

**Московский государственный технический  
университет им. Н. Э. Баумана**

Факультет «Информатика и системы управления»  
Кафедра ИУ5 «Системы обработки информации и управление»

Курс «Основы программирования»

Отчет по лабораторной работе №3  
«Нахождение корней нелинейного уравнения»

Выполнил:

Студент группы ИУ5-11Б  
Алехин Сергей

Подпись и дата:

Проверил:

Преподаватель каф. ИУ5  
Правдина Анна Дмитриевна

Подпись и дата:

Москва, 2018 г.

## Задание

1. Найти корень уравнения

$$x - \cos(x) = 0$$

простой итерацией, половинным делением и методом Ньютона с погрешностью  $\text{eps} < 0.000001$  и для каждого из трех методов определить количество шагов алгоритма.

2. Выполнить п.1 для  $\text{eps} < 0.00000001$ .

3. Выполнить п.1 для уравнения

$$x - 10\cos(x) = 0$$

и объяснить результаты.

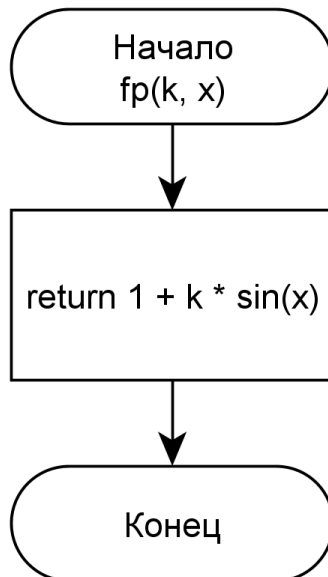
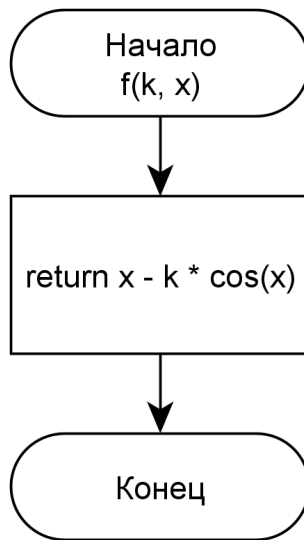
## Разработка алгоритма

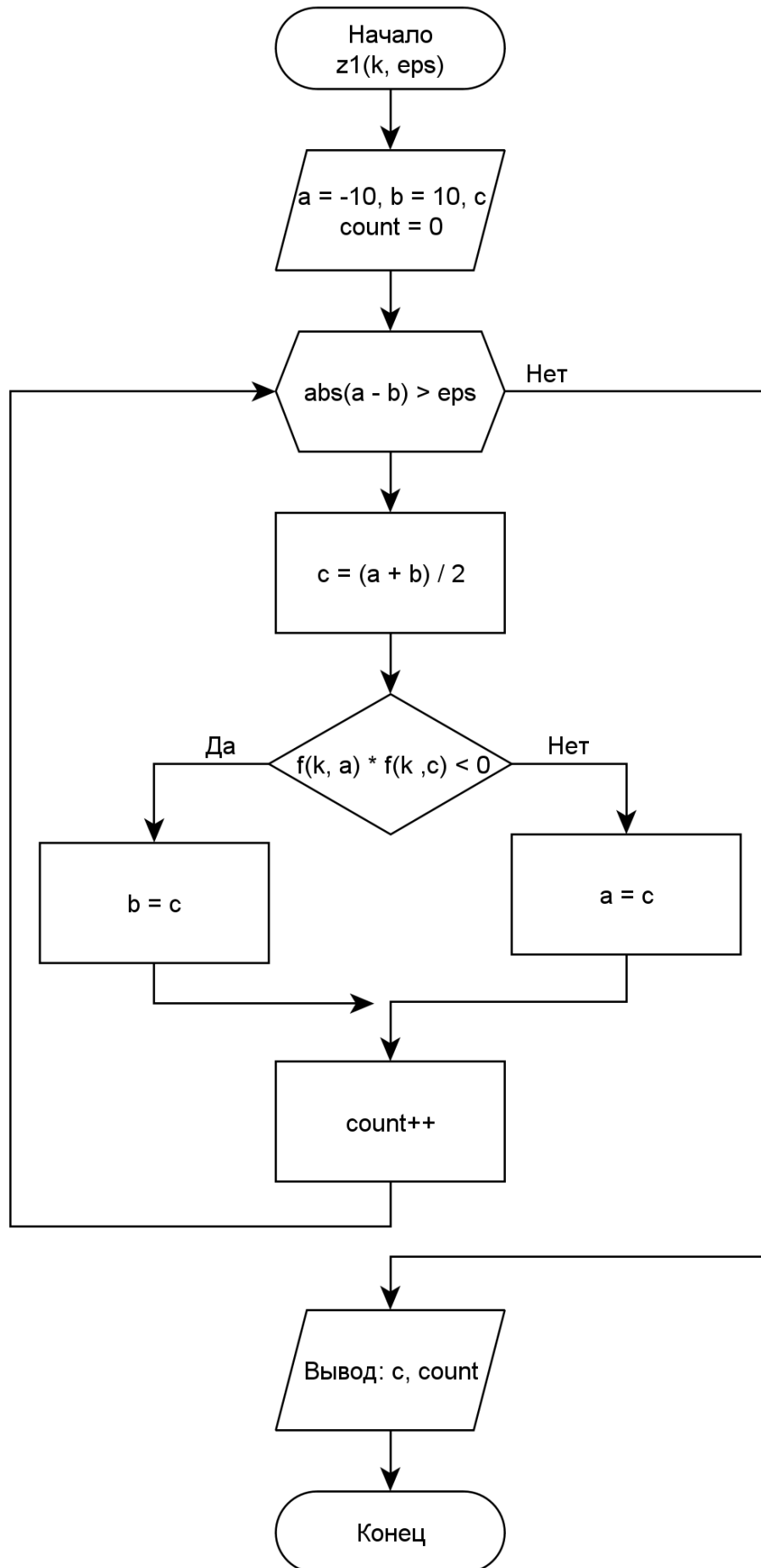
### Входные переменные:

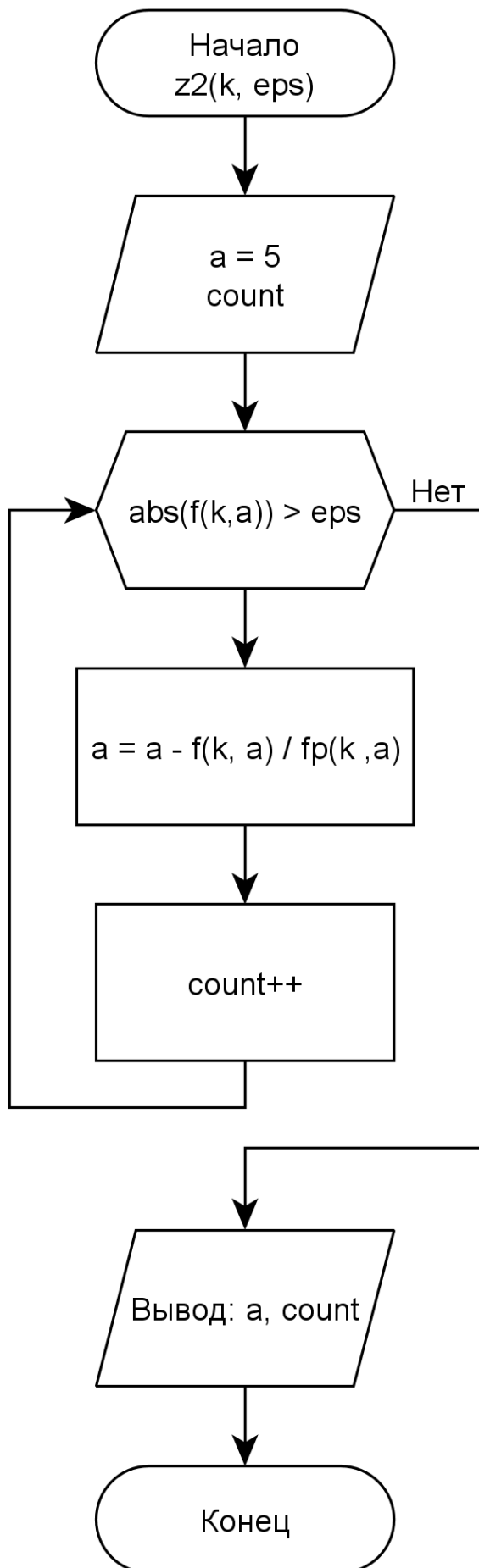
- 1) double eps – отвечает за точность измерений;
- 2) int k – значение коэффициента перед  $\cos(x)$ ;
- 3) int i – переменная цикла;

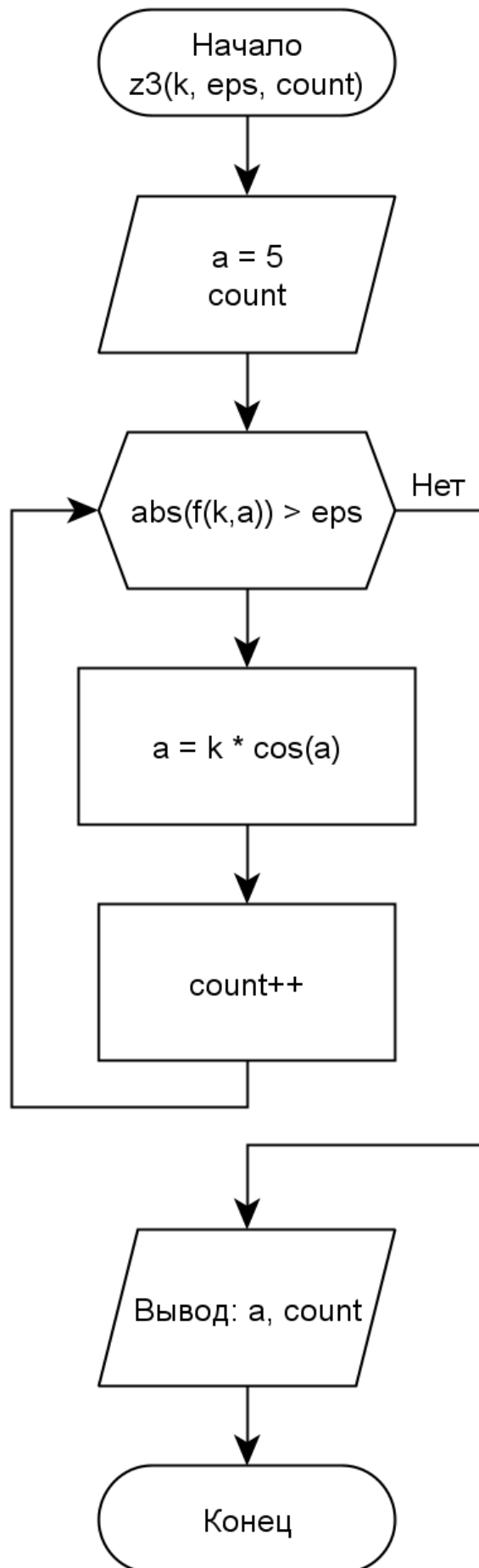
### Функции:

- 1) double f – функция  $x - k * \cos(x)$ ;
  - a. *Входные переменные:*
    - i. int k – коэффициент перед  $\cos(x)$ ;
    - ii. double x – значение переменной x;
  - b. *Возвращенное значение функции:*
    - i. Значение функции  $x - k * \cos(x)$ ;
- 2) double fp – функция производной f;
  - a. *Входные переменные:*
    - i. int k – коэффициент перед  $\cos(x)$ ;
    - ii. double x – значение переменной x;
  - b. *Возвращенное значение функции:*
    - i. Значение производной f;
- 3) void z1 – функция метода половинного деления;
  - a. *Входные переменные:*
    - i. int k – коэффициент перед  $\cos(x)$ ;
    - ii. double eps - отвечает за точность измерений;
  - b. *Локальные переменные:*
    - i. double a – левая граница;
    - ii. double b – правая граница;
    - iii. double c – середина отрезка (a, b);
    - iv. int count – количество итераций;
- 4) void z2 – функция метода Ньютона;
  - a. *Входные переменные:*
    - i. int k – коэффициент перед  $\cos(x)$ ;
    - ii. double eps - отвечает за точность измерений;
  - b. *Локальные переменные:*
    - i. double a – начальное значение;
    - ii. int count – количество итераций;
- 5) void z3 – функция метода простой итерации;
  - a. *Входные переменные:*
    - i. int k – коэффициент перед  $\cos(x)$ ;
    - ii. double eps - отвечает за точность измерений;
  - b. *Локальные переменные:*
    - i. double a – начальное значение;
    - ii. int count – количество итераций;









## Текст программы

```
#include <iostream>
#include <cmath>
#include <iomanip>

using namespace std;

double f(int, double);
double fp(int, double);
void z1(int, double);
void z2(int, double);
void z3(int, double);

int main()
{
    double eps;
    int k;
    setlocale(0, "RUSSIAN");
    for (int i = 0; i < 3; i++)
    {
        switch (i)
        {
            case 0: k = 1; eps = 0.000001;
                    cout << "k = " << k << endl << "eps = " << fixed
                    << setprecision(6) << eps << endl;
                    z1(k, eps); z2(k, eps); z3(k, eps);
                    break;
            case 1: k = 1; eps = 0.000000001;
                    cout << "k = " << k << endl << "eps = " << fixed
                    << setprecision(8) << eps << endl;
                    z1(k, eps); z2(k, eps); z3(k, eps);
                    break;
            case 2: k = 10; eps = 0.000001;
                    cout << "k = " << k << endl << "eps = " << fixed
                    << setprecision(6) << eps << endl;
                    z1(k, eps); z2(k, eps); z3(k, eps);
                    break;
        }
    }
    return 0;
}

double f(int k, double x)
{
    return x - k * cos(x);
}

double fp(int k, double x)
{
    return 1 + k * sin(x);
}
```

```

void z1(int k, double eps)
{
    int count = 0;
    double a = -10, b = 10, c;
    while (abs(a - b) > eps)
    {
        c = (a + b) / 2;
        if (f(k, a) * f(k, c) < 0) b = c;
        else a = c;
        count++;
    }
    cout << "Метод половинного деления " << fixed << setw(11) << setprecision(8)
    << c << ", итераций " << count << endl;
}




void z2(int k, double eps)
{
    int count = 0;
    double a = 5;
    while (abs(f(k, a)) > eps)
    {
        a = a - f(k, a) / fp(k, a);
        count++;
    }
    cout << "Метод Ньютона " << fixed << setw(23) << setprecision(8) << a << ",
    итераций " << count << endl;
}

void z3(int k, double eps)
{
    int count = 0;
    double a = 5;
    while (abs(f(k, a)) > eps)
    {
        a = k * cos(a);
        count++;
    }
    cout << "Метод простой итерации " << fixed << setw(14) << setprecision(8) <<
    a << ", итераций " << count << endl;
}

```



## Анализ результатов

№	Входные данные	Полученный результат
1	k = 1 eps = 0.000001	<div>  Консоль отладки Microsoft Visual Studio         </div> <div>           k=1            eps=0.000001            Метод половинного деления      0.739085, итераций 24            Метод Ньютона                      0.739085, итераций 3            Метод простой итерации          0.739085, итераций 35         </div>
2	k = 1 eps = 0.00000001	<div>  Консоль отладки Microsoft Visual Studio         </div> <div>           k=1            eps=0.00000001            Метод половинного деления    0.73908512, итераций 30            Метод Ньютона                    0.73908513, итераций 3            Метод простой итерации        0.73908513, итераций 47         </div>
3	k = 10 eps = 0.000001	<div>  Выбрать Консоль отладки Microsoft Visual ...         </div> <div>           k=10            eps=0.000001            Метод половинного деления    -9.678884, итераций 21            Метод Ньютона                    -9.678884, итераций 4            Метод простой итерации        -9.678884, итераций 3732747         </div>

**Вывод:** Мы изучили 3 метода нахождения корней нелинейного уравнения. При  $k = 10$  у нас получается несколько корней, и методом простой итерации необходимо очень большое количество итераций чтобы найти значение уравнения.