

С# ДА АМАЛИЙ МАСАЛАЛАРНИ ДАСТУРЛАШТИРИШ ТАДБИҚЛАРИ**Файзуллаев Шерали Бахтиёр ўғли**

Термез давлат университет Ахборот технологиялари факультети талабаси

Файзуллаев Самандарбек Бахтиёр ўғли

Термез давлат университет Кимё факультети талабаси

<https://doi.org/10.5281/zenodo.7105838>

Аннотация. Маълумки, С#да программа тузиш учун маълум бир кўникмалар талаб қилинади. Бу С# алфавитидан бошлаб, то охирги маърузуларгача давом этадиган назарий ва амалий машгулотлардир. Дастлабки тушунчалардан энг асосийси-бу С# да сонларнинг (бутун, ҳақиқий ва экспоненциал) ёзилиши тўлиқ ўрганилган бўлиши талаб қилинади. Сўнгра ўзгармас, ўзгарувчи, тамга тушунчалари, индексли ўзгарувчилар, массив тушунчаси, асосий математик функцияларнинг ёзилиши асосида ифодаларни ёзишни ўрганиши мумкин. Навбатдагиси, бу асосий тасвирлаш операторлари- бутун, ҳақиқий ва символли, ўзлаштириши, киритиши, чиқариши, шартли ва шартсиз ўтиши, такрорлаш операторлари бўлиб ҳисобланади. Фақат кейинчалик функция қисм программаси, процедура қисм программаси каби мавзуларни ўзлаштириши талаб қилинади. Амалий машгулотлардаги программалаштириши элементлари мана шундай мавзуларни қамраб олиб, оддийдан мураккабга томон олиб борилиши лозим. Тайёрланган прогаммаларни 2 хил режимда бажариши мумкин. Шулардан биттаси -Console режимида ишлашдир. Принцип жиҳатидан бу Паскальга ўхшаб кетади. Лекин, имкониятлари бўйича бу тил анча устун туради. Шу боисдан ҳам ҳар хил соҳалардан олинган оддий мисолларнинг қисқача алгоритмлари ва программаларини тўплаш мақсадга мувофиқдир. Таклиф қилинаётган мазкур ишда мисолларнинг алгоритмлари ва программалари келтирилиши билан бир қаторда, программаларнинг компьютердаги бажарилишлари учун қисқача кўрсатма ҳам келтирилади. Булар асосан-интегрални бирор тақрибий формула билан ҳисоблаш, матрицаларнинг симметрикларини текшириш, матрицаларнинг кўпайтмаларини ҳисоблаш, тажриба натижаларини қайта ишлаш, итерация усули ва оралиқни иккига бўлиш ёрдамида трансцендент тенламаларни ечиш, тенламалар системасини Крамер усули билан ечиш кабилардир. Буларнинг олиншига сабаб шуки, бундай мисоллар ихтиёрий табиий йўналишидаги соҳаларда учраб туради ва уларни бирор алгоритмик тилда ечиш доим биринчи планда туради. Мисоллар иложи борича содда йўлларда ечилган ва компьютерда бажариши кўрсатмалари ҳам келтирилган.

Калит сўзлар: аниқ интеграл, дисперсия, System, функция, массив, матрица.

ПРОГРАММИРОВАНИЕ ПРИЛОЖЕНИЙ ПРАКТИЧЕСКИХ ЗАДАЧ НА С#

Аннотация. Как известно, программирование на С# требует определенных навыков. Это серия теоретических и практических упражнений, начиная с алфавита С# и заканчивая последними лекциями. Самая основная из первоначальных концепций заключается в том, что вы должны иметь полное представление о том, как записывать числа (целые, действительные и экспоненциальные) в С#. Затем вы можете научиться писать выражения на основе констант, переменных, символьных понятий, индексных переменных, понятий массива, писать основные математические функции. Далее, это основные операторы представления - целочисленные, вещественные и символьные, присваивания, ввода, вывода, условного и безусловного перехода, операторы итерации. Только позже необходимо изучить такие темы, как программа части функции,

программа части процедуры. Элементы программирования в практических занятиях должны охватывать такие темы и двигаться от простого к сложному. Подготовленные программы могут выполняться в 2 различных режимах. Один из них — запустить в режиме -Console. В принципе он похож на Паскаль. Однако с точки зрения возможностей этот язык намного превосходит его. Поэтому желательно собирать короткие алгоритмы и программы на простых примерах из разных областей. В предлагаемой работе, помимо предоставления алгоритмов и программ примеров, также приводится краткая инструкция по выполнению программ на компьютере. В основном это вычисление интеграла по приближенной формуле, проверка симметрий матриц, вычисление произведения матриц, обработка результатов эксперимента, решение трансцендентных уравнений методом итераций и деления интервала на два, решение системы уравнений по методу Крамера. Они взяты потому, что такие примеры встречаются в полях произвольного естественного направления, и их решение на алгоритмическом языке всегда находится на первом плане. Примеры решены максимально просто, а также даны инструкции для компьютерной реализации.

Ключевые слова: определенный интеграл, дисперсия, система, функция, массив, матрица.

PROGRAMMING PRACTICE APPLICATIONS IN C#

Abstract. As you know, programming in C# requires certain skills. This is a series of theoretical and practical exercises, starting with the C# alphabet and ending with the last lectures. The most basic of the initial concepts is that you should have a complete understanding of how to write numbers (integer, real, and exponential) in C#. Then you can learn how to write expressions based on constants, variables, symbolic concepts, index variables, array concepts, write basic mathematical functions. Further, these are the main representation operators - integer, real and symbolic, assignment, input, output, conditional and unconditional transition, iteration operators. Only later it is necessary to study such topics as the program of the part of the function, the program of the part of the procedure. Elements of programming in practical classes should cover such topics and move from simple to complex. Prepared programs can be run in 2 different modes. One of them is to run in -Console mode. In principle, it is similar to Pascal. However, in terms of features, this language is far superior to it. Therefore, it is desirable to assemble short algorithms and programs based on simple examples from different areas. The proposed work, in addition to providing algorithms and example programs, also provides a brief instruction on how to run programs on a computer. Basically, this is the calculation of the integral using an approximate formula, checking the symmetries of matrices, calculating the product of matrices, processing the results of the experiment, solving transcendental equations by iteration and dividing the interval by two, solving the system of equations using the Cramer method. They are taken because such examples occur in fields of arbitrary natural direction, and their solution in the algorithmic language is always in the foreground. The examples are solved as simply as possible, and instructions for computer implementation are given.

Keywords: definite integral, dispersion, system, function, array, matrix.

КИРИШ

Аниқ интегрални ҳисоблаш. Маълумки,

$$S = \int_a^b f(x)dx = \frac{b-a}{n-1} \left[\frac{f(a)+f(b)}{2} + \sum_{i=1}^{n-2} f(a+i*h) \right]$$

– трапеция формуласи ёрдамида амалга оширилади. Бу ерда $h = \frac{b-a}{n-1}$, a ва b - интегралнинг қўйи ва юқори чегалари бўлиб, n – эса оралиқнинг бўлиниш тугун нуқталари сонини ифодалайди улрнинг қийматлари берилади. Масалан, $a=0$, $b=1$ ва $n=11$ бўлсин.

ТАДҚИҚОТ МАТЕРИАЛЛАРИ ВА МЕТОДОЛОГИЯСИ

Ушбу мисолнинг трапеция усули орқали ҳисоблаш программаси қуйидагича бўлади.

using System;

/* Инт остидаги функция кўриниши */

class **inthis** {

static Double func(Double x){

Double rez=x*x*x;

return (rez);

}

/* Асосий программа */

public static void Main()

{ Console.WriteLine(“ C# да 1-пр. ”);

Double a,b,s,l,h; int i,n;

Console.WriteLine(“ Киритинг a,b= ”);

/* a,b,n- лар диалог режимда экрандан киритиш ва конвертация қилиш */

a=Convert.ToDouble(Console.ReadLine());

b=Convert.ToDouble(Console.ReadLine());

Console.WriteLine(“ n= ”);

n=Convert.ToInt32(Console.ReadLine());

h=(b-a)/(n-1); /* бўлиниш қадами */

s1=0; /* Йиғинди ҳосил қилиш учун */

for (i=1; i<n-1; i++) s1=s1+func(a+i*h);

s=h*((func(a)+ func(b))/2+s1);

Console.WriteLine(); /* бўш сатр ташлаш */

Console.WriteLine(“ Трапеция усули бўйича инт қиймати s= ”+s);

Console.WriteLine(“Инт аниқ қиймати s= ”+0.25);

Console.WriteLine();

Console.ReadLine(); /* диалог режимда кутиш учун */

}

}

Бу программани бажаришга бериш учун C# ойнасидаги *Debug --> Start Debugging F5* лар кетма кет босилади. Натижада қора ойнада

C# да 2-пр

Киритинг a,b=

чиқади. Шунда,

0 Enter

Enter босилади. Энди экранда

Трапеция усули бўйича инт қиймати $s = 0,253086$

Инт аниқ қиймати $s = 0,25$

чиқади.

ТАДҚИҚОТ НАТИЖАЛАРИ

1. Берилган $A(n \times m)$ матрицанинг симметриклигини текшириш программаси

Фараз қилайлик,

$$A = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1m} \\ a_{21} & a_{22} & \dots & a_{2m} \\ \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{nm} \end{pmatrix}$$

Берилган бўлсин. A симметрик дейлади, агарда $a_{ij} = a_{ji}$ бажарилса ($i = \overline{1, n}; j = \overline{1, m}$). Программаси қуйидагича бўлади:

```
using System;
/* Асосий программа */
class Simmatr
{
    public static void Main()
    {
        Console.WriteLine("C# да 2-пр. Матрицанинг симметрик-ни текшириш ");
        int n,m,i;
        Console.WriteLine("Матрицанинг ўлчамини киритинг n,m= ");
        n=Convert.ToInt32(Console.ReadLine());
        m=Convert.ToInt32(Console.ReadLine());
        Double [, ] a=new Double[n,m]; /* А массивни эълон қилиш */
        Console.WriteLine(" a11, a12,,, a1m ларни киритинг=");
        /* aij - ларни экрандан киритиш ва конвертация қилиш */
        for(i=1; i<n; i++)
        {
            for(j=1; j<m; j++)
                a[i,j]=Convert.ToDouble(Console.ReadLine());
        }
        k=0;
        for(i=1; i<m; i++)
        {
            for(j=1; j<l; j++)
                if( a[i,j]/== a[j,i]) k=1;
        }
        if( k==0) Console.WriteLine("Матрицанинг симметрик ");
        else Console.WriteLine("Матрица симметрик эмас");
        /* Матрицани чоп этиш */
        for(i=1; i<n; i++)
        {
            Console.WriteLine( );
        }
    }
}
```

```

for(j=1; j<l; j++)
    Console.WriteLine( a[i,j]);
}
Console.ReadLine(); /* диалог режимида кутиш учун */
}
} /* 5-пр охири */

```

Бу программани бажаришга бериш учун C# ойнасидаги Debug --> Start Debugging F5 лар кетма кет босилади. Натижасида қора ойнада

C# да 2-пр. Матрицанинг симметрик-ни текшириш

Деган ёзув чиқади ва n, m ларнинг қийматларини қиритиш сўралади:

Матрицанинг ўлчамини киритинг n, m=

буларнинг мос қийматларини киритгандан кейин эса

$a_{11}, a_{12}, \dots, a_{1m}$ —ларни киритинг

деган хабар чиқади. Шунда, A матрицаларнинг элементлари сатр бўйича киритилади.

Натижасида экранда текшириш натижаси k га боғлиқ ҳолда

Матрицанинг симметрик

ёки

Матрица симметрик эмас

деган хабар босиб чиқарилади. Сўнгра A матрицанинг $a[i,j]$ элементлари чоп этилади.

2. A (n x m) ва B (m x l) матрицалар кўпайтмасини ҳисобланг.

Фараз қилайлик,

$$A = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1m} \\ a_{21} & a_{22} & \dots & a_{2m} \\ \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{nm} \end{pmatrix}, \quad B = \begin{pmatrix} b_{11} & b_{12} & \dots & b_{1l} \\ b_{21} & b_{22} & \dots & b_{2l} \\ \dots & \dots & \dots & \dots \\ b_{m1} & b_{m2} & \dots & b_{ml} \end{pmatrix}.$$

$$\begin{aligned}
 \text{Шунда } C = AB &= \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1m} \\ a_{21} & a_{22} & \dots & a_{2m} \\ \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{nm} \end{pmatrix} \begin{pmatrix} b_{11} & b_{12} & \dots & b_{1l} \\ b_{21} & b_{22} & \dots & b_{2l} \\ \dots & \dots & \dots & \dots \\ b_{m1} & b_{m2} & \dots & b_{ml} \end{pmatrix} = \\
 &= \begin{pmatrix} \sum_{j=1}^m a_{1j}b_{j1} & \sum_{j=1}^m a_{1j}b_{j2} & \dots & \sum_{j=1}^m a_{1j}b_{jl} \\ \sum_{j=1}^m a_{2j}b_{j1} & \sum_{j=1}^m a_{2j}b_{j2} & \dots & \sum_{j=1}^m a_{2j}b_{jl} \\ \dots & \dots & \dots & \dots \\ \sum_{j=1}^m a_{nj}b_{j1} & \sum_{j=1}^m a_{nj}b_{j2} & \dots & \sum_{j=1}^m a_{nj}b_{jl} \end{pmatrix}
 \end{aligned}$$

кўпайтмани ҳисоблашга тузилган программа қуйидагича бўлади:

```

using System;
/* Асосий программа */
class Matkop
{
    public static void Main()
    {

```

```

Console.WriteLine(" C# да 3-пр. Матрицалар кўпайтмасини ҳисоблаш ");
int n,m,l;
n=Convert.ToInt32(Console.ReadLine());
m=Convert.ToInt32(Console.ReadLine());
l=Convert.ToInt32(Console.ReadLine());
Double [, ] a=new Double[n,m]; /* A массивни эълон қилиш */
Double [, ] b=new Double[m,l]; /* B массивни эълон қилиш */
Double [, ] c=new Double[n,l]; /* C массивни эълон қилиш */
Console.WriteLine(" a11, a12, , a1m ларни киритинг=");
/* aij, bij - ларни экрандан киритиш ва конвертация қилиш */
for(i=1; i<n; i++) {
    for(j=1; j<m; j++)
        a[i,j]=Convert.ToDouble(Console.ReadLine());
}
Console.WriteLine(" b11, b12, , b1l – ларни киритинг=");
for(i=1; i<m; i++) {
    for(j=1; j<l; j++)
        b[i,j]=Convert.ToDouble(Console.ReadLine());
}
/* Кўпайтмани ҳисоблаш қисми */
for(i=1; i<m; i++) {
    for(j=1; j<l; j++) {
        c[i,j]=0;
        for(k=1; k<n; k++) {
            c[i,j]= c[i,j]+ a[i,k]* b[k,j];
        }
    }
}
/* Кўпайтмани чоп этиш */
for(i=1; i<n; i++) {
    Console.WriteLine( );
    for(j=1; j<l; j++) {
        Console.WriteLine( c[i,j]);
    }
}
Console.ReadLine(); /* диалог режимида кутиш учун */
}
} /* 3-пр охири */

```

Бу программани бажаришига бериш учун C# ойнасидаги Debug --> Start Debugging F5 лар кетма кет босилади. Натижада қора ойнада

C# да 3-пр Матрицалар кўпайтмасини ҳисоблаш

Деган ёзув чиқади ва n,m,l ларнинг қийматларини қиритиш сўралади. Кейин эса

a₁₁, a₁₂, , , a_{1m} –ларни киритинг

деган хабар чиқади. Шунда, A ва сўнгра B матрицаларнинг элементлари сатр бўйича киритилади.

Натижада экранда кўпайтма матрицанинг $c[i,j]$ элементлари чоп этилади.

3. Дисперсияни ҳисоблашга доир масалалар

1) n тажриба ўтказилиб, $x_1, x_2, x_3, \dots, x_n$ натижалар олинган. Уларнинг ўртачасини:

$$\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i$$

ва дисперсиясини

$$\sigma = \frac{1}{n-1} \sqrt{\sum_{i=1}^n (x_i - \bar{x})^2}$$

ҳисоблаш программаларини тузинг.

```
using System;
/* Асосий программа */
class Urt_Dis{
public static void Main()
{
Console.WriteLine(" C# да 4-пр. Ўртача ва дисперсия ҳисоблаш ");
int n;
Console.WriteLine(" Киритинг n= ");
n=Convert.ToInt32(Console.ReadLine());
Double s,sr,disp;
Double [ ] x=new Double[n]; /* x массивни эълон қилиш */
s=0;
Console.WriteLine(" x1, x2, , , xn ларни киритинг=");
/* xi - ларни экрандан киритиш ва конвертация қилиш */
for(i=1; i<n; i++)
{
x[i]=Convert.ToDouble(Console.ReadLine());
s=s+x[i];
}
sr=s/n; /* Ўртачаси топилади */
s=0; //Дисперсия ҳис-ди
for(i=1; i<n; i++)
{
s=s+Math.Pow((x[i]-sr),2);
}
disp=Math.Sqrt(s)/(n-1);
Console.WriteLine( );
Console.WriteLine( " Ўрта арифметик қиймати "+sr);
Console.WriteLine( "Дисперсияси "+disp);
```

```

}
} /* 4-пр охири */

```

Бу программани бажаришга бериш учун C# ойнасидаги `Debug --> Start Debugging F5` лар кетма кет босилади. Натижада қора ойнада

C# да 4-пр. Ўртача ва дисперсия ҳисоблаш

Деган ёзув чиқади ва

Кирилинг $n=$

деган хабар чиқади. Шундан сўнг

x_1, x_2, \dots, x_n -ларни кирилинг

деган хабар чиқади. Буларни киришиб бўлгач, экранда

Ўрта арифметик қиймати $\bar{x}=$

Дисперсияси $\sigma=$

қийматлари билан чоп этилади.

4. Итерация усули ёрдамида қуйидаги

$$\ln x - x + 1,8 = 0 \quad (1)$$

Тенглама илдизини [2, 3] оралигида $\varepsilon = 10^{-4}$ аниқликда топинг.

Бунинг учун (1)ни $x = \ln x + 1,8$ кўринишига келтириб олинади. Сўнгра

$$x_{n+1} = \ln x_n + 1,8$$

деб, x_0 ни ихтиёрий олган ҳолда ҳисоблаймиз, бунда $n=0,1,2,\dots$ Ҳисоблашни токи

$$|x_{n+1} - x_n| < \varepsilon$$

шарти бажарилмагунча давом эттирилади. Программаси қуйидагича бўлади:

`using System;`

```

/* Асосий программа */

```

```

class Iterus

```

```

{

```

```

    public static void Main()

```

```

    {

```

```

        Console.WriteLine(" C# да 5-пр. Итерация усули ");

```

```

        int k,ki;

```

```

        Double x0, x1,eps;

```

```

        Console.WriteLine("Бошланғич яқинлашишни кирилинг  $x_0 =$  ");

```

```

        x0=Convert.ToDouble (Console.ReadLine());

```

```

        Console.WriteLine("Яқинлашиш хатолигини кирилинг  $\varepsilon =$  ");

```

```

        eps=Convert.ToDouble (Console.ReadLine());

```

```

        Console.WriteLine("Итерация сонига чегарани кирилинг  $ki =$  ");

```

```

        ki=Convert.ToInt32 (Console.ReadLine());

```

```

        k=-1;

```

```

        kmet: k++;

```

```

        if(k>ki)goto met1;

```

```

        x1=log(x0)+1,8;

```

```

        if(Math.Abs(x1-x0)<eps) goto met1;

```

```

            {x0=x1; goto kmet;

```


met1:

```
Console.WriteLine(" Тақр илдиз x1=" + x1; " аниқлик ε=" + eps );
Console.WriteLine(" Итерациялар сони k=" + k);
Console.ReadLine(); /* диалог режимида кутиш учун */
}
```

```
} /* 5-пр охири */
```

МУҲОКАМА

Бу программани бажаришига бериш учун C# ойнасидаги Debug --> Start Debugging F5 лар кетма кет босилади. Натижада қора ойнада

C# да 5-пр. Итерация усули

Бошланғич яқинлашишни киритинг $x_0 =$

Яқинлашиш хатолигини киритинг $\varepsilon =$

Итерация сонига чегарани киритинг $ki =$

Тақр илдиз x1=" + x1; " аниқлик ε=" + eps

Итерациялар сони k="

чоп этилади.

5. Оралиқни иккига бўлиш усули ёрдамида қуйидаги

$$\cos x - e^{-\frac{x^2}{2}} + x - 1 = 0$$

тенгламанинг илдизини [1;2] оралиқда $\varepsilon = 10^{-4}$ аниқликда топинг.

Программа тузишдан олдин алгоритмни кўриб ўтайлик. Фараз қилайлик,

$f(x) = 0$ тенгламанинг илдизини $[a, b]$ оралиқда ε аниқликда топши талаб қилинсин.

Агар берилган $[a, b]$ оралиқда битта ечим борлиги аниқ бўлса, у ҳолда $f(a)f(b) < 0$ бўлиши ҳам маълум. Оралиқни иккига бўлсак, у ҳолда изланаётган илдиз ушбу $[a, c]$ ва $[c, b]$

оралиқлардан бирида ётади, бу ерда $c = \frac{a+b}{2}$. Илдизнинг айнан қайси оралиқда эканлигини аниқлаш учун $f(c)$ ҳисобланади ва $d = f(a)f(c)$ ишораси текиширилади.

Бунда учта ҳол бўлиши мумкин:

1-ҳол. Агар $d < 0$ бўлса илдиз $[a, c]$ да ётади. Энди бу оралиқни яна иккига бўламиз.

Шунда оралиқлар $[a, \frac{a+c}{2}]$ ва $[\frac{a+c}{2}, c]$ бўлиб, илдизнинг булардан қайси оралиқда эканлиги

эса $f(a)f(\frac{a+c}{2}) < 0$ ёки $f(\frac{a+c}{2})f(c) < 0$ ларнинг қайси бирида бажарилишига боғлиқ ва худди шунга ўхшаш давом этади.

2-ҳол. Агар $d > 0$ бўлса илдиз $[c, b]$ да ётади. Оралиқни бўлиш амалини $[c, \frac{c+b}{2}]$ ва $[\frac{c+b}{2}, b]$ лар бўйича давом этилади. Бу ерда ҳам худди олдингидек $f(c)f(\frac{c+b}{2}) < 0$ ёки

$f(\frac{c+b}{2})f(b) < 0$ шартларидан бирини бажарилиши бўйича давом этади. Аммо, ҳар

иккала ҳолда ҳам оралиқни бўлиш жараёни токи бўлиниши оралиқчаси $|b - a| < \varepsilon$ бўлгунча такрорланаверади. Шарт бажарилиши биланоқ илдиз сифатида оралиқ чеккасидаги битта қиймати олинади.

3-ҳол. Агар $d = 0$ бўлса илдиз c нуқтада ётади, яъни $x=c$ бўлади.

Шуларга асосан ёзилган программа қуйидагича бўлади:

```
using System;
```

```
/* Асосий программа */
```

```

class Oral2b
{
    Static Double f(Double x)
    {
        /*       $\cos x - e^{-\frac{x^2}{2}} + x - 1 = 0$  */
        Double rez=Math.cos(x)- Math.exp(-x*x/2)+x-1;
    }
    public static void Main()
    {
        Console.WriteLine(" C# да 6-пр. Оралиқни иккига бўлиш усули ");
        Double a,b,eps,d,c;
        Console.WriteLine("Оралиқнинг чеккаларини киритинг a,b = ");
        a=Convert.ToDouble (Console.ReadLine());
        b=Convert.ToDouble (Console.ReadLine());
        Console.WriteLine("Яқинлашиш хатолигини киритинг  $\varepsilon$  = ");
        eps=Convert.ToDouble (Console.ReadLine());
        Console.WriteLine( );
        met1: c=(a+b)/2;
        d=f(a)*f(c);
        if(d<0) {
            b=c;
            if(Math.Abs(b-a)>eps) goto met1;
            else goto met2;
        }
        if(d>0) { a=c;
            if(Math.Abs(b-a)>eps) goto met1;
            else goto met2;
        }
        met2: x=c;
        Console.WriteLine(" Тақр илдиз x="+x; " аниқлик  $\varepsilon$ = "+eps );
        Console.ReadLine(); /* диалог режимида кутиш учун */
    }
} /* 6-пр охири */

```

Бу программани бажаришига бериш учун C# ойнасидаги Debug --> Start Debugging F5 лар кетма кет босилади. Натижада қора ойнада

C# да 6-пр. Оралиқни иккига бўлиш усули
Оралиқнинг чеккаларини киритинг a,b =
Яқинлашиш хатолигини киритинг ε =
Тақр илдиз x="+x; " аниқлик ε = "+eps
чоп этилади.

6. Ньютон усули билан қуйидаги

$$3x - 4\ln x - 5 = 0 \quad (1)$$

тенгламанинг илдизини [2; 4] оралигида $\varepsilon = 10^{-4}$ аниқликда топинг.

Бунинг учун (1)ни $f(x) = 3x - 4\ln x - 5 = 0$ кўринишга келтириб олинади. Сўнгра уни Тейлор қаторига бирор x_0 нуқта атрофида ёйсак, унда

$$f(x) = f(x_0) + (x - x_0)f'(x_0) = 0$$

дан агар $f'(x_0) \neq 0$ бўлса, унда

$$x = x_0 - f(x_0)/f'(x_0)$$

келиб чиқади. Бундан эса

$$x_{n+1} = x_n - f(x_n)/f'(x_n) \quad (n=0, 1, \dots)$$

рекуррент формулага эга бўламиз. Агар x_0 ни ихтиёрый деб олинса, ҳолда ҳисоблашни токи

$$|x_{n+1} - x_n| < \varepsilon \quad (n=0, 1, \dots)$$

шарти бажарилмагунча давом эттирилади. Программаси қуйидагича бўлади:

using System;

```
/* Асосий программа */
```

```
class IterNus
```

```
{
```

```
    Static Double funk(Double x)
```

```
    {
```

```
        /*  $f(x) = 3x - 4\ln x - 5 = 0$  */
```

```
        Double rez=3*x-4*Math.log(x)- 5;
```

```
    }
```

```
    Static Double funkPr(Double x)
```

```
    {
```

```
        /*  $f'(x) = 3 - 4/x$  */
```

```
        Double rez=3-4/x;
```

```
    }
```

```
    public static void Main()
```

```
    {
```

```
        Console.WriteLine(" C# да 7-пр. Ньютон усули ");
```

```
        int k,ki;
```

```
        Double x0, x1,eps;
```

```
        Console.WriteLine("Бошланғич яқинлашишни киритинг  $x_0 =$  ");
```

```
        x0=Convert.ToDouble (Console.ReadLine());
```

```
        Console.WriteLine("Яқинлашиш хатолигини киритинг  $\varepsilon =$  ");
```

```
        eps=Convert.ToDouble (Console.ReadLine());
```

```
        Console.WriteLine("Итерация сонига чегарани киритинг  $ki =$  ");
```

```
        ki=Convert.ToInt32 (Console.ReadLine());
```

```
        k=-1;
```

```
        kmet: k++;
```

```
        if(k>ki)goto met1;
```

```
        if(funkPr(x0)==0) go to metkon;
```

```
        x1=x0-funk(x0)/ funkPr(x0);
```

```
        if(Math.Abs(x1-x0)<eps) goto met1;
```

```
            { x0=x1; goto kmet;
```

```

    }
met1:
    Console.WriteLine(" Такр илдиз x1="+x1; " аниқлик  ε=" +eps );
    Console.WriteLine(" Итерациялар сони k="+k; );
    Console.ReadLine(); /* диалог режимида кутиш учун */
}

```

metkon:

```

} /* 7-пр охири */

```

Бу программани бажаришига бериш учун C# ойнасидаги Debug --> Start Debugging

F5 лар кетма кет босилади. Натижада қора ойнада

C# да 7-пр. Ньютон усули

Бошланғич яқинлашишни киритинг $x_0 =$

Яқинлашиш хатолигини киритинг $\varepsilon =$

Итерация сонига чегарани киритинг $ki =$

Такр илдиз $x1 =$ ”+x1; “ аниқлик $\varepsilon =$ ”+eps

Итерациялар сони $k =$ ”

чоп этилади.

7. Чизикли алгебраик тенгламалар системасини ечишда оддий итерациялар усулини қўллаш учун уни каноник кўринишга келтириб ечинг.

$$\begin{cases} 2x_1 - 3x_2 + 6x_3 + 20x_4 = 10 \\ x_1 + 2x_2 - 15x_3 + 3x_4 = 17 \\ -8x_1 - x_2 + 10x_3 + 19x_4 = 10 \\ 11x_1 - 9x_2 - 2x_3 - x_4 = 6 \end{cases}$$

Бундан диагонал элементларига нисбатан ечиб олсак,

$$\begin{cases} x_1 = (10 + 3x_2 - 6x_3 - 20x_4)/2 \\ x_2 = (17 - x_1 + 15x_3 - 3x_4)/2 \\ x_3 = (10 + 8x_1 + x_2 - 19x_4)/10 \\ x_4 = 11x_1 + 9x_2 + 2x_3 - 6 \end{cases}$$

келиб чиқади. Энди оддий итерация усулига кўра системани ёзиб олсак

$$\begin{cases} x_1^{k+1} = (10 + 3x_2^k - 6x_3^k - 20x_4^k)/2 \\ x_2^{k+1} = (17 - x_1^k + 15x_3^k - 3x_4^k)/2 \\ x_3^{k+1} = (10 + 8x_1^k + x_2^k - 19x_4^k)/10 \\ x_4^{k+1} = 11x_1^k + 9x_2^k + 2x_3^k - 6 \end{cases}$$

у ҳолда x_i^k ларга кўра x_i^{k+1} ларни ҳисоблаш мумкин. ($i=1,4$; $k=0, 1, \dots$)

Ҳисоблашлар токи $\max\{|x_i^{k+1} - x_i^k|\} < \varepsilon$ ($k=0, 1, \dots$)

шарти бажарилмагунча давом эттирилади. Программаси қуйидагича бўлади:

using System;

```

    /* Асосий программа */

```

```

class IterOdd

```

```

{

```

```

    public static void Main()

```

```

    {

```

```

        Console.WriteLine(" C# да 8-пр. ЧАТСни оддий итерация билан ечиш");

```

```

        int k,ki;

```

```

Double x10, x1, x20, x2, x30, x3, x40, x4, eps;
    Console.WriteLine(
        “Бошланғич қийматларни киритинг x10, x20, x30, x40 = ” );
        x10=Convert.ToDouble (Console.ReadLine());
        x20=Convert.ToDouble (Console.ReadLine());
        x30=Convert.ToDouble (Console.ReadLine());
        x40=Convert.ToDouble (Console.ReadLine());
        Console.WriteLine(“Яқинлашиш хатолигини киритинг ε= ” );
        eps=Convert.ToDouble (Console.ReadLine());
        Console.WriteLine(“Итерация сонига чегарани киритинг ki= ” );
        ki=Convert.ToInt32 (Console.ReadLine());
        k=-1;
        kmet: k++;
        if(k>ki)goto met1;
        x1=(10-3*x20-6*x30-20*x40)/2;
        x2=(17-x10+15*x30-3*x40)/2;
        x3=(10+8*x10+x20-19*x40)/10;
        x4=11*x10+9*x20 +2*x30-6)/10;
        if(Math.Abs(x1-x10) >eps)
            { x10=x1; x20=x2; x30=x3; x40=x4;    goto kmet);
            }
        if(Math.Abs(x2-x20) >eps)
            { x10=x1; x20=x2; x30=x3; x40=x4;    goto kmet);
            }
        if(Math.Abs(x3-x30) >eps)
            { x10=x1; x20=x2; x30=x3; x40=x4;    goto kmet);
            }
        if(Math.Abs(x4-x40) >eps)
            { x10=x1; x20=x2; x30=x3; x40=x4;    goto kmet);
            }
        met1:
        Console.WriteLine(“ Такр илдиз x1=”+x1; “ x2=”+x2; “ x3=”+x3;
        “ x4=”+x4);
        Console.WriteLine(“ аниқлик ε= ”+eps );
        Console.WriteLine(“ Итерациялар сони k=”+k; );
        Console.ReadLine(); /* диалог режимида кутиш учун */
    }
} /* 8-пр охири */

```

Бу программани бажаришига бериш учун C# ойнасидаги Debug --> Start Debugging F5 лар кетма кет босилади. Натижада қора ойнада

ХУЛОСА

C# да 8-пр. ЧАТСни оддий итерация билан ечиш

Бошланғич қийматларни киритинг x10, x20, x30, x40 =

деган хабар чиқади. Шунда, $x_{10}, x_{20}, x_{30}, x_{40}$ лар учун

1Enter

2Enter

3Enter

4Enter

деб фараз қилсак, унда

Яқинлашиш хатолигини киритинг $\epsilon=$

Итерация сонига чегарани киритинг $ki=$

ларнинг қийматлари киритилса, ундан кейин

аниқлик $\epsilon=$ ”...

Итерациялар сони $k=$...

ва системанинг ечимлари

Тақр илдиз $x_1=$...; $x_2=$...; $x_3=$...; $x_4=$...

қўринишларда босилиб чиқади.

8. Тенгламалар системасини ечиш

Фараз қилайлик, қуйидаги икки номаълумли иккита тенглама системасини ечиш талаб қилинсин:

$$\begin{cases} a_{11}x + a_{12}y = b_1 \\ a_{21}x + a_{22}y = b_2 \end{cases}$$

Бу ерда a_{ij} ва b_i ларнинг қийматлари берилади ($i=1,2; j=1,2$).

Системани Крамер формуласи ёрдамида ечиш учун, асосий детерминант

$$D = a_{11} * a_{22} - a_{21} * a_{12}$$

ҳисобланади ва агар $D \neq 0$ бўлсагина ёрдамчи детерминантларни қуйидаги формулалар билан ҳисобланади:

$$Dx = b_1 * a_{22} - b_2 * a_{12}$$

$$Dy = a_{11} * b_2 - a_{21} * b_1$$

Системанинг ечимлари эса $x = Dx/D$ ва $y = Dy/D$

формулалар орқали топилади. Акс ҳолда, яъни $D=0$ бўлганда тенгламалар системаси ечимга эга бўлмайди. Шунга асосан ёзилган программа қуйидагича бўлади:

using System;

/* Асосий программа */

class Kramer

{

public static void Main()

{

Console.WriteLine(“ C# да 8-пр. Тенгламалар системасини Крамер усулида ечиш ”);

Double x, y, D, Dx, Dy ;

Double [,] a=new Double[3,3]; /* А массивни эълон қилиш */

Double [] b=new Double[3]; /* В массивни эълон қилиш */

Console.WriteLine(“ $a_{11}, a_{12}, a_{21}, a_{22}$ ларни киритинг= ”);

/* a_{ij} - лар диалог режимда экрандан киритиш ва конвертация қилиш */

a[1,1]=Convert.ToDouble(Console.ReadLine());

a[1,2]=Convert.ToDouble(Console.ReadLine());

```

a[2,1]=Convert.ToDouble(Console.ReadLine());
a[2,2]=Convert.ToDouble(Console.ReadLine());
Console.WriteLine(" b1, b2 – ларни киритинг=");
b[1]=Convert.ToDouble(Console.ReadLine());
b[2]=Convert.ToDouble(Console.ReadLine());
D=a[1,1]* a[2,2] - a[2,1]*a[1,2];    /* Асосий детерминантни ҳисоблаш */
If(D≠ 0){
    Dx=b[1]* a[2,2] - b[2]*a[1,2];    /* Ёрдамчи детерминант */
    Dy=a[1,1]*b[2] - a[2,1]*b[1]; /* Ёрдамчи детерминантни */
    x= Dx/D;   y= Dy/D;    /* Система ечимларини ҳисоблаш */
    Console.WriteLine(" Системанинг ечимлари x="+x, " y="+y);
    Console.WriteLine();
} else
    Console.WriteLine(" Системанинг ечимлари йўқ");
}
} /* 8 пр охири */

```

Бу программани бажаришига бериши учун C# ойнасидаги *Debug --> Start Debugging F5* лар кетма кет босилади. Натижада қора ойнада

C# да 4-пр

a_{11} , a_{12} , a_{21} , a_{22} – ларни киритинг

деган хабар чиқади. Шунда, A матрицани $A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$ ва B векторни $B = \begin{pmatrix} 3 \\ 7 \end{pmatrix}$ деб фараз қилсак, унда A нинг элементлари

1Enter

2Enter

3Enter

4Enter

каби босиб киритилади. Сўнгра ойнада

b_1 , b_2 – ларни киритинг

деган хабар чиққандан кейин B нинг элементлари

3Enter

7Enter

сингари киритилади. Натижада экранда

C# да 8-пр. Тенламалар системасини Крамер усулида ечиш

Системанинг ечимлари $x=1$ $y=1$

чиқади ва иш яқунланади. Агар $A = \begin{pmatrix} 1 & 2 \\ 2 & 4 \end{pmatrix}$ бўлса, экранда

Системанинг ечимлари йўқ

деган хабар чиқади.

9. Квадрат тенгламани ечиш

Малумки, $ax^2 + bx + c = 0$ тенгламани ечиш учун $d = b^2 - 4ac > 0$ бўлганда иккита ечим ушбулар орвали топилади:

$$x_1 = \frac{-b+\sqrt{d}}{2a}, \quad x_2 = \frac{-b-\sqrt{d}}{2a}. \text{ Агар } d < 0 \text{ бўлса, у ҳолда } x \in \emptyset.$$

Шунга кўра унинг программаси қуйидагича бўлади:

```
using System;
    /* Асосий программа */
class kvtye{
    public static void Main()
    {
        Console.WriteLine(" C# да 10-пр. Кв тенгламани ечиш" );
        Double a,b,c,d,x1,x2;
        Console.WriteLine(" Киритинг a,b,c= ");
        /* a,b,c- лар диалог реж-да экрандан киритиш ва конвертация қилиш */
        a=Convert.ToDouble(Console.ReadLine());
        b=Convert.ToDouble(Console.ReadLine());
        c=Convert.ToDouble(Console.ReadLine());
        d=b*b-4*a*c;
        if(d>=0){
            x1=(-b-Math.Sqrt(d))/(2*a);
            x2=(-b+Math.Sqrt(d))/(2*a);
            Console.WriteLine(); /* бўш сатр ташлаш */
            Console.WriteLine(" Ечимлар x1,x2= "+x1,x2);
        }
        else Console.WriteLine(" Ечим йўқ ");
        Console.ReadLine(); /* диалог режимда кутиш учун */
    }
}
```

Бу программани бажаришига бериш учун C# ойнасидаги *Debug --> Start Debugging F5* лар кетма кет босилади. Натижада қора ойнада

C# да 10-пр. Кв тенгламани ечиш

Киритинг a,b,c=

чиқади. Шунда,

1 Enter

2 Enter

1 Enter босилади. Натижада экранда

Ечимлар x1,x2= -1 -1

чиқади ва иш якунланади. Ҳақиқатдан ҳам a,b,c –лар 1,2,1 бўлганда мана шундай аниқ ечим олинади. Агар a,b,c –лар сифатида 2,2,1 киритилса, экранда

Ечим йўқ

деган хабар чиқади.

10. k натурал сон берилган. Унинг рақамлар сонини ҳисоблаш

Бунинг учун:

1). у сон киритилади

2)ҳисобчи олиб, унга 0 бериб қўйилади.

3)берилган сонни 10 га бўлинади

4) агар бўлинма 1 дан кичик бўлса ҳисобчи чоп этилади ва иш тўхтатилади.

Акс ҳолда ҳисобчини 1 тага оширади ва бошқаришни 3-бандга узатади.

Мана шу алгоритм бўйича ёзилган программа қуйидагича бўлади:

using System;

```
    /* Асосий программа */
    class Natson
    {
        public static void Main()
        {
            Console.WriteLine(
                “С# да 11-пр. к натурал соннинг рақамлар сонини ҳисоблаш ”);
            int n,k;
            Double l;
            Console.WriteLine(“ к натурал сонни киритинг k = ”);
            k=Convert.ToInt32 (Console.ReadLine());
            n=0; l=k;
            kmet: n++; /* Рақамлар сонини ҳисоблаш */
            l=l/10.0;
            if(l>=1)goto kmet;
            Console.WriteLine(“Рақамлар сони n=”+n );
            Console.ReadLine(); /* диалог режимида кутиш учун */
        }
    } /* 11-пр охири */
```

Бу программани бажаришига бериш учун С# ойнасидаги Debug --> Start Debugging F5 лар кетма кет босилади. Натижада қора ойнада

С# да 12-пр. к натурал соннинг рақамлар сонини ҳисоблаш **Натурал сонни киритинг**
k =

деган хабар экранга чиқади ва керакли бутун сон киритилади. Ҳисоблаш натижасида

Рақамлар сони n=

деб қиймати билан чоп этилади.

REFERENCES

1. Шильд,Г.Питер. С#. Учебный курс. (рус тилида), 2004.
2. Павловская Т. А. Программирование на языке высокого уровня. С# . Питер. 2006.