

Богатов Р.Н.

Программирование на языке высокого уровня

Лекция 7.

Методы класса как подпрограммы.

Решение нелинейных уравнений

Кафедра АСОИУ ОмГТУ, 2012

Подпрограмма. Процедура. Функция. Метод

Машинный язык
Подпрограмма
глобальная

```
// пример вызова процедуры на языке Assembler  
mov    ax, 0  
mov    dx, 123  
call   my_proc
```

Pascal:

Процедура
Функция =

```
// пример вызова процедур и функций на языке Pascal  
writeln;  
y := sin(x);
```

C:

Функция м
(процедур

```
// пример вызова функций на языке Си  
getch();  
c = getch();  
y = sin(x);
```

C#, Java и другие:

Данные и код инкапсулированы в классы. Инкапсулированные подпрограммы называются **методами**.

```
// пример вызова методов на языке C#  
textBox1.Hide();  
x = r.NextDouble();  
y = Math.Sin(x);
```

```
public partial class Form1 : Form
```

```
{
```

```
    int N;  
    int[] a;
```

```
private partial class Form1 : Form
```

```
{ {
```

```
    private void button1_Click(object sender, EventArgs e)
```

```
{
```

```
    int N = (int)numericUpDown1.Value;  
    Massiv a = new Massiv(N);  
    Massiv b;
```

```
    a.sluchaino(0, 100);  
    a.vyvod(textBox1.Text);  
    a.sort();  
    a.vyvod(textBox1.Text);  
    b = a;  
    a.perevorot();
```

```
    ...  
    label1.Text = "Максимум = " + a.maximum();  
    label2.Text = "Сумма без крайних эл-в = " + a.part_sum(1, N-2);
```

```
}
```

```
}
```

```
{
```

```
    int sum = 0;  
    for (int k = i; k <= j; k++)  
        sum += a[k];  
    return sum;
```

```
}
```

Д

le

Решение нелинейного ура

```
void metod_del_popolam()
{
    textBox1.Text = "";
    double a = x1, b = x2, c = 0, fc = 0;
    double fa = y(a);
    double fb = y(b);
    // проверка на наличие корней (fa*fb>=0)

    int i = 1;
    for (; i < N; i++)
    {
        c = (a + b) / 2;
        fc = y(c);
        // ...вывод протокола итерации (если нужд
        if (b - c < eps) break;
        if (fa * fc < 0)
        {
            // вывод протокола итерации
            textBox1.Text += String.Format(
                "a = {0:0.###}\t c = {1:0.###}\t b = {2:0.###}\t f(c) = {3:0.###}\r\n",
                a, c, b, fc);

            // вывод результата
            textBox1.Text = String.Format(
                "x = {0}\r\ny(x) = {1}\r\ndx = {2}\r\nИтераций = {3}\r\n\r\n",
                c, fc, b-c, i) + textBox1.Text;
        }
    }
}
```

cos x - x³ = 0

x1 = 0 eps = 0,0001 Вычислить

x2 = 1 N = 1000

Деления пополам | Простых итераций | Касательных (Ньютона) | Хорд

x = 0,86541748046875
y(x) = 0,000170131436310395
dx = 6,103515625E-05
Итераций = 14

a = 0	c = 0,5	b = 1	f(c) = 0,753
a = 0,5	c = 0,75	b = 1	f(c) = 0,31
a = 0,75	c = 0,875	b = 1	f(c) = -0,029
a = 0,75	c = 0,813	b = 0,875	f(c) = 0,151
a = 0,813	c = 0,844	b = 0,875	f(c) = 0,064
a = 0,844	c = 0,859	b = 0,875	f(c) = 0,018
a = 0,859	c = 0,867	b = 0,875	f(c) = -0,005
a = 0,859	c = 0,863	b = 0,867	f(c) = 0,007
a = 0,863	c = 0,865	b = 0,867	f(c) = 0,001
a = 0,865	c = 0,866	b = 0,867	f(c) = -0,002
a = 0,865	c = 0,866	b = 0,866	f(c) = -0,001
a = 0,865	c = 0,865	b = 0,866	f(c) = 0
a = 0,865	c = 0,865	b = 0,865	f(c) = 0
a = 0,865	c = 0,865	b = 0,865	f(c) = 0

Решение нелинейного уравнения

```
double y_x(double x)
{
    return Math.Pow(Math.Cos(x), 1/3.0);
}
```

```
double dy(double x)
{
    return - Math.Sin(x) - 3 * x * x;
}
```

```
void metod_hord()
{
    textBox8.Text = "";
    double a = x1, b = x2, c = 0;
    int i = 1;
    for (; i < N; i++)
    {
        c = a - y(a) * (a - b) / (y(a) - y(b));
        // ...вывод протокола итерации(если нужно)
        if (Math.Abs(c - a) < eps)
            break;
        b = a;
        a = c;
    }
    // ...вывод или возврат результата
}
```

Решение нелинейного уравнения

The image displays four screenshots of a software application used for solving the equation $\cos x - x^3 = 0$. The interface includes input fields for initial values x_1 and x_2 , precision ϵ , and the number of iterations N . A 'Вычислить' (Calculate) button is present. Below the inputs are tabs for different methods: 'Деления пополам' (Bisection), 'Простых итераций' (Simple iterations), 'Касательных (Ньютона)' (Tangent method/Newton's method), and 'Хорд' (Chord method).

The screenshots show the following results:

- Top-left:** Shows the initial state with $x_1 = 0$, $x_2 = 1$, $\epsilon = 0.0001$, and $N = 20$. The results for the bisection method are: $x = 0.865417480$, $y(x) = 0.0001701$, $dx = 6.10351562$, and 14 iterations.
- Top-right:** Shows the results for the simple iteration method: $x = 0.865490915388543$, $y(x) = -5.07918436186694E-05$, $dx = 6.67091921291441E-05$, and 9 iterations.
- Middle-left:** Shows the results for the tangent method (Newton's method): $x = \text{NaN}$, $y(x) = \text{NaN}$, $dx = \text{NaN}$, and 20 iterations. The derivative is $x'=0$, and the function value is $x=\text{бесконечность}$.
- Middle-right:** Shows the results for the chord method: $x = 0.865474303202692$, $y(x) = -8.12609719291757E-07$, $dx = 8.57740354045511E-05$, and 8 iterations. The function values at the endpoints are: $x=0.54, y=0.7$; $x=1.823, y=-6.306$; $x=0.668, y=0.486$; $x=0.751, y=0.307$.
- Bottom-left:** Shows the results for the bisection method with a different set of parameters: $x = 0.865474033101708$, $y(x) = -2.80109269112927E-13$, $dx = 3.0974829889896E-07$, and 12 iterations. The function values at the endpoints are: $x'=25, x=16.667, y=-4630.1$; $x'=16.667, x=11.105, y=-1369.3$; $x'=11.105, x=7.394, y=-403.73$; $x'=7.394, x=4.945, y=-120.70$; $x'=4.945, x=3.278, y=-36.207$; $x'=3.278, x=2.15, y=-10.481$; $x'=2.15, x=1.437, y=-2.832$; $x'=1.437, x=1.042, y=-0.629$; $x'=1.042, x=0.89, y=-0.076$; $x'=0.89, x=0.866, y=-0.002$.
- Bottom-right:** Shows the results for the chord method: $x = 0.000400080016178478$, $y(x) = 0.999999919903953$, $dx = 4.00639956265399E-08$, and 4 iterations. The function values at the endpoints are: $x=0, y=1$; $x=4996.005, y=-124700589343.019$; $x=0, y=1$; $x=0, y=1$.

Домашнее задание

В таблице ниже приведены наиболее часто используемые параметры линейных конгруэнтных генераторов, в частности, в стандартных библиотеках различных компиляторов (функция `rand()`).

»КИХ

Source	m	a	c	выдаваемые биты результата в <code>rand()</code> / <code>Random(L)</code>
Numerical Recipes (англ.)	2^{32}	1664525	1013904223	
MMIX by Donald Knuth	2^{64}	6364136223846793005	1442695040888963407	
Borland C/C++	2^{32}	22695477	1	биты 30..16 в <code>rand()</code> , 30..0 в <code>lrand()</code>
GNU Compiler Collection	2^{32}	69069	5	биты 30..16
ANSI C: Open Watcom, Digital Mars, Metrowerks, IBM VisualAge C/C++	2^{32}	1103515245	12345	биты 30..16
Borland Delphi, Virtual Pascal	2^{32}	134775813	1	биты 63..32 числа (<code>seed * L</code>)
Microsoft Visual/Quick C/C++	2^{32}	214013	2531011	биты 30..16
Apple CarbonLib	$2^{31} - 1$	16807	0	см. Метод Лемера (англ.)

$$X_{k+1} = (aX_k + c) \bmod m,$$

где a и c — некоторые целочисленные коэффициенты. Получаемая последовательность зависит от выбора стартового числа X_0 и при разных его значениях получают различные последовательности случайных чисел. В то же время, многие свойства этой последовательности определяются выбором коэффициентов в формуле и не зависят от выбора стартового числа.