Пермский Национальный Исследовательский Политехнический Университет

Электротехнический факультет

Кафедра «Информационные технологии и автоматизированные системы»

I семестр

«Основы алгоритмизации и программирования»

Лабораторная работа № 4

Тема: «Решение нелинейных уравнений»

Вариант № 17

Выполнил:

студент группы РИС-20-2б

Пономарев Егор Витальевич

Проверила:

доцент кафедры ИТАС

Полякова О. А.

Пермь, 2020

**Постановка задачи**

Задана функция и интервал [a, b], где a = 0, а b = 1. Найти корень уравнения , используя три метода **(**метод Ньютона, метод половинного деления, метод итераций).

**Анализ задачи**

1. **Метод Ньютона**
2. На отрезке [a, b] задать начальное приближение **x0 = a** и точность вычисления **ε**.
3. Вычислить очередное приближение по формуле:
4. До тех пор, пока |**x0 – x1| > ε,** приравниваем **x0** к **x1**и возвращаемся к шагу 2
5. В ответе выводим значение **x1**
6. **Метод половинного деления (дихотомии)**

*#функция должна быть монотонна и непрерывна!*

1. Задать концы отрезка [a, b] и точность вычислений **ε**
2. Вычислить
3. Если f(a) \* f(c) < 0, положить b = c,

Если f(b) \* f(c) < 0, положить a = c

(Переносим границу интервала в точку c)

1. Возвращаемся к шагу 2, до тех пор, пока **| a - b| < ε**
2. В ответе выводим значение a или b.
3. **Метод итераций**
4. На интервале [a, b] произвольно берётся начальное приближение корня **x1** (допустим: (a + b) / 2)и задаётся точность **ε.**
5. Приравниваем **x0** к **x1**
6. Следующее значение корня **x1** находится по формуле:

**x1 = x0 - λ \*f(x0)**

1. До тех пор, пока |x0 – x1| > **ε,** повторяем 2 и 3 шаг
2. В ответе выводим значение **x1**

**/**\*

*Достаточное условие сходимости таково:*

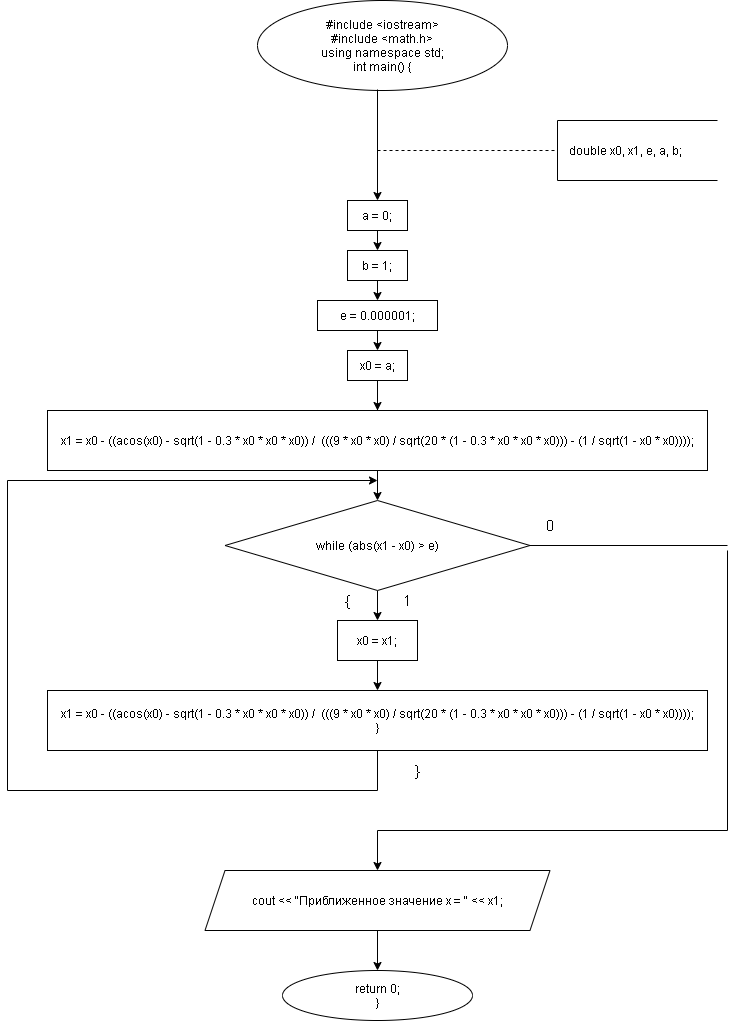
*В данном уравнении значение λ находится в интервале [-1.06332; 0]*

*Допустим, λ = -1*

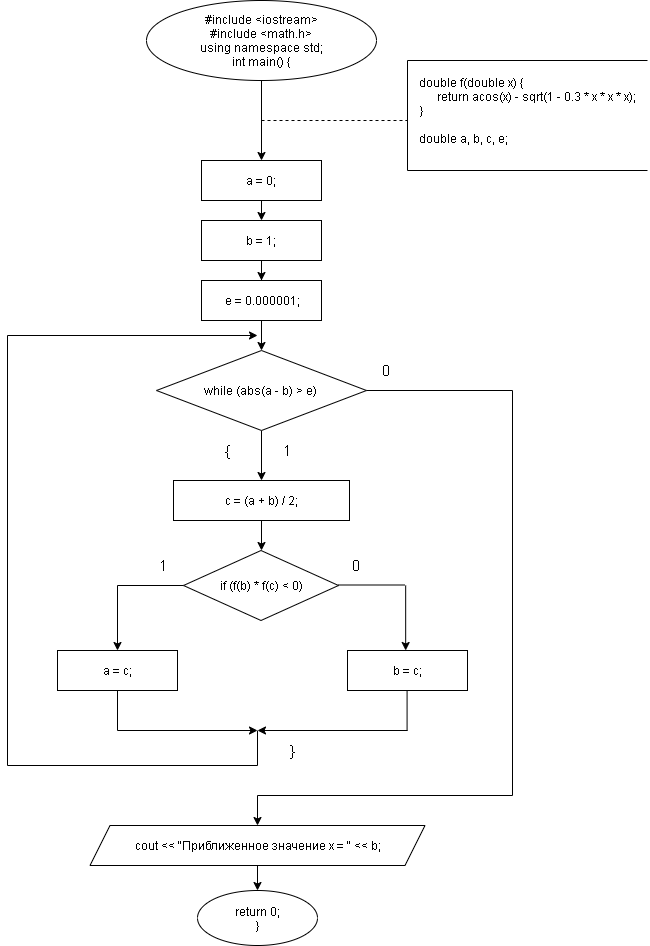
\*/

**Блок-схемы**

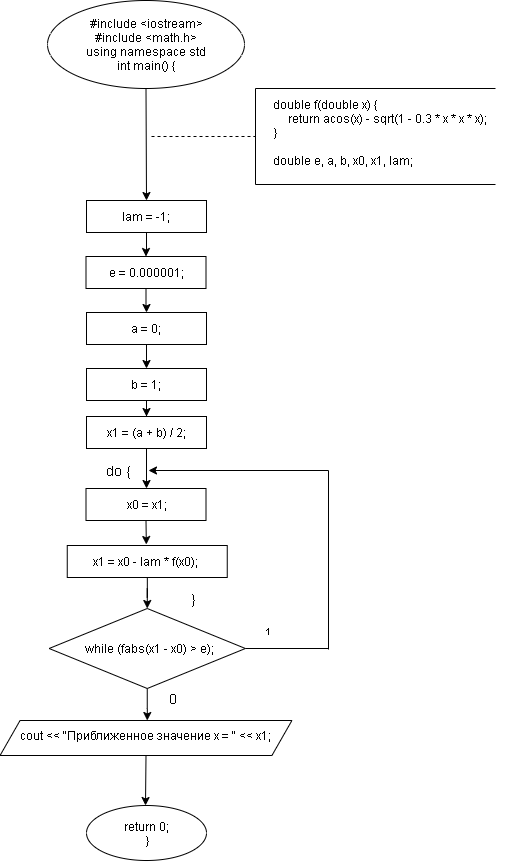
1. **Метод Ньютона**



1. **Метод половинного деления**



1. **Метод итераций**

****

**Код С++**

1. **Метод Ньютона**

#include <iostream>

#include <math.h>

using namespace std;

int main()

{

double x0, x1, e, a, b;

a = 0;

b = 1;

e = 0.000001;

x0 = a;

x1 = x0 - ((acos(x0) - sqrt(1 - 0.3 \* x0 \* x0 \* x0)) / (((9 \* x0 \* x0) / sqrt(20 \* (1 - 0.3 \* x0 \* x0 \* x0))) - (1 / sqrt(1 - x0 \* x0))));

while (abs(x1 - x0) > e)

{

x0 = x1;

x1 = x0 - ((acos(x0) - sqrt(1 - 0.3 \* x0 \* x0 \* x0)) / (((9 \* x0 \* x0) / sqrt(20 \* (1 - 0.3 \* x0 \* x0 \* x0))) - (1 / sqrt(1 - x0 \* x0))));

};

cout << x1 << endl;

return 0;

}

1. **Метод половинного деления**

#include <iostream>

#include <math.h>

using namespace std;

double f(double x)

{

return acos(x) - sqrt(1 - 0.3 \* x \* x \* x);

}

int main() {

double a, b, c, e;

a = 0;

b = 1;

e = pow(10, -6);

while (abs(a - b) >= e) {

c = (a + b) / 2;

if (f(b) \* f(c) < 0)

a = c;

else

b = c;

}

cout << b << endl;

return 0;

}

1. **Метод итераций**

#include <iostream>

#include <math.h>

using namespace std;

double f(double x)

{

return acos(x) - sqrt(1 - 0.3 \* x \* x \* x);

}

int main() {

double e, a, b, x0, x1, lam;

lam = -1;

e = 0.000001;

a = 0;

b = 1;

x1 = (a + b) / 2;

do {

x0 = x1;

x1 = x0 - lam \* f(x0);

} while (fabs(x1 - x0) > e);

cout << x1;

return 0;

}

**Результаты выполнения**

1. **Метод Ньютона**

****

1. **Метод половинного деления**

****

1. **Метод итераций**

****