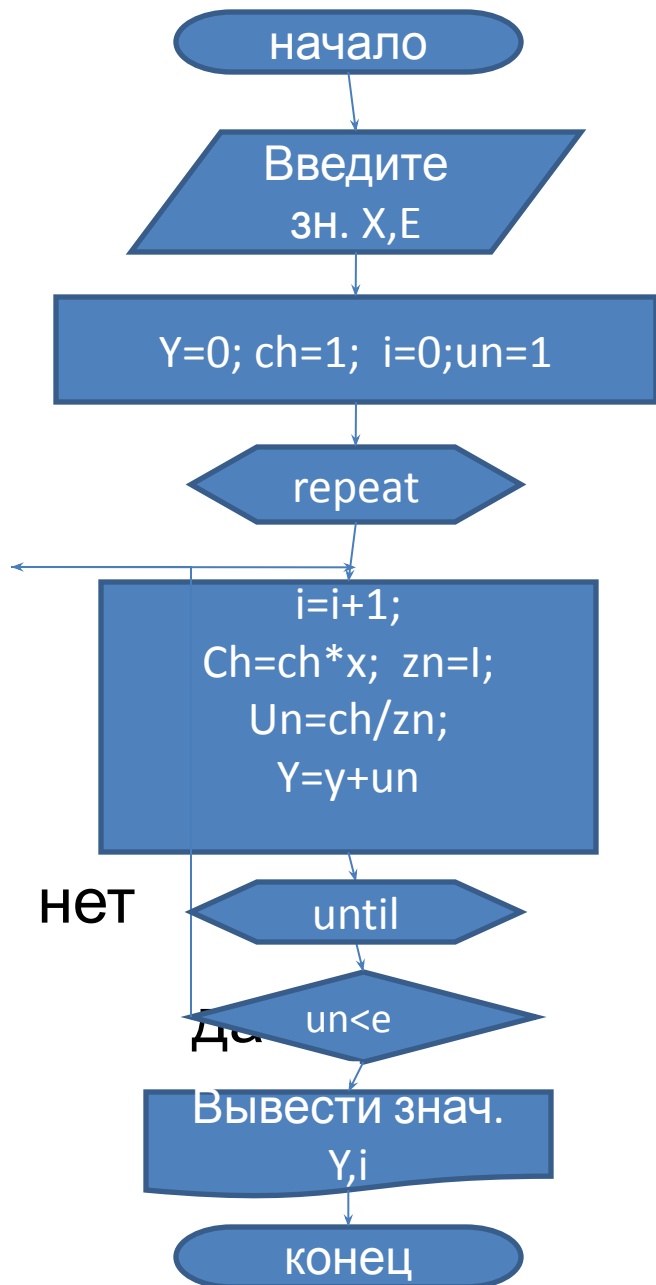
E – точность тип real;

UN - член ряда тип real;

ch- числитель тип real;

zn- знаменатель тип real.

# Разрабатываем блок-схему алгоритма:



```
Program cikl;  
Var x,y,ch,zn,e,un:real;  
i:integer;  
Begin  
writeln('vv.zn. X,e');  
readln(x,e);  
y:=0; ch:=1; zn:=0;un:=1;i:=0;  
Repeat  
i:=i+1;  
  ch:=ch*x;  
zn:=1;  
un:=ch/zn;  
y:=y+un;  
until un<e;  
writeln('y=', y:6:2, ' k-во ЦИКЛОВ i=', i:3);  
readln; end.
```

Пример 2:

Разработать алгоритм и написать программу вычисления суммы членов ниже приведенного ряда с неизвестным числом повторений, суммирование завершить при выполнении условия : **UN** < **E**

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

Переменные:

X – тип real;

Y – тип real;

I – счетчик повторений цикла тип integer;

E – точность тип real;

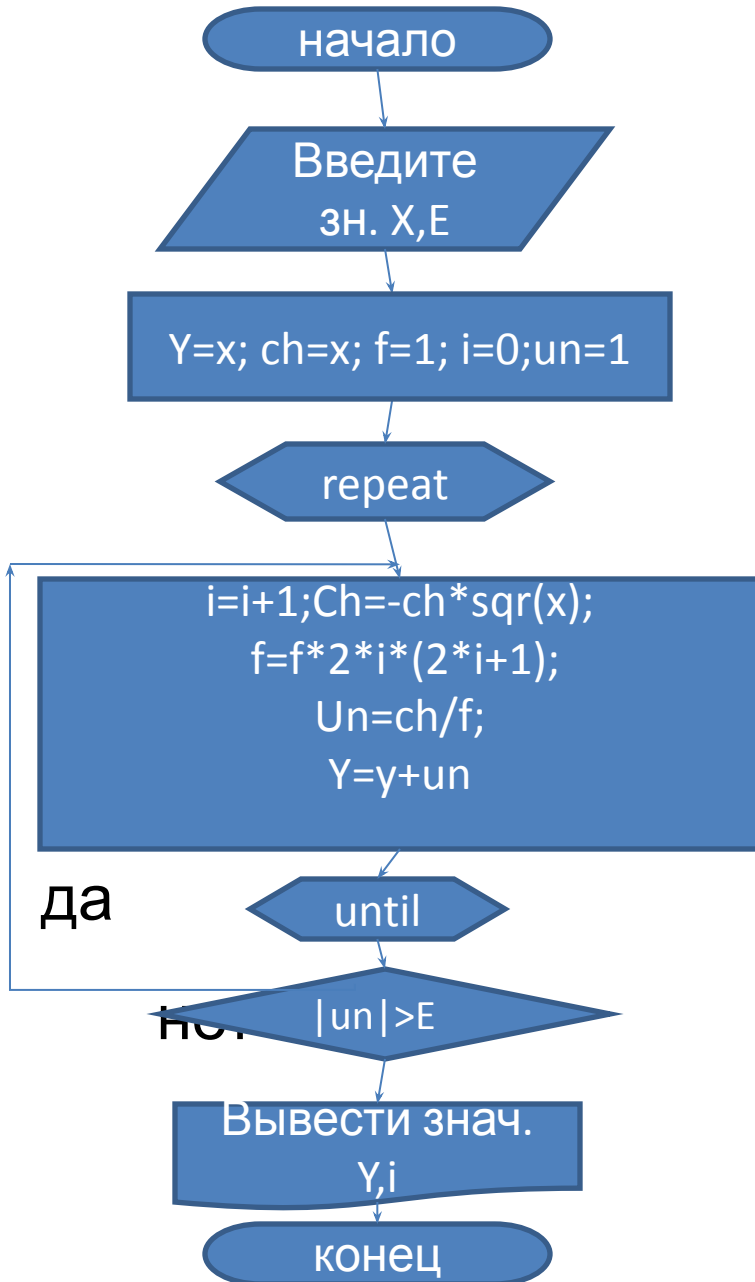
UN - член ряда тип real;

ch- числитель тип real;

f- факториал тип longint.



# Разрабатываем блок-схему алгоритма:



```
Program cv;  
var l: integer;  
f: longint;  
ch, un, x, y, e: real;  
begin  
writeln('vv.zn. x,e');  
readln(x,e);  
y:=x; ch:=x; f:=1; un:=1; i:=0;  
Repeat  
i:=i+1  
ch:=-ch*sqr(x);  
f:=f*2*i*(2*i+1);  
un:=ch/f;  
y:=y+un;  
until abs(un)<e  
writeln('y=', y:6:2, ' ЧИСЛО ЦИКЛОВ i=', i:3);  
readln; end.
```

# Табулирование функций

При табулировании функций аргумент  $X$  в цикле должен меняться с шагом  $dX$ .

Прежде чем организовывать в программе цикл необходимо ввести начальное значение аргумента  $X$  и значение шага  $dX$ , построить шапку таблицы, а после выхода из цикла вывести линию закрытия таблицы.

Пример:

Разработать алгоритм и написать программу табулирования функции  $\sin(x)$

Переменные:

Y – функция тип real;

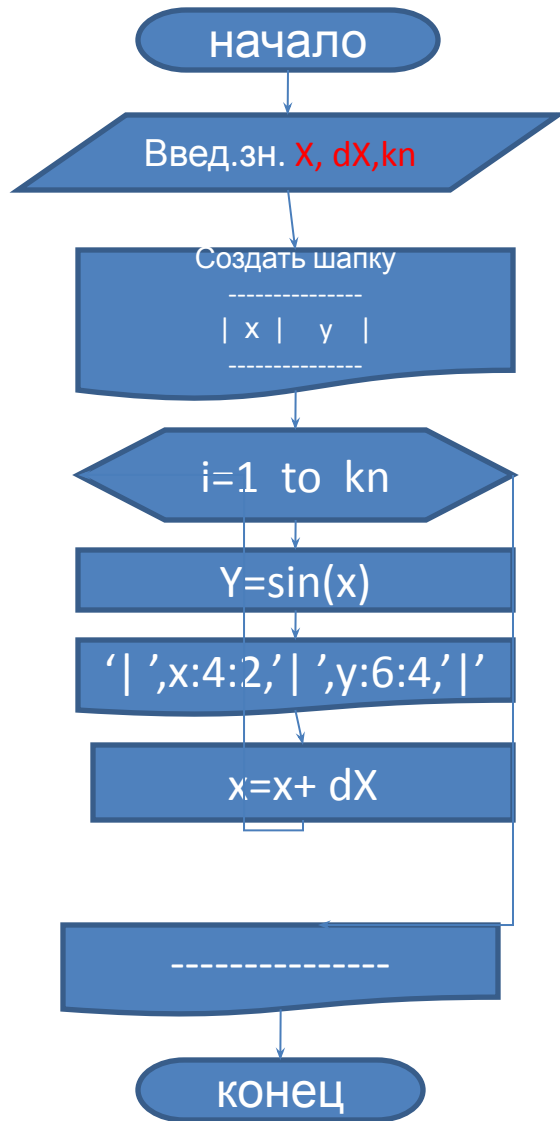
X – аргумент тип real;

dX – шаг изменения аргумента тип real;

I – параметр цикла тип integer;

Kn – конечное значение параметра цикла тип integer;

# Разрабатываем блок–схему алгоритма:



```
program tab;
var l,kn: integer;
x,y,dx: real;
begin
writeln('vv.zn. X,dx,kn');
readln(x,dx,kn);
writeln('-----');
writeln('| x | y |');
writeln('-----');
For i:=1 to kn do
begin
Y:=sin(x);
writeln('| ',x:4:2,' | ',y:6:4,' |');
x:=x+dx;
end;
writeln('-----');
readln; end.
```

Пример 2:

Разработать алгоритм и написать программу табулирования функций  **$y_1 = \sin(x)$  если  $x < 0.6$  иначе  $y_2 = \cos(x)$**

Переменные:

$y_1, y_2$  – функция тип real;

$x$  – аргумент тип real;

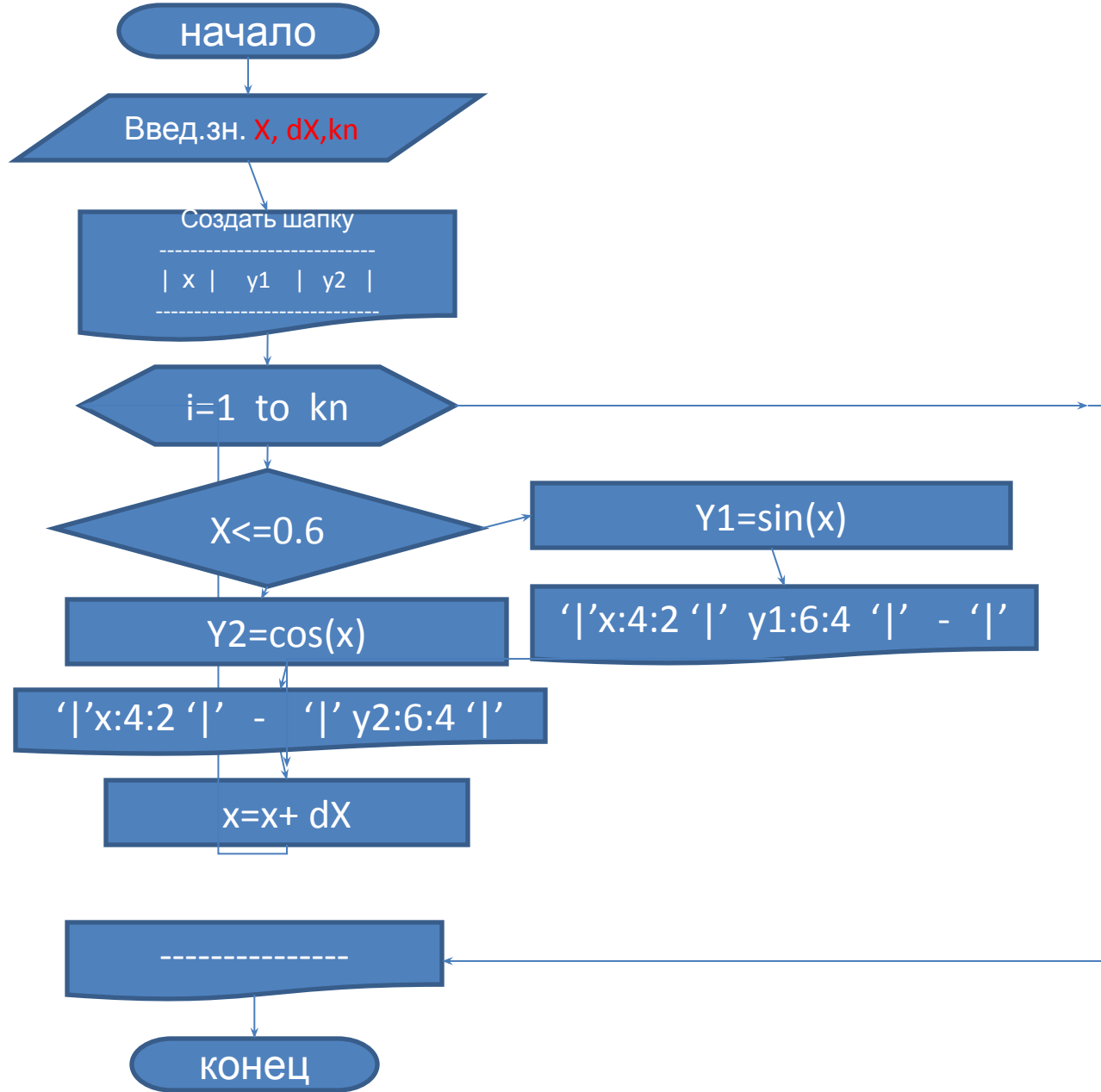
$dx$  – шаг изменения аргумента тип real;

$i$  – параметр цикла тип integer;

$k_n$  – конечное значение параметра цикла тип integer;



# Разрабатываем блок-схему алгоритма:



```

program tab;
var l,kn: integer;
x,y,dx: real;
begin
writeln('vv.zn. X,dx,kn');
readln(x,dx,kn);
writeln('-----');
writeln('| x | y1 | y2 |');
writeln('-----');
For i:=1 to kn do
begin
If x<=0.6 then begin Y1:=sin(x); writeln(' | ',x:4:2,' | 'y1:6:4 , ' | - | ');
end
else begin y2:=cos(x); writeln(' | ',x:4:2,' | - | ',y2:6:4,' | '); end;
x:=x+dx;
end;
writeln('-----');
readln; end.

```