İLERİ PROGRAMLAMA
BÖLÜM 1
BİLGİSAYAR VE PROGRAMLAMAYA GİRİŞ

Nuri ÖZALP

1.1 Amaç

- C programlama dili kullanarak genel programcılık/matematiksel problemlerin çözümü

- Yapısal programlama ve programlama teknikleri öğrenimi

- Ayrıca

- C++

- Hedef Kitle

- Matematikçiler
- Çok az veya hiç programlama bilmeyen bilim/teknik öğrencileri
- Dilde kapsamlı ilerleme isteyen tecrübeli programcılar

- Kaynaklar

- Kernigan ve Ritchie nin kitabı (The C Programming Language, Prentice-Hall, 1978)

-

1.2 Bilgisayar Nedir?

- Bilgisayar

- Hesap yapabilme ve mantıksal karar alabilme kapasiteli aygıtlar
- Programlar olarak adlandırılan komutların kontrolünde veri işler

İki yapı içerir:

1. Donanım
   - Bilgisayarı oluşturan değişik aygıtlar
   - Klavye, ekran, mouse(fare), disketler, bellek, CD-ROM, ve işlemci birimleri
2. Yazılım
   - Bilgisayarda çalışan programlar

1.3 Yapılandırma

- **Her bilgisayarda altı mantıksal birim vardır:**
- Girdi birimi

1.3 Yapılandırma

- **Her bilgisayarda altı mantıksal birim vardır:**
- Girdi birimi
- Girdi aygıtlarından (klavye, mouse) bilgi alır

1.3 Yapılandırma

- **Her bilgisayarda altı mantıksal birim vardır:**
- Girdi birimi
  - Girdi aygıtlarından (klavye, mouse) bilgi alır
- Çıktı birimi

1.3 Yapılandırma

- **Her bilgisayarda altı mantıksal birim vardır:**
- Girdi birimi
  - Girdi aygıtlarından (klavye, mouse) bilgi alır
- Çıktı birimi
  - ekrana, yazıcıya, diğer aygıtları kontrol etmek için) çıktı gönderir

1.3 Yapılandırma

- **Her bilgisayarda altı mantıksal birim vardır:**
- Girdi birimi
  - Girdi aygıtlarından (klavye, mouse) bilgi alır
- Çıktı birimi
  - ekrana, yazıcıya, diğer aygıtları kontrol etmek için) çıktı gönderir
- Bellek birimi

1.3 Yapılandırma

- **Her bilgisayarda altı mantıksal birim vardır:**
- Girdi birimi
  - Girdi aygıtlarından (klavye, mouse) bilgi alır
- Çıktı birimi
  - ekrana, yazıcıya, diğer aygıtları kontrol etmek için) çıktı gönderir
- Bellek birimi
  - Hızlı ulaşım, düşük kapasite, girdi bilgilerini yükler

1.3 Yapılandırma

- **Her bilgisayarda altı mantıksal birim vardır:**
- Girdi birimi
  - Girdi aygıtlarından (klavye, mouse) bilgi alır
- Çıktı birimi
  - ekrana, yazıcıya, diğer aygıtları kontrol etmek için) çıktı gönderir
- Bellek birimi
  - Hızlı ulaşım, düşük kapasite, girdi bilgilerini yükler
- Aritmetik ve mantıksal birim (ALU)

1.3 Yapılandırma

- **Her bilgisayarda altı mantıksal birim vardır:**
- Girdi birimi
  - Girdi aygıtlarından (klavye, mouse) bilgi alır
- Çıktı birimi
  - ekrana, yazıcıya, diğer aygıtları kontrol etmek için) çıktı gönderir
- Bellek birimi
  - Hızlı ulaşım, düşük kapasite, girdi bilgilerini yükler
- Aritmetik ve mantıksal birim (ALU)
  - Aritmetik işlemleri ve mantıksal kararları işler

1.3 Yapılandırma

- **Her bilgisayarda altı mantıksal birim vardır:**
- Girdi birimi
  - Girdi aygıtlarından (klavye, mouse) bilgi alır
- Çıktı birimi
  - ekrana, yazıcıya, diğer aygıtları kontrol etmek için) çıktı gönderir
- Bellek birimi
  - Hızlı ulaşım, düşük kapasite, girdi bilgilerini yükler
- Aritmetik ve mantıksal birim (ALU)
  - Aritmetik işlemleri ve mantıksal kararları işler
- Merkezi işlem birimi (CPU)

1.3 Yapılandırma

- **Her bilgisayarda altı mantıksal birim vardır:**
- Girdi birimi
  - Girdi aygıtlarından (klavye, mouse) bilgi alır
- Çıktı birimi
  - ekrana, yazıcıya, diğer aygıtları kontrol etmek için) çıktı gönderir
- Bellek birimi
  - Hızlı ulaşım, düşük kapasite, girdi bilgilerini yükler
- Aritmetik ve mantıksal birim (ALU)
  - Aritmetik işlemleri ve mantıksal kararları işler
- Merkezi işlem birimi (CPU)
  - Bilgisayarın diğer kısımları arasındaki koordinasyonu sağlar

1.3 Yapılandırma

- **Her bilgisayarda altı mantıksal birim vardır:**
- Girdi birimi
  - Girdi aygıtlarından (klavye, mouse) bilgi alır
- Çıktı birimi
  - ekrana, yazıcıya, diğer aygıtları kontrol etmek için) çıktı gönderir
- Bellek birimi
  - Hızlı ulaşım, düşük kapasite, girdi bilgilerini yükler
- Aritmetik ve mantıksal birim (ALU)
  - Aritmetik işlemleri ve mantıksal kararları işler
- Merkezi işlem birimi (CPU)
  - Bilgisayarın diğer kısımları arasındaki koordinasyonu sağlar
- İkincil depolama birimi

1.3 Yapılandırma

- **Her bilgisayarda altı mantıksal birim vardır:**
- Girdi birimi
  - Girdi aygıtlarından (klavye, mouse) bilgi alır
- Çıktı birimi
  - ekrana, yazıcıya, diğer aygıtları kontrol etmek için) çıktı gönderir
- Bellek birimi
  - Hızlı ulaşım, düşük kapasite, girdi bilgilerini yükler
- Aritmetik ve mantıksal birim (ALU)
  - Aritmetik işlemleri ve mantıksal kararları işler
- Merkezi işlem birimi (CPU)
  - Bilgisayarın diğer kısımları arasındaki koordinasyonu sağlar
- İkincil depolama birimi
  - Ucuz, uzun-dönemli, yüksek kapasiteli depo

1.3 Yapılandırma

- **Her bilgisayarda altı mantıksal birim vardır:**
- Girdi birimi
  - Girdi aygıtlarından (klavye, mouse) bilgi alır
- Çıktı birimi
  - ekrana, yazıcıya, diğer aygıtları kontrol etmek için) çıktı gönderir
- Bellek birimi
  - Hızlı ulaşım, düşük kapasite, girdi bilgilerini yükler
- Aritmetik ve mantıksal birim (ALU)
  - Aritmetik işlemleri ve mantıksal kararları işler
- Merkezi işlem birimi (CPU)
  - Bilgisayarın diğer kısımları arasındaki koordinasyonu sağlar
- İkincil depolama birimi
  - Ucuz, uzun-dönemli, yüksek kapasiteli depo
  - Aktif olmayan programları yükler

1.4 İşletim Sistemi Evrimi

- Kümesel işlem (Batch processing)

1.4 İşletim Sistemi Evrimi

- Kümesel işlem (Batch processing)
  - Her anda tek bir iş veya görev yapar

1.4 İşletim Sistemi Evrimi

- Kümesel işlem (Batch processing)
  - Her anda tek bir iş veya görev yapar
- İşletim sistemi (Operating system)

1.4 İşletim Sistemi Evrimi

- Kümesel işlem (Batch processing)
  - Her anda tek bir iş veya görev yapar
- İşletim sistemi (Operating system)
  - İşler(görevler) arasındaki geçişi düzenler

1.4 İşletim Sistemi Evrimi

- Kümesel işlem (Batch processing)
  - Her anda tek bir iş veya görev yapar
- İşletim sistemi (Operating system)
  - İşler(görevler) arasındaki geçişi düzenler
  - Artan verimlilik

1.4 İşletim Sistemi Evrimi

- Kümesel işlem (Batch processing)
  - Her anda tek bir iş veya görev yapar
- İşletim sistemi (Operating system)
  - İşler(görevler) arasındaki geçişi düzenler
  - Artan verimlilik
    - Bilgisayarın yaptığı iş miktarı

1.4 İşletim Sistemi Evrimi

- Kümesel işlem (Batch processing)
  - Her anda tek bir iş veya görev yapar
- İşletim sistemi (Operating system)
  - İşler(görevler) arasındaki geçişi düzenler
  - Artan verimlilik
    - Bilgisayarın yaptığı iş miktarı
- Çoklu programlama

1.4 İşletim Sistemi Evrimi

- Kümesel işlem (Batch processing)
  - Her anda tek bir iş veya görev yapar
- İşletim sistemi (Operating system)
  - İşler(görevler) arasındaki geçişi düzenler
  - Artan verimlilik
    - Bilgisayarın yaptığı iş miktarı
- Çoklu programlama
  - Bilgisayar kaynaklarının birçok iş veya görev tarafından paylaşımı

1.4 İşletim Sistemi Evrimi

- Kümesel işlem (Batch processing)
  - Her anda tek bir iş veya görev yapar
- İşletim sistemi (Operating system)
  - İşler(görevler) arasındaki geçişi düzenler
  - Artan verimlilik
    - Bilgisayarın yaptığı iş miktarı
- Çoklu programlama
  - Bilgisayar kaynaklarının birçok iş veya görev tarafından paylaşımı
- Zaman paylaşımı

1.4 İşletim Sistemi Evrimi

- Kümesel işlem (Batch processing)
  - Her anda tek bir iş veya görev yapar
- İşletim sistemi (Operating system)
  - İşler(görevler) arasındaki geçişi düzenler
  - Artan verimlilik
    - Bilgisayarın yaptığı iş miktarı
- Çoklu programlama
  - Bilgisayar kaynaklarının birçok iş veya görev tarafından paylaşımı
- Zaman paylaşımı
  - Bir kullanıcının işinin kısa bir kısmını çalıştırıp bir başkasının işine geçer v.s.

1.5 Bilgisayar Dilleri

Üç tip bilgisayar dili vardır

1. *Makine dili*

   Makineye özel talimatlar veren sayı katarları

   Örnek:

   +1300042774
   +1400593419
   +1200274027

2. *Assembly (ortak) dili*

   Temel bilgisayar işlemlerini temsil eden İngilizce-benzeri kısaltmalar

   Örnek:

   LOAD BASEPAY
   ADD OVERPAY
   STORE GROSSPAY

1.5 Bilgisayar Dilleri
(Devam)

3. *Yüksek düzey diller*
   Günlük İngilizcede kullanılanlara benzer kodlar
   Matematik notasyonu kullanır

Örnek:

$$\text{toplam} = \text{temel} + \text{ekstra}$$

1.6 C nin Tarihi

- **C**

  Ritchie tarafından önceki iki dilden geliştirildi, BCPL and B
  UNIX i geliştirmek için kullanıldı
  Modern bir işletim sistemi yaratmak için kullanıldı
  Donanımdan bağımsızdır (kullanılabilirlik)
  1970‘ lerin sonunda “Geleneksel C“ ye dönüştü

- **Standardlaştırma**

  C nin birbirinden uyumsuz bir çok varyasyonu oluştu
  Ortaklık için bir komisyon oluşturuldu
  1989 da Standard konuldu ve 1999 da yenilendi

1.6 C nin Tarihi
(Devam)

- **C ++**
  - Bell Lab.larında Bjarne Stroustrup tarafından C nin geliştirilmişi
  - C nin "toparlanmış" hali olup, nesne-tabanlılığı sağlar
  - Nesne-tabanlı dizayn oldukça güçlüdür
    10 - 100 kat daha üreticidir
  - Günümüzde, bilim ve endüstride dominant durumdadır
- **C++ ' öğrenme**
  - C++, C yi de içerdiği için, önce C de uzmanlaşıp, daha sonra C++ öğrenmeden yana olanlar çoğunluktadır
  - Bölüm 15 den itibaren, C++ -a başlanacak

1.7 Diğer Yüksek Düzey Diller

- FORTRAN
  Bilimsel ve mühendislik uygulamalarında kullanılır
- COBOL
  Çoklu verilerin yönetimi için kullanılır
- Pascal
  Akademik amaçlı kullanıma yöneliktir
- JAVA
  Dinamik ve interaktif içerikli Web tasarımı
  İş çevreleri için geniş boyutlu tasarımlar
  Web sunucularının fonksiyonelliğini geliştirir
  Müşteri servisleri için uygulama sağlar
  (cep telefonları, sayfa yapıcıları gibi)

1.8 Program Yazımı

- Programlama dili öğrenme $\longrightarrow$ Bilgisayar aracılığı ile problem çözme.
  - Dört temel adım içerir:

1. Problem Analizi
2. Algoritma Geliştirme
3. Kodlama, Çalıştırma ve Test
4. Bakım ve Yenileme (gerçek hayat kullanımında)

1.8 Program Yazımı

1. Problem analizi

- Problemin girdi/çıktı (I/O)sını belirle

  Problemi belirginleştirmenin iki ana parçası.

- Problemin çözüm analizi

  En çok zaman alan ve çaba gerektiren parçası
  Bu aşamada problemin çözüm analizi yapılmalıdır.
  Mümkün olduğunca problemi genelleştir.

Girdilerden çıktılar nasıl elde edilir?

Bu aşama problemin çözüm aşamasıdır.

Çözüm açık ve seçik olarak organize edilmeli ve yapılandırılmalı

Akış şeması/ basit komutlar ve bilindik sembollerle yazılmalı.

Çözüm üç temel kontrol yapısı kullanılarak tasarlanır:

1. Dizisel kontrol: Çözüm adımları ardarda sıralanır
2. Seçim kontrolü: Seçeneklerden biri seçilip devam edilir
3. Tekrar (döngü) kontrolü : Bir veya daha çok adım tekrarlanır

Her problemde bu üç yapıdan en az biri vardır.

Aşağıdaki adımlar izlenir

1. Düzenle(Edit) - Program editörde yazılır ve diske kaydedilir
2. Önişlemler(preprocess) - Önişlemci kodları işler.
3. Derle(Compile) - Derleyici nesne kodu oluşturur ve diske kaydeder
4. Bağlan(Link) - Bağlayıcı nesne kodu ile kütüphaneler arasında bağlantı kurar
5. Yükle(Load) - Yükleyici programı belleğe yükler.
6. Çalıştır(Execute) - CPU program çalışırken verilen görevleri yerine getirir (oluşan olası yeni verileri yüklemek gibi)
7. Test(Debug) - Program'ın doğruluğunu verilerle kontrol et

1.8 Program Yazımı

5. Program bakımı ve geliştirme (gerçek hayat kullanımı)

- Profesyonel aşama

  Program değişik ortamlarda çalışacak şekilde hazırlanmalı

  Geniş kullanıcı kitlesine sunulabilir olmalı

  Satış sonrası bakım ve teknik destek sağlanabilmeli

  Geliştirilmeye ve yenilenmeye uygun olmalı

  Program satışa sunulduktan sonra çıkacak sorunların çözümü/servis sürekliliği olmalı

  Kullanıcıdan gelecek yeni istek ve geliştirilmeye karşı uygun yazılmış olmalı.

1.9 Program Yazılımı Örnekleri

Üç örnek üzerinde uygulama

Üç farklı örnek problemin programını yapalım:

- **Problem 1.** Radyoaktif element polonyumun yarılanma ömrü 140 gündür. 150 gün sonunda 10 gr polonyumun kalan miktarını veren bir program yazınız.

1.9 Program Yazılımı Örnekleri

Üç örnek üzerinde uygulama

Üç farklı örnek problemin programını yapalım:

- **Problem 1.** Radyoaktif element polonyumun yarılanma ömrü 140 gündür. 150 gün sonunda 10 gr polonyumun kalan miktarını veren bir program yazınız.
- **Problem 2.** $x^2 + 4x + 2 = 0$ denkleminin köklerini bulan bir program yazınız.

1.9 Program Yazılımı Örnekleri

Üç örnek üzerinde uygulama

Üç farklı örnek problemin programını yapalım:

- **Problem 1.** Radyoaktif element polonyumun yarılanma ömrü 140 gündür. 150 gün sonunda 10 gr polonyumun kalan miktarını veren bir program yazınız.
- **Problem 2.** $x^2 + 4x + 2 = 0$ denkleminin köklerini bulan bir program yazınız.
- **Problem 3.** 1 den 100 e kadar tam sayıların toplamını bulan bir program yazınız.

1.9 Program Yazılımı Örnekleri

Üç örnek üzerinde uygulama (devam)

- **Program Analizi**

Problemi mümkün olduğunca genelleştir

	Girdiler	Çıktılar
Problem 1	yarılanma ömrü ($y = 140$ gün) süre ($s = 150$ gün) miktar ($m = 10$ gr)	kalan miktar (k) $k = m * (0.5)^{s/y}$
Problem 2	Katsayılar: $a = 1, b = 4, c = 2$	Kökler: $x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$
Problem 3	$N(= 100)$	$1 + 2 + ... + N$

- **Algoritma**

Problem1 (Dizisel yapı)

1. y, s, m değerlerini oku

- **Algoritma**

Problem1 (Dizisel yapı)

1. y, s, m değerlerini oku
2. $k \leftarrow m * (0.5)^{s/y}$ işlemini yap

- **Algoritma**

Problem1 (Dizisel yapı)

1. y, s, m değerlerini oku
2. $k \leftarrow m * (0.5)^{s/y}$ işlemini yap
3. k değerini yaz

- **Algoritma**

Problem 2 (Seçim yapısı)

1. a, b, c değerlerini oku

- **Algoritma**

Problem 2 (Seçim yapısı)

1. a, b, c değerlerini oku
2. $d \longleftarrow b^2 - 4ac$ hesapla

- **Algoritma**

Problem 2 (Seçim yapısı)

1. a, b, c değerlerini oku
2. $d \longleftarrow b^2 - 4ac$ hesapla
3. **Eğer** $d \geq 0$ ise $x_{1,2} \leftarrow \frac{-b \pm \sqrt{b^2-4ac}}{2a}$; **aksi halde** $x_{1,2} \leftarrow \frac{-b \pm (\sqrt{b^2-4ac})i}{2a}$

- **Algoritma**

Problem 2 (Seçim yapısı)

1. a, b, c değerlerini oku
2. $d \longleftarrow b^2 - 4ac$ hesapla
3. **Eğer** $d \geq 0$ ise $x_{1,2} \leftarrow \frac{-b \pm \sqrt{b^2-4ac}}{2a}$; **aksi halde** $x_{1,2} \leftarrow \frac{-b \pm (\sqrt{b^2-4ac})i}{2a}$
4. $x_{1,2}$ değerlerini yaz

- **Algoritma**

Problem 3 (Tekrar yapısı)

1. *N* değerlerini oku

- **Algoritma**

Problem 3 (Tekrar yapısı)

1. N değerlerini oku
2. toplam$\longleftarrow$ 0, sayac$\longleftarrow$ 0 al

- **Algoritma**

Problem 3 (Tekrar yapısı)

1. N değerlerini oku
2. toplam$\longleftarrow$ 0, sayac$\longleftarrow$ 0 al
3. sayac $\leq N$ **oldukça;**
   toplam$\longleftarrow$toplam+sayac
   sayac$\longleftarrow$sayac+1 **döngüsünü tekrarla**

- **Algoritma**

Problem 3 (Tekrar yapısı)

1. N değerlerini oku
2. toplam$\longleftarrow$ 0, sayac$\longleftarrow$ 0 al
3. sayac $\leq N$ **oldukça;**
   toplam$\longleftarrow$toplam+sayac
   sayac$\longleftarrow$sayac+1 **döngüsünü tekrarla**
4. toplam değerini yaz

1.9 Program Yazılımı Örnekleri

Üç örnek üzerinde uygulama (devam)

- **Kodlama çalıştırma ve test**

Problem 1 (Dizisel yapı)

```
#include<stdio.h>
#include<stdlib.h>
#include<math.h>
int main( void )
{
 float y,s,m,k;
 printf("Yarılanma ömrünü gir:");
 scanf("%f",&y);
 printf("Zaman süresini gir:");
 scanf("%f",&s);
 printf("Başlangıç miktarını gir:");
 scanf("%f",&m);
 k=m*pow(0.5,s/y);
   printf( " Kalan miktar: %f\n",k);
   system("pause");
   return 0;
 }
```

Çıktı:

```
Yarılanma ömrünü gir:140
Zaman süresini gir:150
Başlangıç miktarını gir:10
 Kalan miktar: 4.758476
Devam etmek için bir tuşa basın . . .
```

1.9 Program Yazılımı Örnekleri

Üç örnek üzerinde uygulama (devam)

- **Kodlama çalıştırma ve test**

Problem 2 (Seçim yapısı)

```
#include<stdio.h>
#include<stdlib.h>
#include<math.h>
int main( void )
{
 float a,b,c,d,x1,x2,rx,imx;
 printf("a,b,c katsayılarını gir:\n");
 scanf("%f%f%f",&a,&b,&c);
 d=b*b-4*a*c;
 if(d>=0){ x1=(-b+sqrt(d))/(2*a);
           x2=(-b-sqrt(d))/(2*a);
 printf("x1=%f x2=%f\n",x1,x2);   }
 else    {   rx=-b/(2*a); imx=sqrt(-d)/(2*a);
 printf("x1=%.2f+(%.2f)i
         x2=%.2f-(%.2f)i\n",rx,imx,rx,imx);}
 system("pause");
 return 0;
}
```

Çıktı:

```
a,b,c katsayılarını gir:
1 4 2
X1=-0585786 x2=-3414214
Devam etmek için bir tuşa basın . . .
```

```
a,b,c katsayılarını gir:
1 1 1
X1=-0.50+(0.87)i x2=-0.50-(0.87)i
Devam etmek için bir tuşa basın . . .
```

1.9 Program Yazılımı Örnekleri

Üç örnek üzerinde uygulama (devam)

- **Kodlama çalıştırma ve test**

Problem 3 (Tekrar yapısı)

```
#include<stdio.h>
#include<stdlib.h>
#include<math.h>
int main( void )
{
 int n,toplam=0,sayac;
 printf("n değerini gir:\n");
 scanf("%d",&n);
 for(sayac=1;sayac<=n;++sayac)
     toplam+=sayac;
 printf("1+2+...+%d = %d\n",n,toplam);
 system("pause");
 return 0;
}
```

Çıktı:

```
N değerini gir:
100
1+2+…+100 = 5050
Devam etmek için bir tuşa basın . . .
```