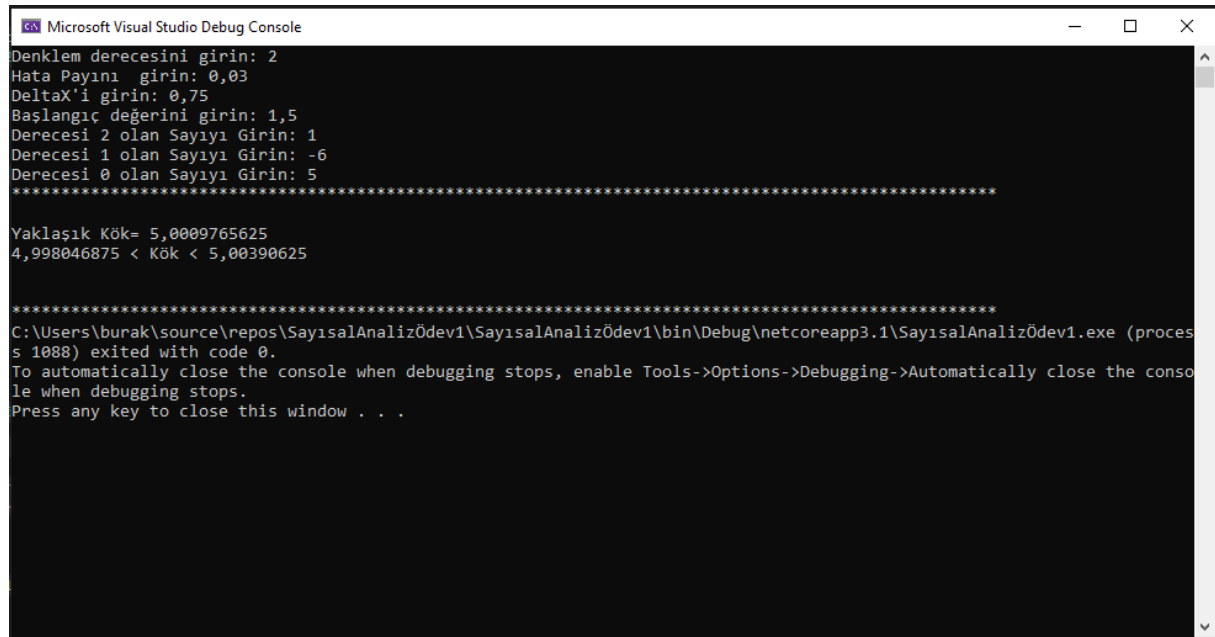


**Burak BOZ**

**18011706**

**C# - Windows Console Application' da hazırlanmıştır.**

**Ekran Çıktısı:**



```
Microsoft Visual Studio Debug Console
Denklemin derecesini girin: 2
Hata Payını girin: 0,03
DeltaX'i girin: 0,75
Başlangıç değerini girin: 1,5
Derecesi 2 olan Sayıyı Girin: 1
Derecesi 1 olan Sayıyı Girin: -6
Derecesi 0 olan Sayıyı Girin: 5
*****
Yaklaşık Kök= 5,0009765625
4,998046875 < Kök < 5,00390625
*****
C:\Users\burak\source\repos\SayısalAnalizÖdev1\SayısalAnalizÖdev1\bin\Debug\netcoreapp3.1\SayısalAnalizÖdev1.exe (process 1088) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .
```

**Kodlar:**

```
using System;

namespace SayısalAnalizÖdev1
{
    class Program
    {
        static void Main(string[] args)
        {
            /*
Değişkenler:_____
double      xBas:      Başlangıç Değeri
double      hPayi:      Hata Payı
double      deltaX:      X için bir sonraki sayının hesaplanmasında kullanılacak Delta X
değeri
double      fark:      eğer işaret değişirse n. adımdaki ve (n+1). adımdaki sayılar
arasındaki farkın tutulacağı değişkendir
double      kok:      en son bulunan kök bu değişkende tutulup yazdırılacaktır.
double      fX :      Fonksiyona gönderilen değerin hesaplanmış çıktısı bu değişken
üzerinde tutulacaktır.
double      değerler[2] : fonksiyona girilen xBas değerinden çıkan sonuç bu dizideki
elemanlarda tutulacak n. değer değerler[0], (n+1). değer değerler[1] içerisinde
saklanacaktır
            */
        }
    }
}
```

```

int         derece:      Fonksiyonun derecesini tutacak değişken
int         sayac:       döngüler içerisinde ilk sayının hesaplanmasında kullanılacak
int         matris[, ] : Fonksiyonun denklemi(denklemdeki katsayılar ve üs dereceleri)
bu matris içerisinde tutulacaktır
string      isaretler[2] : n. adımdaki ve (n+1) adımdaki fonksiyon sonucunun işaretini
tutmak için kullanılacak string dizidir
bool        isaretKontrol : İşaretin değişip değişmediği bu boolean değişken üzerinden
kontrol edilecektir.
        */

//Değişkenlerin Tanımlanması
double xBas,hPayi,deltaX,fark,kok=0;
int derece;
int sayac = 0;
double[] degerler = new double[2];
string[] isaretler = new string[2];
double fX;
bool isaretKontrol = true;
//-----

//Gerekli değerlerin kullanıcıdan alınması
Console.Write("Denklem derecesini girin: ");
derece = Convert.ToInt32(Console.ReadLine());
Console.Write("Hata Payını girin: ");
hPayi = Convert.ToDouble(Console.ReadLine());
fark = hPayi + 10;
Console.Write("DeltaX'i girin: ");
deltaX = Convert.ToDouble(Console.ReadLine());
Console.Write("Başlangıç değerini girin: ");
xBas = Convert.ToDouble(Console.ReadLine());
//-----

//Alınan değerlerin matris isimli integer diziye girilmesi
int[, ] matris = new int[2, derece + 1];
int gecici = derece;
for (int i = 0; i < derece + 1; i++)
{
    Console.Write("Derecesi " + gecici.ToString() + " olan Sayıyı Girin:
");
    matris[0, i] = Convert.ToInt32(Console.ReadLine());
    matris[1, i] = gecici;
    gecici--;
}
//-----

//Kökün bulunması
while (fark>hPayi)//Fark hata payından büyük olduğu sürece kök aranmaya
devam edilecek.
{
    sayac = 0;
    while (isaretKontrol)
    {
        fX = sayiHesapla(derece, xBas, matris);//f(x) değeri sayiHesapla()
fonksiyonuna gönderiliyor ve alınan sonuç fX değişkenine yazılıyor.
        if (sayac == 0)//Sayac 0 iken İlk adımda ilk sayının işareti
belirleniyor ve isaretler[0] dizisinin elemanına yazılıyor.
        {
            degerler[0] = fX;
            if (fX > 0)

```

```

        {
            isaretler[0] = "+";
            isaretler[1] = "+";
        }
        else
        {
            isaretler[0] = "-";
            isaretler[1] = "-";
        }
    }
    else//Sayaç 0 değilken sonraki sayıların hesaplanması sağlanıyor.
    {
        xBas = xBas + deltaX;//Başlangıç değerine deltaX değeri
eklenerek yeni sayı hesaplanıyor.
        fX = sayiHesapla(derece, xBas, matris);//Yeni sayının
fonksiyondaki çıktısı hesaplanıyor.
        degerler[1] = fX;

        if (degerler[1]>0)//Yeni sayının işareti belirleniyor.
        {
            isaretler[1] = "+";
        }
        else
        {
            isaretler[1] = "-";
        }

        if (isaretler[0] != isaretler[1])//Bulunan sayı ile ilk
hesaplanan sayının işareti karşılaştırılıyor.
        {
            isaretKontrol = false;//Eğer işaret değişmişse döngüden
çıkılması için isaretKontrol değişkeni false olarak güncelleniyor.
            fark = xBas - (xBas-deltaX);//Bir önceki sayı ile yeni
hesaplanan sayı arasındaki fark hesaplanıyor.
            if (fark<0)//Farkın mutlak değeri alınıyor.
            {
                fark = fark * -1;
            }
        }
    }

    degerler[0] = degerler[1];//Bulunan yeni sayı eski sayının yerine
yazılıyor ve yeni sayı hesaplanması için degerler[1] alanı boşaltılıyor.
    degerler[1] = 0;

    sayac++;
}

xBas = xBas - deltaX;//İşaret değişmiş ve döngüden çıkılmışsa
başlangıç değeri bir önceki değer olarak güncelleniyor.
deltaX = deltaX / 2;//deltaX değeri 2 ye bölünerek sonucun
hassaslaşması sağlanıyor.
isaretKontrol = true;//isaretKontrol tekrar true değerine getirilerek
iç döngünün tekrar çalışması sağlanıyor.
}
//-----

//Kökün ekrana yazdırılması
kok = ((xBas + deltaX)+(xBas+deltaX/2))/2;//Kök bulunan alt değer ve üst
değerin arasında bir değerdir. Bu yüzden iki değer ortalama alınır.

```

```

        string sonuc = (xBas + deltaX / 2).ToString() + " < Kök < " + (xBas +
deltaX).ToString();//Kökün aralığı hesaplanıyor.
        for (int i = 0; i < 100; i++)
        {
            Console.Write("*");
        }
        Console.WriteLine("\n\n");
        Console.WriteLine("Yaklaşık Kök= " + kok.ToString()+"\n" +
sonuc.ToString());//Yaklaşık kök ve aralık değeri ekrana yazdırılıyor.
        Console.WriteLine("\n\n");
        for (int i = 0; i < 100; i++)
        {
            Console.Write("*");
        }

        //-----

    }

    //f(x)=y Fonksiyona verilen değere göre sonuç üretilmesi
    public static double sayiHesapla(int derece, double sayi, int[, ]
veriler)//sayiHesapla() isimli fonksiyon f(x)=y hesaplanması için kullanılmaktadır.
    {
        double snc=0,us=1,toplam =0;//gerekli değişkenlerin tanımlanması
        for (int i = 0; i < derece+1; i++)
        {
            toplam = 0;
            us = 1;
            for (int j = 0; j < veriler[1,i]; j++)
            {
                us = us * sayi;
            }
            toplam += veriler[0,i]*(us);
            snc += toplam;
        }
        return snc;//Bulunan sonuç double tipinde döndürülüyor.
    }
    //-----
}
}

```