

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ  
Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
«Московский Авиационный Институт»  
(Национальный Исследовательский Университет)

Институт: №8 «Информационные технологии и прикладная  
математика»  
Кафедра: 806 «Вычислительная математика и программирование»

Курсовая работа  
по курсу «Фундаментальная  
информатика»  
I семестр  
Задание 4  
«Процедуры и функции в качестве параметров»

Группа	М8О-109Б-22
Студент	Моравская В.И.
Преподаватель	Сысоев М.А.
Оценка	
Дата	

Москва, 2022

## Задание

Составить программу на Си с процедурами решения трансцендентных алгебраических уравнений различными численными методами (итераций, Ньютона и половинного деления — дихотомии). Нелинейные уравнения оформить как параметры-функции, разрешив относительно неизвестной величины в случае необходимости. Применить каждую процедуру к решению двух уравнений, заданных двумя строками таблицы, начиная с варианта с заданным номером. Если метод неприменим, дать математическое обоснование и графическую иллюстрацию, например, с использованием gnuplot.

## Вариант

6	$x + \cos(x^{0.52} + 2) = 0$	[0.5, 1]	итераций	0.9892
7	$3 \ln^2 x + 6 \ln x - 5 = 0$	[1, 3]	Ньютона	1.8832

# Теоретическая часть

## 2. Метод итераций.

Идея метода заключается в замене исходного уравнения  $F(x) = 0$  уравнением вида  $x = f(x)$ .

Достаточное условие сходимости метода:  $|f'(x)| < 1, x \in [a, b]$ . Это условие необходимо проверить перед началом решения задачи, так как функция  $f(x)$  может быть выбрана неоднозначно, причем в случае неверного выбора указанной функции метод расходится.

Начальное приближение корня:  $x^{(0)} = (a + b) / 2$  (середина исходного отрезка).

Итерационный процесс:  $x^{(k+1)} = f(x^{(k)})$ .

Условие окончания:  $|x^{(k)} - x^{(k-1)}| < \varepsilon$ .

Приближенное значение корня:  $x^* \approx x^{(\text{конечное})}$ .

## 3. Метод Ньютона.

Метод Ньютона является частным случаем метода итераций.

Условие сходимости метода:  $|F(x) \cdot F''(x)| < (F'(x))^2$  на отрезке  $[a, b]$ .

Итерационный процесс:  $x^{(k+1)} = x^{(k)} - F(x^{(k)}) / F'(x^{(k)})$ .

Более совершенное с программистской точки зрения решение задачи может быть получено с помощью изучаемого в курсе «Языки программирования» (II семестр) процедурного типа данных. В этом случае

---

различные уравнения и методы как переменные процедурного типа подставляются в качестве фактических параметров соответствующих подпрограмм. Решение задачи на языке Си, фактически базирующееся на указателях на функции, близко к этому.

## Алгоритм решения

В целом, методы решения обоими методами довольно похожи. Для метода итераций надо преобразовать уравнение вида  $x = f(x)$ , а затем найти для этого уравнения производную. Для метода же Ньютона преобразовывать уравнение не надо, но нужно найти производные первого и второго порядка. Для каждого из методов прописываем команду проверки на сходимость, а затем и сам способ нахождения корня (условия прописаны в самом задании). Ну и выводим результаты.

## Используемые переменные

Название переменной	Тип переменной	Смысл переменной
check_I	int	Проверка на сходимость метода итерации
check_N	int	Проверка на сходимость метода Ньютона
step	long double	Шаг при проверке на сходимость
x0	long double	Временная переменная для хранения значения $x$ при расчете методом
x1	long double	Результат работы методов
a	long double	Начало отрезка
b	long double	Конец отрезка

## Код программы

```
#include <stdio.h>
#include <float.h>
#include <math.h>

long double var6(long double x) {
    return (-cosl(powl(x,0.52)+2));
}

long double var6_der(long double x) {
    return ( (13*sinl(powl(x,0.52)+2)) / 25 / powl(x,0.52) );
}

int var6_conv(long double a, long double b) {
    int check_I = 1;
    long double step = (b - a) / 100;
    for (long double x = a; x <= b; x += step) {
        if (var6_der(x) >= 1) {
            check_I = 0;
        }
    }
    return check_I;
}

long double Iteration(long double a, long double b) {
    long double x0 = (a + b) / 2;
    long double x1 = var6(x0);
    while (fabsl(x1 - x0) >= LDBL_EPSILON) {
        x0 = x1;
        x1 = var6(x0);
    }
    return x1;
}

long double var7(long double x) {
    return ( 3*logl(x)*logl(x) + 6*logl(x) - 5 );
}
```

```

long double var7_der1(long double x) {
    return ( (6*log1(x)+6) / x );
}

long double var7_der2(long double x) {
    return ( (-6*log1(x))/x/x );
}

int var7_conv(long double a, long double b) {
    int check_N = 1;
    long double step = (b - a) / 100;
    for (long double x = a; x <= b; x += step) {
        if (fabs1(var7(x) * var7_der2(x)) >= pow1(var7_der1(x), 2)) {
            check_N = 0;
        }
    }
    return check_N;
}

long double Newton(long double a, long double b) {
    long double x0 = (a + b) / 2;
    long double x1 = x0 - var7(x0) / var7_der1(x0);
    while (fabs1(x1 - x0) >= LDBL_EPSILON) {
        x0 = x1;
        x1 = x0 - var7(x0) / var7_der1(x0);
    }
    return x1;
}

int main() {
    long double a = 0.5, b = 1;
    printf("Variant 6 (Iteration):\t x+cos(x^0.52 + 2) \n");
    if (var6_conv(a, b)) {
        printf("Convergent:\tYes\n");

        long double root = Iteration(a, b);
    }
}

```

```

printf("Root:\t%.10Lf\n", root);

long double result = root + cosl(powl(root,0.52)+2);
printf("Verify:\t%.10Lf\n\n", result);
} else {
    printf("Convergent:\tNo\n\n");
}

a = 1;
b = 3;
printf("Variant 7 (Newton):\t 3(ln(x))^2 + 6ln(x) - 5 \n");
if (var7_conv(a, b)) {
    printf("Convergent:\tYes\n");

    long double root = Newton(a, b);
    printf("Root:\t%.10Lf\n", root);

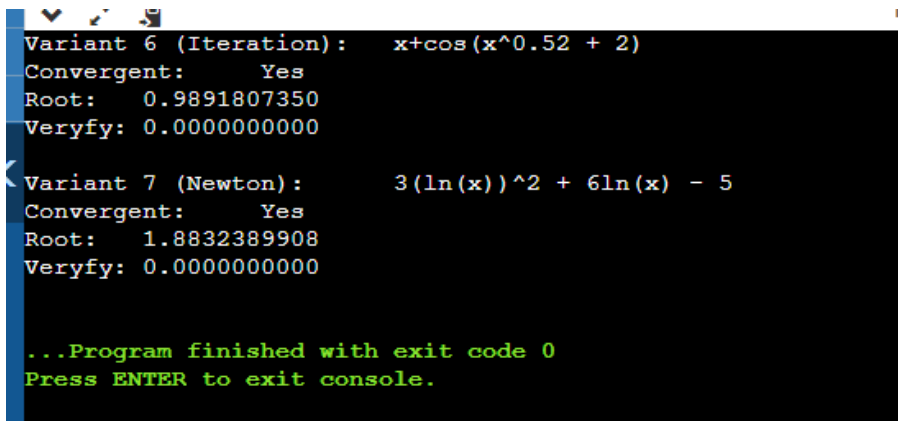
    long double result = var7(root);
    printf("Verify:\t%.10Lf\n", result);
} else {
    printf("Convergent:\tNo\n\n");
}
}

```

## Входные данные

Отсутствуют

## Выходные данные



```

Variant 6 (Iteration):  x+cos(x^0.52 + 2)
Convergent:      Yes
Root:   0.9891807350
Verify: 0.0000000000

Variant 7 (Newton):    3(ln(x))^2 + 6ln(x) - 5
Convergent:      Yes
Root:   1.8832389908
Verify: 0.0000000000

...Program finished with exit code 0
Press ENTER to exit console.

```

## **Вывод**

Благодаря этому заданию я глубже изучила язык программирования Си, улучшила свои знания в области методов решения трансцендентных алгебраических уравнений.

Также благодаря этой работе, я поняла, что Code::Blocks – довольно плохая программа-компилятор, ибо он отказывался правильно запускать мою программу, хотя в онлайн компиляторе все сработало нормально. Я потратила из-за этого пару часов, ища несуществующую ошибку. Но это многому меня научило.