BİLGİSAYAR PROGRAMLAMA 1

Ders Notu 12 – Matematiksel Fonksiyonlar

Konya Teknik Üniversitesi
Elektrik – Elektronik Mühendisliği Bölümü

16.05.2024
Konya

Matematiksel Fonksiyonlar

Kod yazarken program içinde matematiksel bir işlem yapma ihtiyacı doğduğunda, kullanılan programlama dilinin hazır kütüphane fonksiyonlarını öncelikli tercih etmek daha avantajlı olabilir.

Eğer problemin doğrudan ya da dolaylı olarak çözümünde hazır fonksiyonlar yetersiz kalıyorsa, söz konusu problemi çözmek için programcı tarafından kod yazılabilir.

Matematik fonksiyonlarının türü *double* olup, kullanılabilmeleri için genellikle **math.h** başlık dosyasının program içerisine dahil edilmesi gerekir.

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

Matematiksel Fonksiyonlar

- math.h kütüphanesi içinde EDOM, HUGE_VAL, gibi bazı simgesel sabit bildirimler de bulunmaktadır.

- Bunlardan EDOM, fonksiyona gelen parametreler fonksiyona tanımlanmış sınırların dışında kaldığında kullanıcıyı uyarmak için kullanılır (Negatif bir sayının karekökünün hesaplanmaya çalışılması gibi.).

- HUGE_VAL ise fonksiyonda hesaplanan sonucun double tipinde gösterilemeyecek kadar büyük ya da küçük olduğunu kullanıcıya bildirir.

Karmaşık Sayılar

- C dilinde, karmaşık sayıları tanımlamak için *int*, *float* gibi özel bir veri tipi yoktur.

- Karmaşık sayılarla çalışabilmek için **complex.h** kütüphanesinde bazı tanımlamalar yapılmıştır, dolayısıyla bu başlık dosyasının programa dahil edilmesi gerekir.

```
#include<complex.h>
```

- **z=a+jb** şeklinde tanımlanan kompleks sayıların **a** ve **b** katsayıları, sırasıyla sayının reel(gerçek) ve imajiner(sanal) bileşenlerini temsil etmekle birlikte, birer *double* ya da *long double* veri olarak tanımlanabilirler.

```
double _Complex a = 3+5*I ;
```

Örnek

```c
#include<stdio.h>
#include<complex.h>
main() {
    double _Complex a=3+5*I;
    double _Complex b=3+4*I;
    printf("%lf\n", creal(a)*cimag(b));
    printf("%lf\n", cimag(a)+creal(b));
    return 0;  }
```

```
12.000000
8.000000

--------------------------------
Process exited after 0.04058 seconds with return value 0
Press any key to continue . . .
```

Trigonometrik Fonksiyonlar

Fonksiyon Adı	İşlevi / Amacı
sin(x)	Sinüs: Radyan olarak verilen x açısının sinüsünü hesaplar.
cos(x)	Kosinüs: Radyan olarak verilen x açısının kosinüsünü hesaplar.
tan(x)	Tanjant: Radyan olarak verilen x açısının tanjantını hesaplar. Açı (pi/2)'nin tek katlarına çok yakınsa hata verebilir.
asin(x)	Ters sinüs: Verilen x değerinin ters sinüsünü hesaplar. x, (-1,1) arasında değilse EDOM hatası üretilir. (sonuç -pi/2 ile pi/2 arasında)
acos(x)	Ters kosinüs: Verilen x değerinin ters kosinüsünü hesaplar. x, (-1,1) arasında değilse EDOM hatası üretilir. (sonuç 0 ile pi arasında)
atan(x)	Ters tanjant: Verilen x değerinin ters tanjantını hesaplar. (sonuç –pi/2 ile pi/2 arasında)
sinh(x)	Hiperbolik sinüs: Radyan olarak verilen x açısının hiperbolik sinüsünü hesaplar.
cosh(x)	Hiperbolik kosinüs: Radyan olarak verilen x açısının hiperbolik kosinüsünü hesaplar.

Örnek

```c
#include<stdio.h>
#include<math.h>
main() {
    float x, y, PI = 22.0/7.0;

    printf("\n\n sin(%.2f)=%.2f",x,sin(x));

    x = 45;
    y = x * (PI/180);

    printf("\n\n sin(%.2f)=%.2f",x,sin(y));
    printf("\n\n cos(%.2f)=%.2f",x,cos(y));
    printf("\n\n tan(%.2f)=%.2f",x,tan(y));

    printf("\n\n *** Program Sonu *** \n\n");
    return 0;  }
```

```
sin(0.00)=0.00

sin(45.00)=0.71

cos(45.00)=0.71

tan(45.00)=1.00

*** Program Sonu ***
```

- math.h kütüphanesindeki Sin( ) gibi fonksiyonları kullanırken açı radyan cinsinden girilmelidir. Sin(45°)=Sin(π/4)

Yuvarlatma Fonksiyonları

- **ceil(x)**: x değerinden büyük en küçük tam sayıyı bulur ve çağırana gönderir.

  Örneğin: x=7.3 ise 8, x=-7.3 ise -7 gönderir.

- **floor(x)**: x' ten küçük en büyük tam sayıyı bulur ve çağırana gönderir.

  Örneğin: x=7.3 ise 7, x=-7.3 ise -8 gönderir.

Logaritmik Fonksiyonlar

- **exp(x)**: e'nin x. kuvvetini hesaplar ve çağırana gönderir. Üstten taşma olursa HUGE_VAL, alttan taşma olursa 0 gönderir.

- **log(x)**: x'in doğal logaritmasını (ln) hesaplar ve çağırana gönderir. x eksi sayı ise EDOM gönderir.

- **log10(x)**: x'in 10 tabanında logaritmasını hesaplar ve çağırana gönderir. x eksi sayı ise EDOM gönderir.

Üstel Fonksiyonlar

- **pow(x,y)**: x'in y. kuvvetini hesaplar ve çağırana gönderir. Eğer x eksi bir sayı ve y bir tam sayı değilse ya da x sıfırsa ve y sıfırdan küçük ya da eşitse EDOM üretilir.

- **ldexp(x,n)**: 2'nin n. kuvvetiyle x'in çarpımını hesaplar ve çağırana gönderir.

Diğer Bazı Fonksiyonlar

- **abs(x):** x'in mutlak değerini gönderir.
- **fabs(x):** Gerçel sayıların mutlak değerini bulmak için kullanılır.
- **labs(x):** Uzun tamsayının mutlak değerini gönderir.
- **sqrt(x)**: x'in karekökünü hesaplayıp gönderir. x negatifse EDOM üretilir.
- **rand()**: Rastgele sayı üretmekte kullanılır.
- **div_t div(x,y):** x'i y'ye böler ve bölme sonucunu div_t içerisinde iki elemana yerleştirir.
- **ldiv_t ldiv(x,y):** div() fonksiyonunun uzun tam sayıları bölmek için kullanılan versiyonudur.
- **fmod(x,y):** x'in y'ye bölümünden kalanı gönderir.
- **modf(x, *y)**: x gerçel sayısının tam ve kesirli kısımlarını ayırır, tam kısmını y işaretçisinin gösterdiği adrese yerleştirip kesirli kısmını çağırana gönderir.

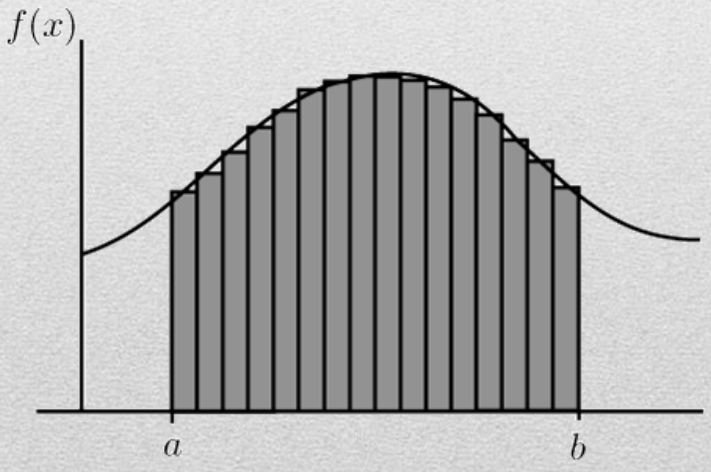
Örnek

```c
#include<stdio.h>
#include<math.h>
main() {
    float x=10.0, y=-5.0, n=2, z=25;

    printf("log10(%.1f)=%.2f \n",x,log10(x));
    printf("log(%.1f)=%.2f \n",x,log(x));
    printf("%.1f 'nin mutlak degeri: %.2f \n",y,fabs(y));
    printf("%.1f uzeri %.1f 'nin degeri: %.2f \n",y,n, pow(y,n));
    printf("%.1f 'nin karekok degeri: %.2f \n",z,sqrt(z));

    return 0;  }
```

```
log10(10.0)=1.00
log(10.0)=2.30
-5.0 'nin mutlak degeri: 5.00
-5.0 uzeri 2.0 'nin degeri: 25.00
25.0 'nin karekok degeri: 5.00
```

İntegral Hesabı

- İntegral, temelde alan hesabı için kullanılır. Sürekli bir fonksiyonun belirli aralıkta hesaplanan integralinin değeri, söz konusu fonksiyonun grafiğinin x ekseniyle arasında kalan alanıdır. Yani şekildeki f(x) fonksiyonunun a'dan b'ye integrali; yaklaşık olarak a ve b arasındaki eşit genişlikli dikdörtgenlerin alanlarının toplamından elde edilebilir.



Buradaki dikdörtgen genişliği duyarlılığı ifade etmekle birlikte, dikdörtgen sayısı arttıkça gerçek sonuca daha çok yaklaşılmakta ancak bu ekstra hesaplamalar gerektirmektedir.

Örnek

```c
#include<stdio.h>

#define F(a) ((a*a)+4)
main() {
    float x, x_alt, x_ust;
    float top=0.0, h, integral;
    int dilim=1000;

    x_alt=1;
    x_ust=4;


    h= (x_ust-x_alt)/dilim;
    x=x_alt;

    while(x<x_ust) {
        top+=F(x);
        x+=h;
    }
    integral=h*top;

    printf("Integral (%.1f ve %.1f sinirlari arasinda yaklasik): %.2f\n",x_alt,x_ust,integral);
    return 0;  }
```

```
Integral (1.0 ve 4.0 sinirlari arasinda yaklasik): 32.98

--------------------------------
Process exited after 0.08965 seconds with return value 0
Press any key to continue . . .
```

ÖDEV

1. Kartezyen formatta verilen bir karmaşık sayıyı Polar formata çeviren ve tam tersini yapan 2 ayrı C programı yazınız.
2. Polar formatta girilen AC Gerilim değerini (Örn: 220V∟30°), kartezyen formatta verilen bir empedans (Örn: 20+j10 Ω) değerine bölerek polar formatta Akımı hesaplayan ve ekrana yazdıran bir C Programı yazınız. (Bölme işleminden önce kartezyen formattaki empedans polar formata çevrilmelidir.)

KAYNAKLAR

- Dr. Rıfat ÇÖLKESEN, Programlama Sanatı, Papatya Yayıncılık, Cilt 1, Kasım 2004.