T.C.
YILDIZ TEKNİK ÜNİVERSİTESİ
ELEKTRİK-ELEKTRONİK FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

Dersin Adı

Sayısal Analiz

PROJE

Ödevin Konusu

Sayısal Analiz Dersinde Öğrenilen Algoritmalar

Dersi Veren Öğretim Üyesi

Prof.Dr. Banu DİRİ

Ödevi Yapan Öğrenci

Berke Kaan Açıkgöz

17011072

Ödev Teslim Tarihi

15.06.2021

PROJE KONU BAŞLIKLARI:

1. Bisection
2. Regula-Falsi
3. Newton-Rapshon
4. NxN'lik bir matrisin tersi
5. Gauss Eleminasyon
6. Gauss Seidal
7. Sayısal Türev (merkezi, ileri ve geri)
8. Simpson yöntemi
9. Trapez yöntemi
10. Değişken dönüşümsüz Gregory Newton Enterpolasyonu

Yukarıda yer alan konu başlıklarından **YEŞİL** ile işaretlenen başlıklar yapılabilmiş, **SARI** ile işaretlenen başlıklar ise yapılamamıştır.

YAPILABİLEN KONU BAŞLIKLARINA AİT KODLAR

1.Bisection:

```
void bisection(){
    char func[16];
    int a, b, c, n, ust1=1, ust2=1, ust3=1, i, itr=2, itrcount=1;
    float kok1, kok2, kok3, hata;

    printf("aX^n +/- b*X +/- c Formatinda bir fonksiyon giriniz lutfen\n(ORN: 3x^2-6x+2): ");
    scanf("%s", &func);
    a = func[0]-'0';
    b = func[5]-'0';
    c = func[8]-'0';
    n = func[3]-'0';
    if(func[4] == '-'){
        b= 0-b;
    }
    if(func[7] == '-'){
        c= 0-c;
    }

    printf("Fonksiyonun [kok1,kok2] aralik degerlerini giriniz:\n");
    printf("Kok1: ");
    scanf("%f", &kok1);
    printf("Kok2: ");
    scanf("%f", &kok2);
    printf("Hata degerini giriniz: ");
    scanf("%f", &hata);

    for(i=0; i<n; i++){
        ust1 = ust1 * kok1;
    }
    for(i=0; i<n; i++){
        ust2 = ust2 * kok2;
    }

    if((a*ust1+(b*kok1)+c)*(a*ust2+(b*kok2)+c)<0){
        while(fabs(kok1-kok2)/itr > hata){
            kok3 = (float)(kok1 + kok2)/2;
            for(i=0; i<n; i++){
                ust3 = ust3 * kok3;
            }
            if((a*ust3+(b*kok3)+c)*(a*ust2+(b*kok2)+c)<0){
                kok1 = kok3;
            }
            if((a*ust3+(b*kok3)+c)*(a*ust1+(b*kok1)+c)<0){
                kok2 = kok3;
            }

            printf("iterasyon %d: KOK: %f\n", itrcount, kok3);

            ust3=1;
            itr = itr * 2;
            itrcount++;
        }
        printf("KOK Yaklasik olarak:%.2f", kok3);
    }
    else{
        printf("Bu aralikta kok YOKTUR...");
    }
}
```

2. Newton-Rapshon:

```
void newtonRapshon(){
    double x1,x2,h;
    int i,j;
    float a[50];
    printf("Fonksiyonunuzun en yuksek katsayisini giriniz: ");
    scanf("%d",&i);
    for (j=0; j<i+1; j++){
        printf("Us degeri %d olan elemani giriniz: \n", j);
        scanf("%f",&a[j]);
    }
    printf("Fonksiyonunuz: ");
    for (j=0; j<i+1; j++){
        printf("(%.2f*x^%d)+",a[j],j);
    }

    printf("\nBaslangic degerinizi giriniz: ");
    scanf("%f",&x1);
    printf("Hata miktarini giriniz: ");
    scanf("%f",&h);
    x2= x1 - fonk(x1,i,j,a)/fonkturev(x1,i,j,a);

    while ( fabs(x2-x1) > h){
        x1=x2;
        x2= x1 - fonk(x1,i,j,a)/fonkturev(x1,i,j,a);
    }
    printf("Fonksiyonun koku: %.2f\n", x2);
}
```

3. NxN'lik bir matrisin tersi:

```
void matrisTersi(){

    int N, i, j, k, x, y;
    float matris[10][10], sonuc[10][10], diagonal;

    printf("NxN olan matrisin boyutunu giriniz: ");
    scanf("%d", &N);
    printf("Tersinin alinmasini istediginiz matrisi giriniz:\n");

    for(i=0; i<N; i++){
        for(j=0; j<N; j++){
            printf("[%d][%d]: ", i+1, j+1);
            scanf("%f", &matris[i][j]);
        }
    }

    for(i=0;i<N;i++){
        for(j=0;j<N;j++){
            if(i==j){
                sonuc[i][j] = 1;
            }
            else{
                sonuc[i][j] = 0;
            }
        }
    }

    for(i=0;i<N; i++){
        diagonal = matris[i][i];
        for(j=0;j<N;j++){
            sonuc[i][j] = sonuc[i][j] / diagonal;
            matris[i][j] = matris[i][j] / diagonal;
        }

        for(j=0;j<N;j++){
            if(j != i){
                for(k=0; k<N; k++){
                    sonuc[j][k] = sonuc[j][k] - (matris[j][i] * sonuc[i][k]);
                    matris[j][k] = matris[j][k] - (matris[j][i] * matris[i][k]);
                }
            }
        }
    }

    printf("Girdiginiz Matrisin Tersi:\n");
    for(x=0; x<N; x++){
        for(y=0; y<N; y++){
            printf("%.2f ", sonuc[x][y]);
        }
        printf("\n");
    }
    printf("\n");
}
```

4. Gauss Eleminasyon:

```
void gaussEleminasyon(){

    int i,j,k,n;
    float matris[20][20],c,x[10],sum=0.0;
    printf("NxN olan matrisin boyutunu giriniz: ");
    scanf("%d",&n);
    printf("Elemanlari, sonuclar da eklenmis sekilde girin [N+1][N] seklinde:\n");

    for(i=1; i<=n; i++){
        for(j=1; j<=(n+1); j++){
            printf("[%d][%d]: ", i,j);
            scanf("%f",&matris[i][j]);
        }
    }

    for(j=1; j<=n; j++){
        for(i=1; i<=n; i++){
            if(i>j){
                c=matris[i][j]/matris[j][j];
                for(k=1; k<=n+1; k++){
                    matris[i][k]=matris[i][k]-c*matris[j][k];
                }
            }
        }
    }

    x[n]=matris[n][n+1]/matris[n][n];

    for(i=n-1; i>=1; i--){
        sum=0;
        for(j=i+1; j<=n; j++){
            sum=sum+matris[i][j]*x[j];
        }
        x[i]=(matris[i][n+1]-sum)/matris[i][i];
    }
    printf("Cevaplar:\n");
    for(i=1; i<=n; i++){
        printf("x%d:%.3f\n",i,x[i]);
    }
}
```

5. Gauss Seidal:

```
void gaussSeidal(){

    int n, i,j, k, maxitr;
    float matris[20][20], cevaplar[20], hata, max,t,s,e;;
    printf("NxN olan matrisin boyutunu giriniz: ");
    scanf("%d",&n);
    printf("Elemanlari, sonuclar da eklenmis sekilde girin [N+1][N] seklinde:\n");

    for(i=0; i<n; i++){
        for(j=0; j<(n+1); j++){
            printf("[%d][%d]: ", i,j);
            scanf("%f",&matris[i][j]);
        }
    }

    printf("Hata Payi: ");
    scanf("%f", &hata);
    printf("Maks Itersayon: ");
    scanf("%d", &maxitr);

    for(i=0;i<n;i++){
        cevaplar[i]=0;
    }

    printf(" ITR      x         y         z\n");

    for(i=0;i<maxitr;i++){
        max=0;
        for(j=0;j<n;j++){
            s=0;
            for(k=0;k<n;k++){
                if(k!=j){
                    s+=matris[j][k]*cevaplar[k];
                }
            }

            t=(matris[j][n]-s)/matris[j][j];
            e=fabs(cevaplar[j]-t);
            cevaplar[j]=t;
        }

        printf("%3d    ", i+1);
        for(j=0;j<n;j++){
            printf("%.4f    ",cevaplar[j]);
        }
        printf("\n");
    }

    printf("%2d iterasyon sonrasi cevaplar:\n", i);
    for(j=0;j<n;j++){
        printf("X[%d]=%.3f\n", j+1,cevaplar[j]);
    }
}
```

6. Sayısal Türev (Merkezi,İleri,Geri):

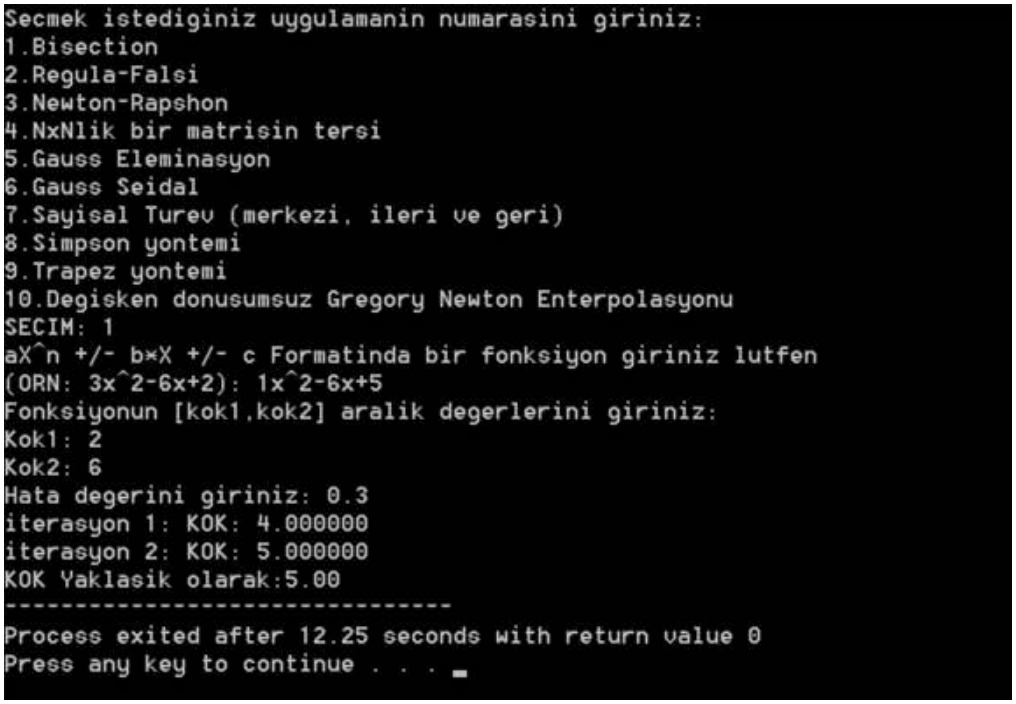
```
void sayisalTurev(){
    float dx,x,tur;
    int i,j;
    int k=0;
    float a[50];
    printf("Fonksiyonunuzun en yuksek katsayisini giriniz: ");
    scanf("%d",&i);
    for (j=0; j<i+1; j++){
        printf("Us degeri %d olan elemani giriniz: \n", j);
        scanf("%f",&a[j]);
    }
    printf("Fonksiyonunuz: ");
    for (j=0; j<i+1; j++){
        printf("(%.2f*x^%d)+",a[j],j);
    }

    printf("\b\nX degerini giriniz: ");
    scanf("%f",&x);
    printf("X'in degisim oranini giriniz(dx): ");
    scanf("%f",&dx);

    printf("Geri icin 1, Merkezi icin 2, Ileri icin 3'u seciniz: ");
    scanf("%d",&k);

    if (k==1){
        tur=(fonk(x,i,j,a)-fonk(x-dx,i,j,a))/dx;
    }
    else if (k==2){
        tur=(fonk(x+dx,i,j,a)-fonk(x-dx,i,j,a))/(2*dx);
    }
    else if (k==3){
        tur=(fonk(x+dx,i,j,a)-fonk(x,i,j,a))/dx;
    }
    else{
        printf("Yanlis giris yaptiniz!!!");
    }

    printf("\nFonksiyonunuzun Turevi: %.2f\n", tur);
}
```

7. Trapez Yöntemi:

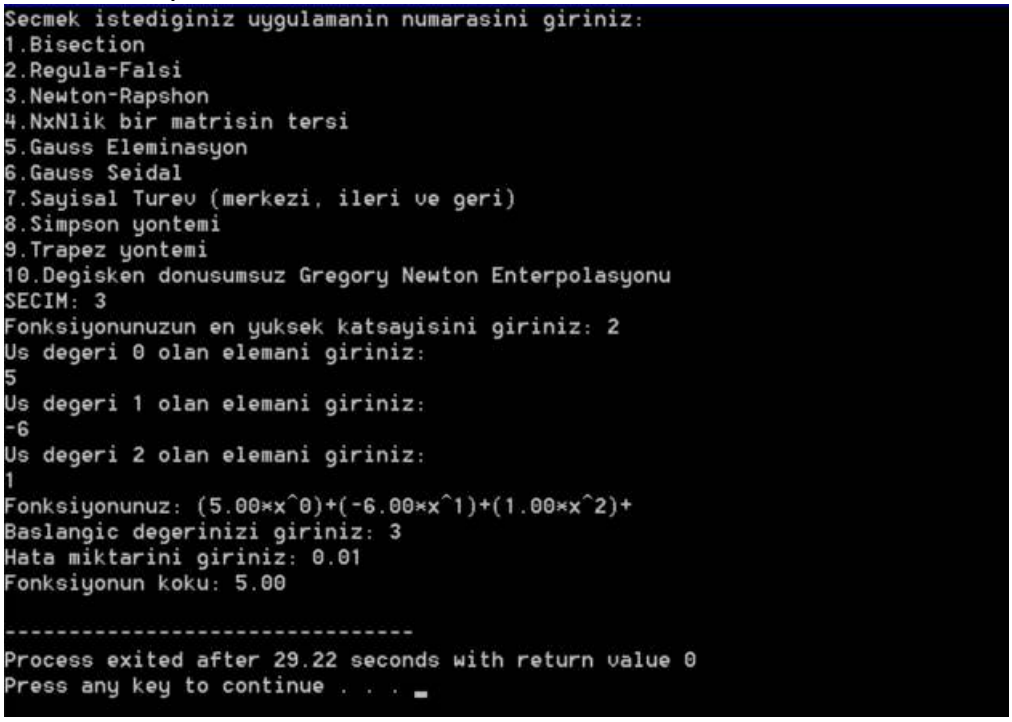
```
void trapez(){
    float dx,x1,x2,area;
    int power,j;
    float a[50];

    printf("Fonksiyonunuzun en yuksek katsayisini giriniz: ");
    scanf("%d",&power);
    for (j=0; j<power+1; j++){
        printf("Us degeri %d olan elemani giriniz: \n", j);
        scanf("%f",&a[j]);
    }
    printf("Fonksiyonunuz: ");
    for (j=0; j<power+1; j++){
        printf("+(%.2f*x^%d)",a[j],j);
    }

    printf("\nX1 degerini giriniz: ");
    scanf("%f",&x1);
    printf("\nX2 degerini giriniz: ");
    scanf("%f",&x2);
    printf("\nAdim boyutunu giriniz: ");
    scanf("%f",&dx);
    area=0;

    while(x1<x2){
        area+=dx*(fonk(x1,power,j,a)+fonk(x1+dx,power,j,a))/2;
        x1+=dx;
    }

    printf("\nFonksiyonunuzun altindaki alan: %.2f\n", area);
}
```

ÇALIŞTIRILIP SONUÇ ALINMIŞ EKRAN ÇIKTILARI

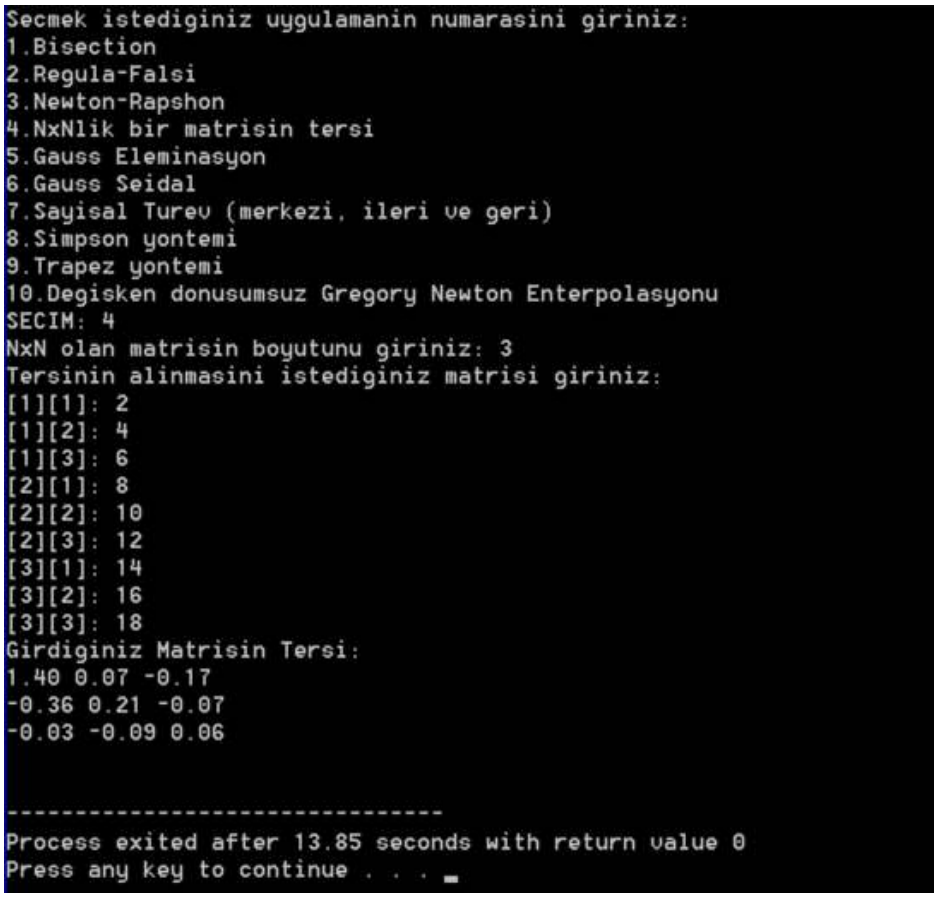
1. Bisection



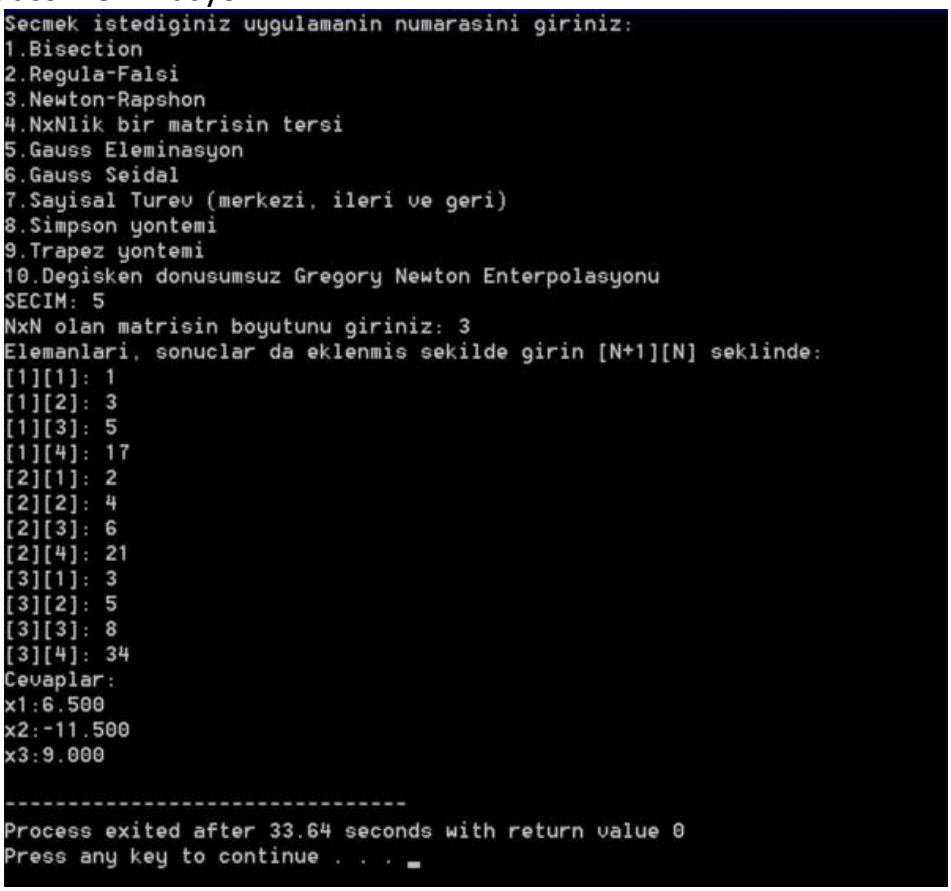
2. Newton-Rapshon



3. NxN’lik bir matrisin tersi



4. Gauss Eleminasyon



5. Gauss Seidal

```
Secmek istediginiz uygulamanin numarasini giriniz:
1.Bisection
2.Regula-Falsi
3.Newton-Rapshon
4.NxNlik bir matrisin tersi
5.Gauss Eleminasyon
6.Gauss Seidal
7.Sayisal Turev (merkezi, ileri ve geri)
8.Simpson yontemi
9.Trapez yontemi
10.Degisken donusumsuz Gregory Newton Enterpolasyonu
SECIM: 6
NxN olan matrisin boyutunu giriniz: 2
Elemanlari, sonuclar da eklenmis sekilde girin [N+1][N] seklinde:
[0][0]: 2
[0][1]: 4
[0][2]: 8
[1][0]: 3
[1][1]: 6
[1][2]: 12
Hata Payi: 0.01
Maks Itersayon: 3
 ITR      x         y         z
  1    4.0000    0.0000
  2    4.0000    0.0000
  3    4.0000    0.0000
 3 iterasyon sonrasi cevaplar:
X[1]=4.000
X[2]=0.000

--------------------------------
Process exited after 19.02 seconds with return value 0
Press any key to continue . . .
```

6. Sayısal Türev (merkezi, ileri ve geri)

```
Secmek istediginiz uygulamanin numarasini giriniz:
1.Bisection
2.Regula-Falsi
3.Newton-Rapshon
4.NxNlik bir matrisin tersi
5.Gauss Eleminasyon
6.Gauss Seidal
7.Sayisal Turev (merkezi, ileri ve geri)
8.Simpson yontemi
9.Trapez yontemi
10.Degisken donusumsuz Gregory Newton Enterpolasyonu
SECIM: 7
Fonksiyonunuzun en yuksek katsayisini giriniz: 2
Us degeri 0 olan elemani giriniz:
2
Us degeri 1 olan elemani giriniz:
4
Us degeri 2 olan elemani giriniz:
6
Fonksiyonunuz: (2.00*x^0)+(4.00*x^1)+(6.00*x^2)+
X degerini giriniz: 3
X'in degisim oranini giriniz(dx): 1
Geri icin 1, Merkezi icin 2, Ileri icin 3'u seciniz: 2

Fonksiyonunuzun Turevi: 40.00

--------------------------------
Process exited after 17.32 seconds with return value 0
Press any key to continue . . .
```

7. Trapez yöntemi