Федеральное агентство по образованию РФ
Новгородский государственный университет им. Ярослава Мудрого

Кафедра ИТИС

Реализация алгоритмов поиска корней уравнений методом половинного деления и методом хорд

Выполнил студент группы 9311
Лопатин А.С.

Великий Новгород, 2009

1. Цель работы

Цель работы — реализовать алгоритм поиска корней уравнения методом половинного деления и методом хорд на языке программирования Pascal.

2. Математическая модель решения

Даны границы a, b и точность ϵ. Необходимо найти корень уравнения (например квадратичного $y = 10x - 2x^2$ или линейного $y = 2x - 20$ уравнения) между границами, который соответствует заданной точности, т. е. значение переменной x при $|y| \leq \epsilon$ или при $|b - a| \leq \epsilon$.

Проверить наличие корней уравнения можно по формуле $F(a) \times F(b)$. Если значение отрицательное — корни есть.

Вычислять же корень можно с помощью разделения суммы границ пополам ($x = \frac{a+b}{2}$) или выразив x из уравнения прямой ($x = \frac{F(b) \times a - F(a) \times b}{F(b) - F(a)}$). После этого надо установить значение новой границы a или b равным найденному x в зависимости от $F(a) \times F(x)$ (для отрицательных — b, для остальных — a) и проверить результат на соответствие точности ϵ (определяется либо по формуле $|F(x)| \leq \epsilon$ либо через $|b - a| \leq \epsilon$). Также, для метода хорд был найден оптимальный способ проверить точность с помощью y, выраженного из уравнения прямой: $|\frac{F(b) \times a - F(a) \times b}{F(b) - F(a)} - x| \leq \epsilon$.

3. Спецификация

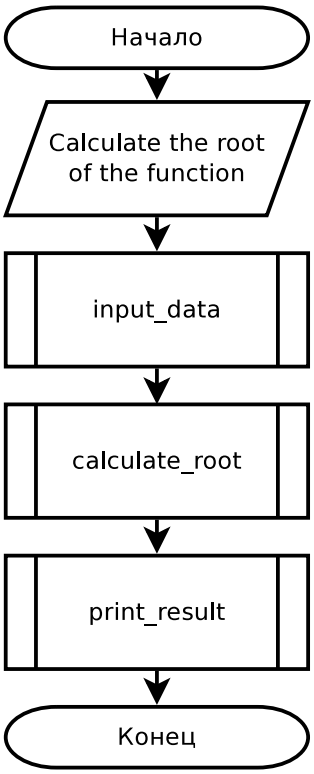
Таблица переменных:

Исходное значение	Идентификатор	Тип	Вид	Размерность	Назначение
a	a	Вещественный	Простой	–	Левая граница
b	b	Вещественный	Простой	–	Правая граница

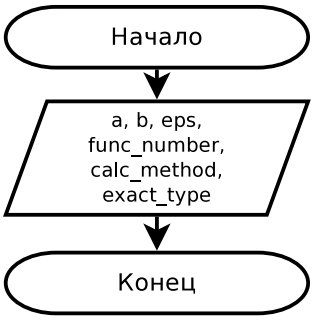
ϵ	eps	Вещественный	Простой	–	Точность
x	x	Вещественный	Простой	–	Корень уравнения
y	y	Вещественный	Простой	–	Значение функции $F(x)$
y_1	y1	Вещественный	Простой	–	Значение функции $F(a)$
y_2	y2	Вещественный	Простой	–	Значение функции $F(b)$
i	i	Целый	Простой	–	Номер итерации
function number	func_number	Целый	Простой	–	Номер функции
calculation method	calc_method	Целый	Простой	–	Метод решения
exactness type	excat_type	Целый	Простой	–	Тип точности

4. Алгоритм

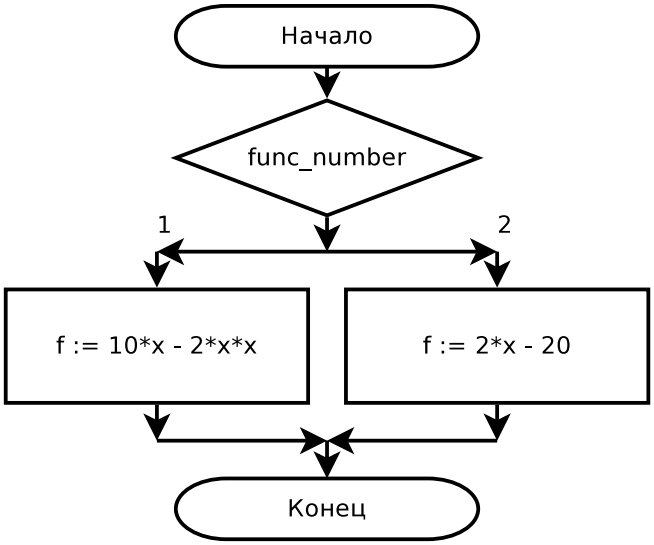
4.1. Главный модуль



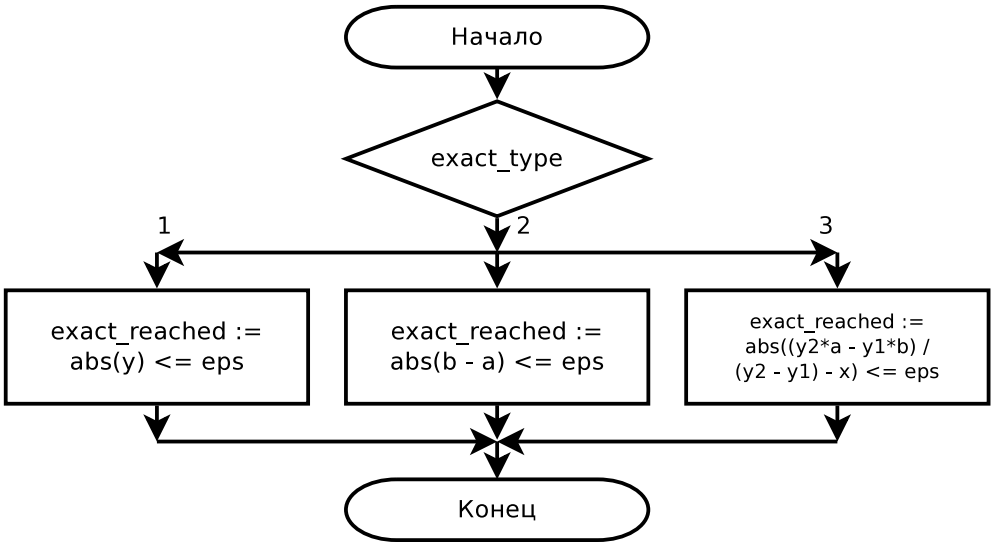
4.2. Процедура «input_data»



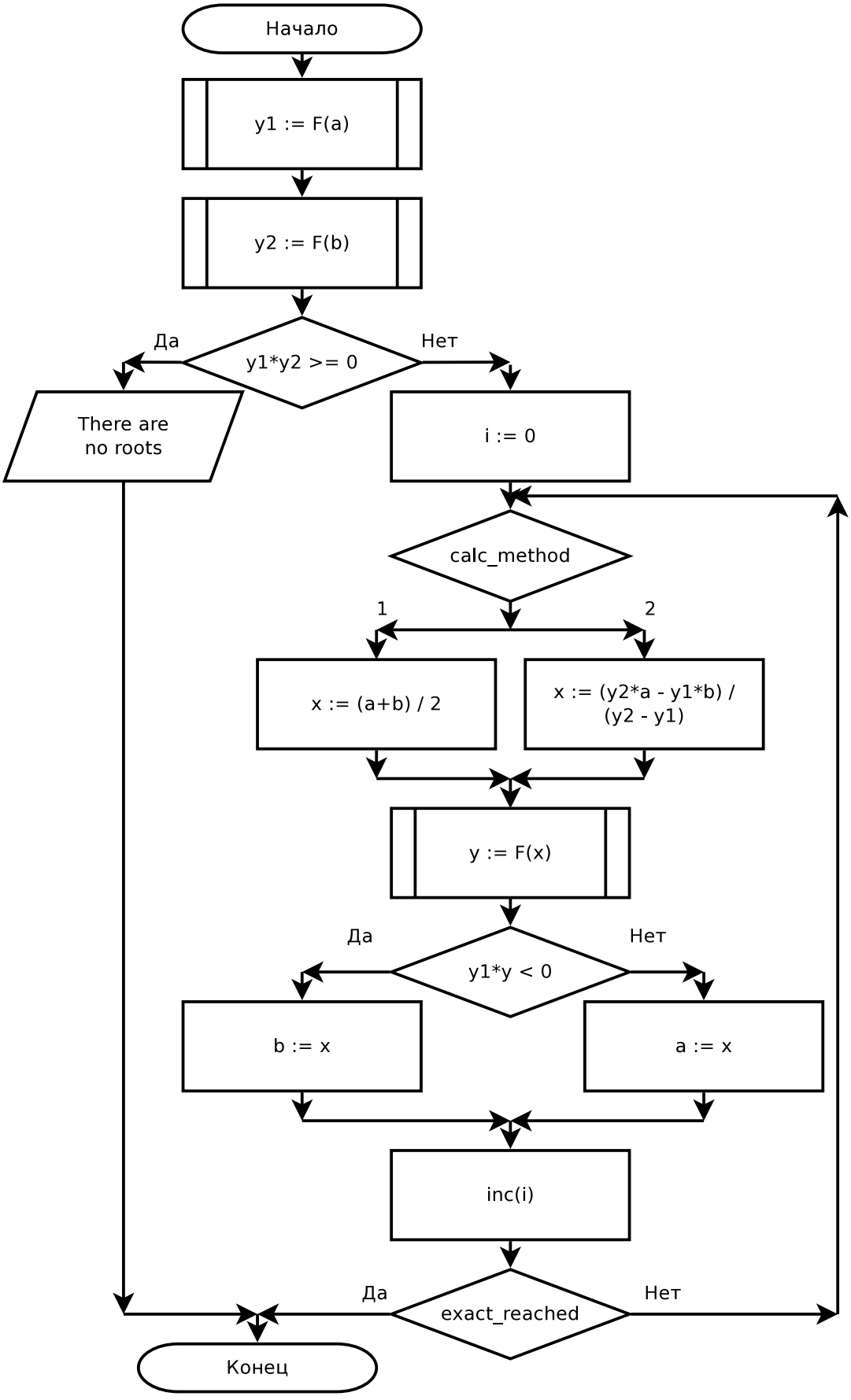
4.3. Функция «$F(x)$»



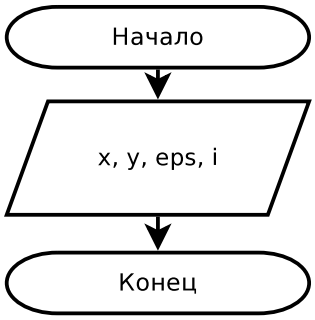
4.4. Функция «exact_reached»



4.5. Процедура «calculate_root»



4.6. Процедура «print_result»



5. Программа

```
var
    a, b, eps, x, y, y1, y2 : real;
    i : integer;
    func_number, calc_method, exact_type : byte;

procedure input_data;
begin
    write('Input a = '); readln(a);
    write('Input b = '); readln(b);
    write('Input eps = '); readln(eps);

    writeln;
    writeln('1) f(x) = 10*x - 2*x*x');
    writeln('2) f(x) = 2*x - 20');
    write('Select the function: '); readln(func_number);

    writeln;
```

```
    writeln('1) Half-division method');
    writeln('2) Chords method');
    write('Select the calculation method: '); readln(calc_method);

    writeln;
    writeln('1) |y| <= eps');
    writeln('2) |b - a| <= eps');
    writeln('3) |(y2*a - y1*b) / (y2 - y1) - x| <= eps');
    write('Select the exactness type: '); readln(exact_type);
end;

function f(x : real) : real;
begin
    case func_number of
        1: f := 10*x - 2*x*x;  { x = 5 }
        2: f := 2*x - 20;  { x = 10 }
    end;
end;

function exact_reached : boolean;
begin
    case exact_type of
        1: exact_reached := abs(y) <= eps;
        2: exact_reached := abs(b - a) <= eps;
        3: exact_reached := abs((y2*a - y1*b)/(y2 - y1) - x) <= eps;
    end;
end;
```

```
procedure calculate_root;
begin
    y1 := f(a); y2 := f(b);

    if y1*y2 >= 0 then
    begin
        writeln('There are no roots');
        halt(1);
    end;

    i := 0;
    repeat
        case calc_method of
            1: x := (a + b) / 2;
            2: x := (y2*a - y1*b) / (y2 - y1);
        end;

        y := f(x);

        if y1*y < 0 then
            b := x
        else
            a := x;
        inc(i);
    until exact_reached;
end;
```

```
procedure print_result;
begin
    writeln('x = ', x:15:15,
            ', y = ', y:15:15,
            ', eps = ', eps:15:15,
            ', i = ', i);
end;

begin
    writeln;
    writeln('Calculate the root of the function');
    writeln;

    input_data;
    calculate_root;
    print_result;
end.
```

6. Шаблон ввода исходных данных

```
1
100
0.01
1
1
1
```

7. Шаблон вывода результата

```
x = 5.000122070312500, y = -0.001220732927322, eps = 0.010000000000000,
i = 13
```

8. Вывод

После экспериментов с входными данными было выяснено следующее:

1. Наилучшую сходимость даёт метод хорд, но только если функция линейная
2. Проверять точность для метода хорд при поиске корня линейного уравнения лучше всего по формуле $F(x) \leq \epsilon$, а при поиске корня квадратичного — через y выраженный из уравнения прямой
3. Сходимость метода половинного деления примерно одинаковая на обеих функциях и всех способах проверки точности, за исключением проверки точности через y выраженный из уравнения прямой — этот способ совершенно неприменим для метода половинного деления, т. к. настоящий y, полученный с помощью вычисления функции $F(x)$ сильно отличается от этого y (однако количество итераций при этом меньше по сравнению с остальными способами проверки точности)