**Курсовой проект №4**

Студент группы М8О-101Б-22 Анциборко Л. В., № по списку: 1.

Контакты: anciborko04@mail.ru

Преподаватель: Сахарин Н. А.., каф. 806.

Входной контроль знаний с оценкой:

Итоговая оценка:

Подпись преподавателя:

Оглавление

[*Задание* 2](file:///C:\Users\leoan\Downloads\Курсовой%20проект%20№4.docx#_Toc122879978)

[*Вариант* 2](file:///C:\Users\leoan\Downloads\Курсовой%20проект%20№4.docx#_Toc122879979)

[*Алгоритм* 2](file:///C:\Users\leoan\Downloads\Курсовой%20проект%20№4.docx#_Toc122879980)

[*Код* 3](file:///C:\Users\leoan\Downloads\Курсовой%20проект%20№4.docx#_Toc122879981)

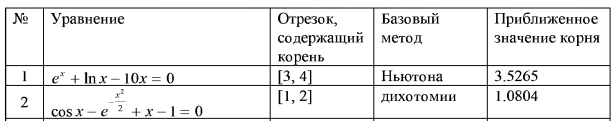
[*Примеры работы программы* 3](file:///C:\Users\leoan\Downloads\Курсовой%20проект%20№4.docx#_Toc122879982)

[*Вывод* 3](file:///C:\Users\leoan\Downloads\Курсовой%20проект%20№4.docx#_Toc122879983)

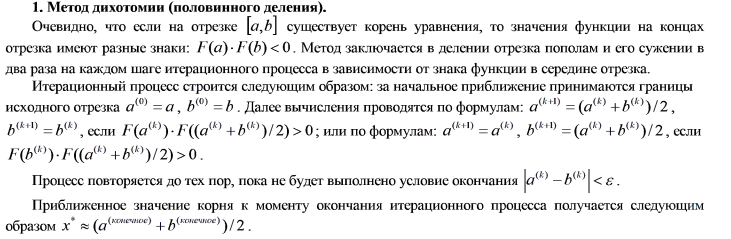
# *Задание*

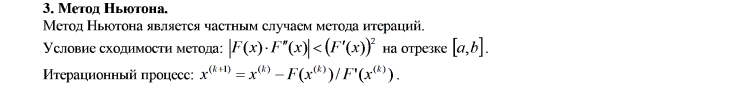
Составить программу на языке Си с процедурами решения трансцендентных алгебраических уравнений различными методами (итераций, Ньютона, дихотомии). Нелинейные уравнения оформить как параметры-функции, разрешив относительно неизвестной величины при необходимости.

# *Вариант*



# *Алгоритм*





# *Код*

#include <stdio.h>

#include <math.h>

typedef double dbl;

dbl epsilon() {

    dbl eps = 1.0;

    while (1 + eps / 2.0 != 1)

        eps /= 2.0;

    return eps;

}

dbl F1(dbl x) {

    return exp(x) + log(x) - 10\*x;

}

dbl F1\_ex\_x(dbl x) {

    return (exp(x) + log(x)) / 10;

}

dbl F1\_ex\_x\_1der(dbl x) {

    return exp(x)/10 + 1/(10\*x);

}

dbl F1\_1der(dbl x) {

    return exp(x) + 1/x - 10;

}

dbl F1\_2der(dbl x) {

    return exp(x) - 1/x\*x;

}

dbl F2(dbl x) {

    return cos(x) - exp(-pow(x, 2) / 2) + x - 1;

}

dbl F2\_ex\_x(dbl x) {

    return 1 + exp(-pow(x, 2) / 2) - cos(x);

}

dbl F2\_ex\_x\_1der(dbl x) {

    return sin(x) - x \* exp(-pow(x, 2) / 2);

}

dbl F2\_1der(dbl x) {

    return -sin(x) + x \* exp(-pow(x, 2) / 2) + 1;

}

dbl F2\_2der(dbl x) {

    return -cos(x) + exp(-pow(x, 2) / 2) - pow(x,2) \* exp(-pow(x, 2) / 2);

}

dbl dich(dbl (\*F)(dbl), dbl a, dbl b, dbl relative\_eps, dbl abs\_eps) {

    dbl x = (a + b) / 2;

    if (F(a) \* F(b) < 0){

        while (fabs(a - b) > fmax(relative\_eps \* fabs(x), abs\_eps)) {

            x = (a + b) / 2;

            if (F(x) \* F(a) < 0) {

                b = x;

            }

            else {

                a = x;

            }

        }

        return x;

    }

    else {

        return NAN;

    }

}

dbl iter(dbl (\*F\_x)(dbl), dbl (\*F\_x\_1der)(dbl), dbl a, dbl b, dbl relative\_eps, dbl abs\_eps) {

    dbl x = (a + b) / 2;

    if (fabs(F\_x\_1der(x)) < 1) {

        while (fabs(F\_x(x) - x) >= fmax(relative\_eps \* fabs(x), abs\_eps)) {

            x = F\_x(x);

        }

        return x;

    }

    else {

        return NAN;

    }

}

dbl newton(dbl (\*F)(dbl), dbl (\*F\_1der)(dbl), dbl (\*F\_2der)(dbl), dbl a, dbl b, dbl relative\_eps, dbl abs\_eps) {

    dbl x = (a + b / 2);

    if (fabs(F(x) \* F\_2der(x)) < (F\_1der(x) \* F\_1der(x))) {

        while (fabs(F(x) / F\_1der(x)) > fmax(relative\_eps \* fabs(x), abs\_eps)) {

            x -= F(x) / F\_1der(x);

        }

        return x;

    }

    else {

        return NAN;

    }

}

void result(dbl d, dbl i, dbl n) {

   if (d != NAN) {

     printf("The root obtained by the dich method: %.10f\n", d);

   }

   else {

     printf("The dechotomy method isn't suitable\n");

   }

   if (i != NAN) {

     printf("The root obtained by the iter method: %.10f\n", i);

   }

   else {

     printf("The iter method isn't suitable\n");

   }

   if (n != NAN) {

     printf("The root obtained by the Newton's method: %.10f\n", n);

   }

   else {

     printf("The Newton's method isn't suitable\n");

   }

}

int main() {

    dbl abs\_eps = epsilon();

    dbl rel\_eps = sqrt(abs\_eps);

    printf("Machine epsilon for long double = %.16e\n", abs\_eps);

    dbl a1 = 3, b1 = 4;

    dbl a2 = 1, b2 = 2;

    dbl d1 = dich(F1, a1, b1, rel\_eps, abs\_eps);

    dbl i1 = iter(F1\_ex\_x, F1\_ex\_x\_1der, a1, b1, rel\_eps, abs\_eps);

    dbl n1 =newton(F1, F1\_1der, F1\_2der, a1, b1, rel\_eps, abs\_eps);

    printf("Function exp(x) + ln(x) - 10x\n");

    result(d1, i1, n1);

    printf("\n");

    dbl d2 = dich(F2, a2, b2, rel\_eps, abs\_eps);

    dbl i2 = iter(F2\_ex\_x, F2\_ex\_x\_1der, a2, b2, rel\_eps, abs\_eps);

    dbl n2 = newton(F2, F2\_1der, F2\_2der, a2, b2, rel\_eps, abs\_eps);

    printf("Function cos(x) - exp(-0.5\*x^2) + x - 1\n");

    result(d2, i2, n2);

    return 0;

}

# *Примеры работы программы*

# *Вывод*

Мной было освоено написание функций на языке С, вычисляющих корень уравнения.