Примеры задач с циклами while и repeat

Итерационные алгоритмы

- Так называются циклические алгоритмы в которых для вычислений в качестве исходных данных для расчета при каждом повторе используется результат полученный на предыдущем шаге. Вычисления повторяются до тех пор, пока разница между двумя последующими результатами не станет достаточно малой.

Примеры заданий

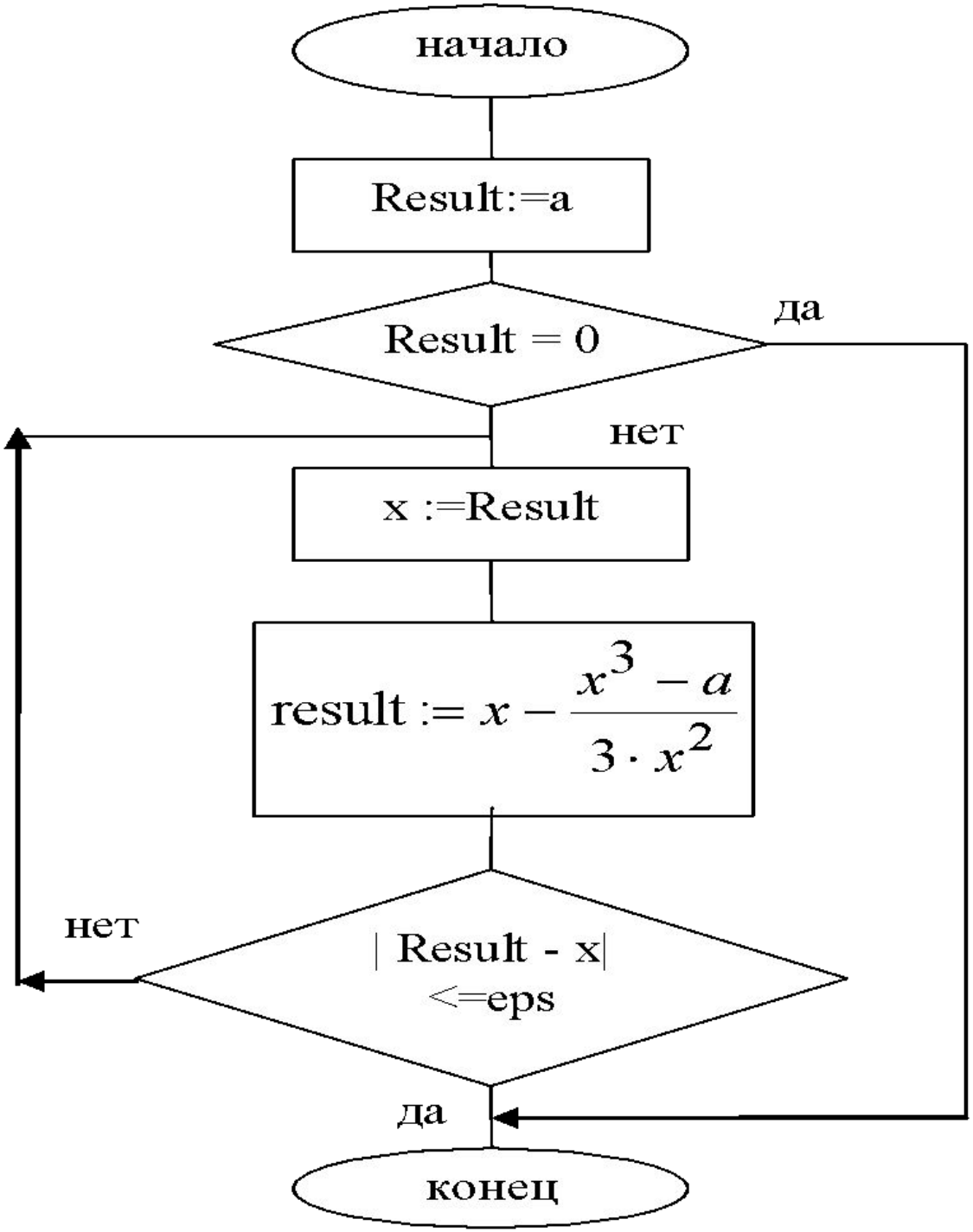
Вариант	Уравнение	Итерационная формула	Ограничения
1	2	3	4
0	$a \cdot e^{-b \cdot x} - x = 0$	$x = a \cdot e^{-b \cdot x}$	ab<1, x0>0
1			a>0, b>0, b/a<6, x0<π/2
2	$\sqrt{a \cdot x} + \sqrt{b \cdot x} - cx = 0$	$x = \dfrac{\sqrt{a \cdot x} + \sqrt{b \cdot x}}{c}$	c>0 x0>0
3	$a \cdot x^2 - b \cdot \sin(x) = 0$	$x = \sqrt{\dfrac{b \cdot \sin(x)}{a}}$	a>1, x0>0

$$arctg(a \cdot x) - bx = 0 \qquad x = \frac{arctg(a \cdot x)}{b}$$

Итерационная формула для вычисления кубического корня

$$x = \sqrt[3]{a}; \qquad x^3 - a = 0;$$

$$x_{i+1} = x_i - \frac{x_i^3 - a}{3 \cdot x_i^2}$$


начало
Result:=a
Result = 0
да
нет
x :=Result
$result := x - \frac{x^3 - a}{3 \cdot x^2}$
нет
| Result - x|
<=eps
да
конец

Программа вычисления кубического корня

```
function mySqrt3(a, eps:real):real;
  var x:real;
begin
   result:=a; if result=0 then exit;
   repeat
     x:=result;
              result:=x-(x*x*x-a)/2/x/x;
   until abs(result-x)<eps;
end;
```

Алгоритмы вычисления сумм бесконечных рядов

- В этих алгоритмах производится суммирование последовательно вычисляемых членов ряда.
- Вычисление каждого следующего члена ряда производится по значению предыдущего.
- Накопление суммы производят до тех пор, пока очередной член ряда не станет достаточно малым числом.
- Алгоритм имеет смысл только в том случае, если ряд сходится, то есть значения членов ряда постепенно уменьшаются.

Примеры заданий

Таблица 5.2 Задания на вычисление сумм бесконечных[рядов			
№	Функция	Ряд	Рекуррентная формула
1	2	3	4
	0 e	$2+\frac{1}{2!}+\frac{1}{3!}+\ldots=2+\sum_{i=2}^{\infty}u_i$	$u_{i+1}=\frac{u_i}{i+1}$
	1 e^x	$1+x+\frac{x^2}{2!}+\frac{x^3}{3!}+\ldots=\sum_{i=0}^{\infty}u_i$	$u_i=u_{i-1}\cdot\frac{x}{i}$

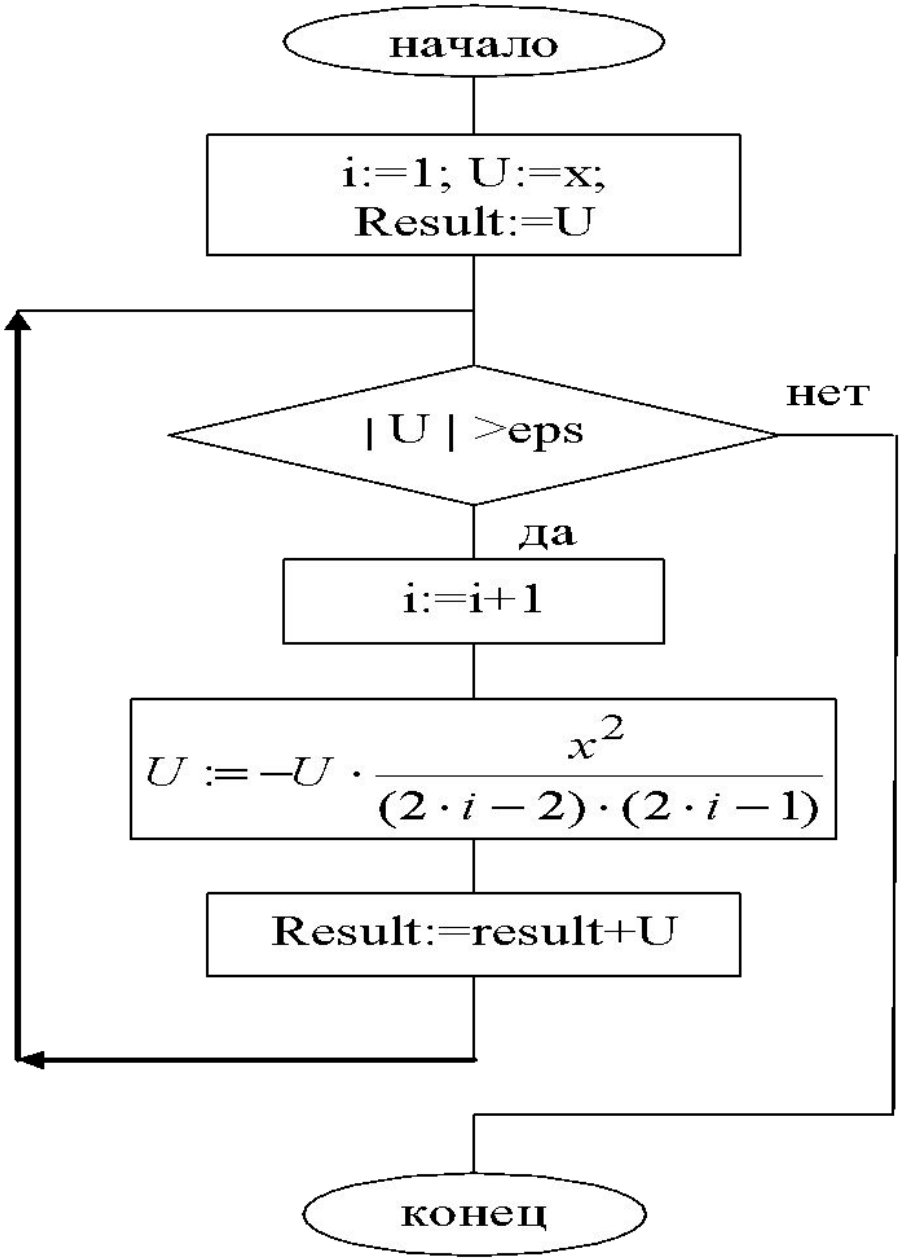
Бесконечный ряд для вычисления синуса

$$\sin(x) =$$

$$x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \ldots =$$

$$\sum_{i=1}^{\infty} (-1)^{i+1} \frac{x^{2 \cdot i - 1}}{(2 \cdot i - 1)!} = \sum_{i=1}^{\infty} U_i$$

$$U_1 = x; \qquad U_i = -U_{i-1} \cdot \frac{x^2}{(2 \cdot i - 2) \cdot (2 \cdot i - 1)}$$


начало
i:=1; U:=x;
Result:=U
| U | >eps
нет
да
i:=i+1
$U := -U \cdot \frac{x^2}{(2 \cdot i - 2) \cdot (2 \cdot i - 1)}$
Result:=result+U
конец

Программа вычисления синуса

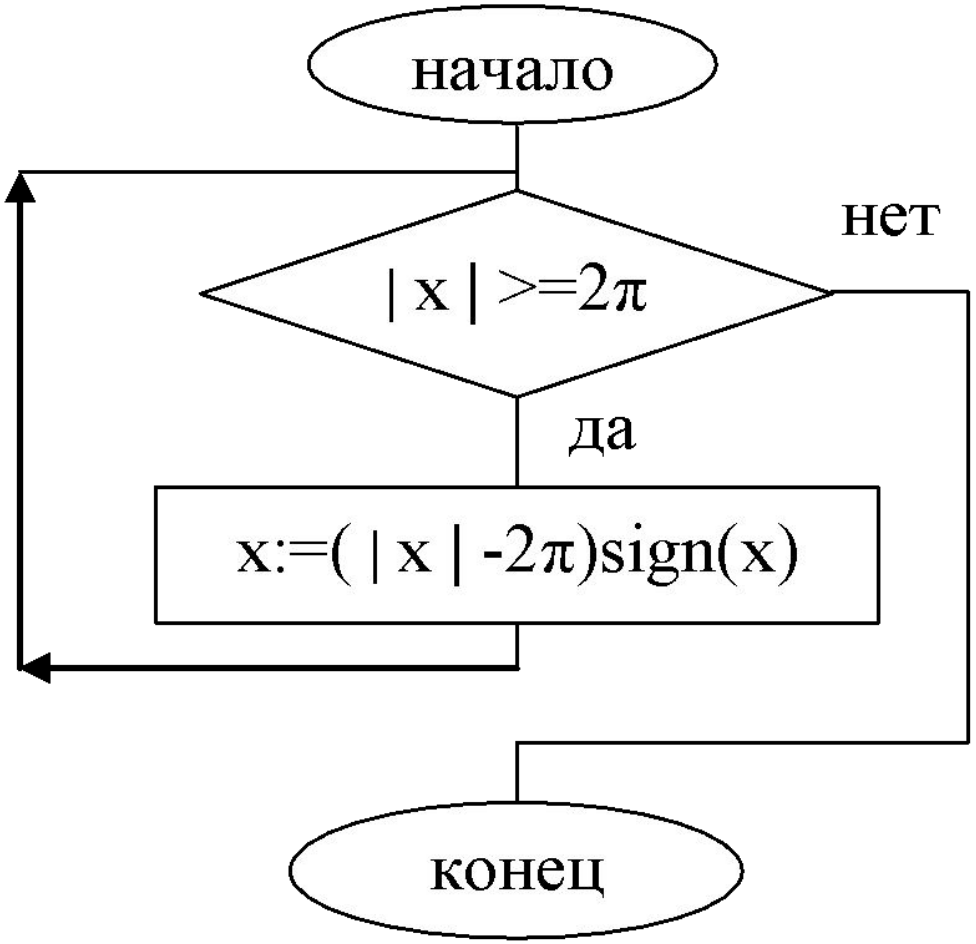
```
function sequence (x,eps:real):real;
  var u:real; i:integer;
begin
   i :=1; u :=x; result := u;
    repeat
          i:=i+1;
          u:=-u*x*x/(2*i-2)/(2*i-1);
          result:=result+u;
   until abs(u)<=eps;
end;
```

Будет ли работать программа при больших значениях *x* ?

Ряд при больших значения**x** *x* сходится медленно,

а факториал и «*x* в степени» растут очень быстро.

Это приводит к тому, что значащие цифры этих чисел перестают помещаться в разрядной сетке и, следовательно, обрезаются, вследствие чего результат искажается.


начало
нет
| x | >=2π
да
x:=(| x | -2π)sign(x)
конец

Реализация программы

```
procedure cutX(var x:real);
begin
   while abs(x)>=2*pi do
    x:=(abs(x)-2*pi)*sign(x);
end;
```

Сколько цифр в целом числе?

```
function numOfFig(n : integer) : integer;
 begin
   result:=0;
   repeat
     n:=n div 10;   result:=result+1;
   until n=0;
 end;
```

Найти сумму цифр в числе

```
function sumOfFig( n : integer ) : integer;
 begin
   result:=0;
   while abs(n)>0 do
   begin
          result := result + (n mod 10);
       n := n div 10;
   end;
 end;
```

Функция random

Существует две функции с таким именем:

function random():real;

Возвращает случайные положительные вещественные числа меньшие1.

function random(m:integer):integer;

Возвращает случайные положительные целые числа меньшие m.

Варианты использования функции random

Двоичные цифры:

fig2 := random(2);

Случайные оценки от 2 до 5:

mark:=2+random(4);

Температура хворих у лікарні:

t:=35+random()*5;

Угадайка

```
function howManyTimes( n : integer ) : integer;
  begin
    result:=0;
    while random(100)<>n do
        result:=result+1;
    end;
```

Среднее арифметическое

```
function average( n : integer ) : integer;
   var sum, count, x : integer;
  begin
   randomize();
   count := n; sum := 0; x:=random(100);
   while count > 0 do
   begin
       sum := sum + howManyTimes(x);
       count := count -1;
   end;
   result := sum div n;
  end;
```