Temel Bilgisayar Programlama

C Programlamaya Giriş

Yrd. Doç. Dr. Tahir Emre Kalaycı

2012

İçerik

Temel Kavramlar

- Programlamayı bir problemi çözmek için yaparız
- Her bir alanın kendine özgü çözüm yöntemleri vardır
- Programlama geliştirme açısından problem çözmede iki aşama vardır:
  - Problem çözme aşaması
    - ★ Problemin tanımlanması
    - ★ Çözümün ana hatlarının ortaya konulması
    - ★ Ana hatlara bağlı olarak bir algoritma geliştirilmesi
    - ★ Algoritma doğruluğunun sınanması
  - Gerçekleştirim aşaması
    - ★ Algoritma kodları belirli bir programlama dili kodlarına dönüştürülür
    - ★ Program bilgisayarda derlenir ve çalıştırılır
    - ★ Programın belgelemesi ve bakımı yapılır
- Algoritma bulma ve programlama süreci şu şekildedir
  - Problemi belirleme ve anlama (Analiz)
  - Algoritma hazırlama (Tasarım)
  - Program geliştirme (Gerçekleştirim)
  - Programı çalıştırma ve doğruluğunu değerlendirme (Sınama)

Algoritma nedir?

- Algoritma bir problemi çözmek için ortaya koyduğumuz çözüm adımlarıdır.
- Sonlu bir işi tanımlamada kullanılan, açık ve seçik tanımlanabilen ve yürütülebilen ardışık adımlardan oluşan kümedir. (Kaynak: Özkan, 2003)
- Algoritma, matematikte ve bilgisayar biliminde bir işi yapmak için tanımlanan, bir başlangıç durumundan başladığında, açıkça belirlenmiş bir son durumunda sonlanan, sonlu işlemler (adımlar) kümesidir (Kaynak: Algoritma, Viki)

Algoritma nasıl geliştirilir

- Problem ele alınır
- Problemin çözümü bulunur
- Çözüme dair adımlar algoritmamızı oluşturacaktır
- Algoritma problemin çözümüne yönelik adımlardır, o adımların akışıdır.

Örnek Bir Algoritma: Elektronik kasadan alışveriş (Çebi, 2006)

Elektronik kasiyerin ürün satma adımları aşağıdaki şekilde olacaktır

1. Satın alınan ürüne bak
2. Ürün fiyatını bul
3. Müşteriden parayı al
4. Alınan paradan ürün fiyatını çıkar ve para üstünü bul
5. Para üstünü ver

Sözdekodlar

- Algoritma geliştirmede kullanılan dillerden biri sözde kodlardır (pseudocode)
- Herkesin anlayabileceği ve rahatlıkla bir programlama diline çevrilebilecek basit komutlardan oluşan bir dildir
- Sözdekodun temel işlevi program geliştirmeye geçmeden algoritmayı oluşturmak ve üzerinde tartışabilmektir.
- Örnek: ortalama_sozdekod.pdf

Akış şemaları

- Sözdekodlar gibi, problemin kodunu anlaşılır bir şekilde ortaya koymaya yarayan bir araçtır
- Geometrik şekillerden yararlanarak algoritmayı göstermeye yarar
- Basit ve yapısal akış şemaları vardır
- Basit akış şemaları belirli anlamlara sahip geometrik şekillerden yararlanır: (Özkan, 2003)
  - İşlemleri gösteren simgeler
  - Veri giriş ve çıkış işlemlerini gösteren simgeler
  - Verilerin saklanmasıyla ilgili simgeler
- Örnek: ortalama_akissemasi.pdf

Program nedir? (Kaynak: Elkner, 2010)

- Program, hesaplamayı gerekleştirmek için gereken birbirini izleyen yönergelerden (komutlardan) oluşan yapıdır.
- Hesaplama matematiksel olabilir (bir eşitlikler sisteminin çözülmesi veya bir polinomun kökünü bulmak)
- Sembolik bir hesaplama olabilir (bir belge içinde metin arama ve değiştirme veya program derleme)
- Temel yönergeler aşağıdakilerdir
  - girdi: Klavyeden, dosyadan veya başka bir aygıttan veriyi alma.
  - çıktı: Ekranda veriyi görüntüleme veya veriyi bir dosya ya da başka bir aygıta gönderme.
  - matematik: Toplama, çarpma gibi bazı temel matematiksel işlemleri gerçekleştirme.
  - koşullu yürütme: Belirli durumlar için sınama yapma ve uygun cümle sırasını çalıştırma.
  - tekrarlama: Bazı eylemleri genellikle ufak tefek bazı değişikliklerle tekrar tekrar yürütme.
- Şu ana kadar kullandığımız her program, ne kadar karmaşık olursa olsun, yukarıdakilere benzeyen komutlarla yapılmıştır

Programlama Dilleri

- Çözülecek problemin niteliğine göre seçilir
- Programlama dillerinin hepsi simgeseldir, bilgisayarın anlayacağı işlemleri insanların da anlayabileceği simgelerle ifade ederler
- Simgeler (kodlar) derleyici yardımıyla bilgisayarın anlayabileceği bir biçime (makine kodu, dili) çevrilirler
- Simgesel kodlar kaynak kod, makine diline çevrilmiş kod nesne kodu olarak adlandırılır
- Programlama dilleri şu şekilde sınıflandırılmaktadır:
  - Makine dilleri: Doğrudan makine kodlarından oluşur
  - Assembly dilleri: Anımsatıcı simgelerin kullanıldığı dillerdir
  - Yüksek düzey diller: Komutlar yapılacak işlemin yapılacağı İngilizce karşılığını anımsatıcı kodlardır
  - Dördüncü kuşak diller: Daha esnek, çoğu kodu otomatik oluşturan dillerdir, az kod ile çok iş

Programlama kavramları

- Değişkenler: Değeri program çalışırken değişen büyüklükler
- Sabitler: Değeri program çalışması boyunca sabit kalan büyüklükler
- Atama işlemleri: Bir değişkene farklı değerleri temsil edebilmesi için farklı değerler yerleştirilmesi/atanması
- Veri türleri: Verinin içerdiği bilgiyi belirlemek için kullanılan türlerdir (tamsayılar, gerçel sayılar, kayan noktalı sayılar, mantıksal değerler, diziler/katarlar)
- Veri yapıları: Veriyi problemin yapısına göre düzenlemede kullanılan ve üzerinde işlemler yapılabilen yapılardır (diziler, listeler, yığıtlar, kuyruklar, ağaçlar)
- Kontrol yapıları: Program içerisinde yerine getirilen işlemlere ilişkin kendi aralarında nasıl birleştirileceğini belirleyen yapılardır
  - Sıra: birbiri ardına sıralanan bir dizi işlemdir
  - Seçim: Belirli bir koşulun sağlanıp sağlanmaması durumuna göre iki sıradan/işlemden birinin seçilmesidir
  - Tekrar: Belirli işlemleri belirli bir koşul sağlandığında art arda tekrarlama işlemidir
- Yordam ve fonksiyonlar: Bir uygulamanın aşamalarını farklı

C Programlama Dili

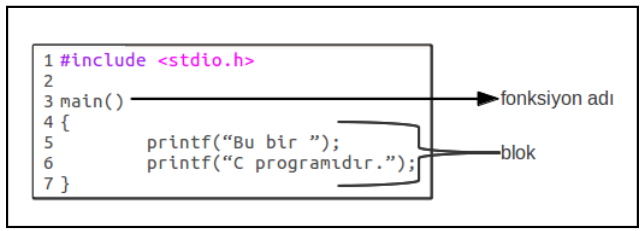
- Genel amaçlı, orta düzeyde, yapısal bir programlama dilidir (Bingül, 2011)
- 1972 yılında Dennis Ritchie tarafından Bell Telefon Labaraturvarında tasarlanmıştır.
- Bir çok alanda kullanılmaktadır
- Özellikle C dili ile geliştirilen UNIX işletim sistemiyla yaygınlığı artmıştır
- Günümüzdeki bazı diller C programlama dilinden esinlenmiştir (C++, Java, JavaScript, PHP, C#)
- Taşınabilir bir dildir, ANSI standartında uygun kodlandığı zaman ANSI destekleyen derleyicilerde derlenir, ANSI destekleyen işletim sistemlerinde çalışır

Neden C? (Bingül, 2011)

- Güçlü ve esnek bir dildir. C ile işletim sistemi veya derleyici yazabilir, kelime işlemciler oluşturabilir veya grafik çizebilirsiniz.
- İyi bir yazılım geliştirme ortamına sahiptir.
- Özel komut ve veri tipi tanımlamasına izin verir.
- Taşınabilir bir dildir.
- Gelişimini tamamlamış ve standardı oluşmuş bir dildir.
- Yapısal bir dildir. C kodları fonksiyon olarak adlandıralan alt programlardan oluşmuştur.
- C++, Java, JavaScript, JavaApplet, PHP, C#, ... gibi diller C dilinden esinlenmiştir.

Programlama yapısı

- C programları bir veya daha fazla fonksiyondan oluşur (çalışabilmesi için mutlaka main() fonksiyonu barındırmalıdır)
- Her fonksiyon bir veya daha fazla deyim içerir
- Fonksiyonlar programın başka bir yerinden çağrılabilir
- Fonksiyonlar C bloklarından oluşur
- Her bir deyim program çalıştırıldığında belirli bir eylemi yerine getirir. Deyimler işlemleri yerine getiren komutlardır.
- Tüm C deyimleri ; (noktalı virgül) işareti ile sonlandırılır
- Örnek: basit.c



main() fonksiyonu

- C programımızda mutlaka yer alması gereken fonksiyondur, özellikle çalıştırabilmek istiyorsak
- Program yürütülmeye başladığında ilk çağrılan fonksiyon main() fonksiyonudur
- C programlarının ana fonksiyonudur
- Kendi fonksiyonlarımızı nasıl hazırlayacağımız ileriki derslerde (6. ve 7. haftalar) anlatılacak
- C programlama dilinde kendi fonksiyonlarımız dışında kullanabileceğimiz hazır kütüphane fonksiyonları da bulunmaktadır
- Örneğin printf() fonksiyonu hazır bir kütüphane fonksiyonudur. Ekranda bir şeyler göstermek için kullanılır

```c
/* main() fonksiyonu */
/* Kaynak: Bingul, 2003 */
#include <stdio.h>// kullanilan fonksiyonlari barindiran baslik dosyalari
main (void) // Ana fonksiyon
{// ana fonksiyon bloku baslangici
  int num; // tamsayi turunde degisken tanimi
  num = 100; // degiskene 100 degeri ataniyor
  printf("Sayı = %d",num); // printf ile ekranda sayiyi gosteriyoruz
}// ana fonksiyon blok bitisi
```

C önişlemcisi

- Derleyicinin kaynak kodunu denetlemesi için önişlemci emirleri vardır
- Bu emirler # işareti ile başlar
- En çok kullanılan emirler #include ve #define emirleridir
  - #include: Bir kaynak dosyasını programa eklemek için kullanılır, özellikle kütüphaneleri ve içerdikleri fonksiyonları kullanabilmek için kullanılır. C programına başlık dosyaları dahil edilir (.h uzantılı). Örnek: stdio.h
  - #define: Sembolik sabilerin tanımlanması için kullanılır. Örnek: #define SON 50

Açıklama satırları

- Programımıza kodla ilgili, programla ilgili yorumlarımızı, açıklamalarımızı açıklama satırlarıyla ekleriz
- C programlama dilinde iki şekilde açıklamalar eklenebilir:
  - // işaretleriyle, işaretten sonra
  - /* */ işaretlerinin arasında satırlara da yayılabilen blok şeklinde

```
 1 /*
 2  * aciklama ornegi
 3  * bu buyuk aciklama ornegidir
 4  */
 5
 6 // kullanilan fonksiyonlari barindiran baslik dosyalari
 7 #include <stdio.h>
 8 main (void) // Ana fonksiyon
 9 {// ana fonksiyon bloku baslangici
10   //bu tek satir aciklama ornegidir
11   // printf ile ekranda sayiyi gosteriyoruz
12   printf("Sayı = %d\n",100);
13 }// ana fonksiyon blok bitisi
```

Anahtar sözcükler

- C dilinde 32 adet anahtar sözcük vardır

Veri tipi	Bellek sınıfı	Deyim	İşleç
char	auto	break	sizeof
const	extern	case	
double	register	continue	
enum	static	default	
float	typedef	do	
int		else	
long		for	
short		goto	
signed		if	
struct		return	
union		switch	
unsigned		while	
void			
volatile			

Temel veri türleri

- Değişken: program içinde kullanılan değerlere bellek üzerinde açılan alanlardır. Değişkenler bir değişken ismi ve değişkende tutulacak olan veri türüyle tanımlanır
- C programlama dilinde kullanılacak değişkenler kullanılmadan önce tanımlanmalıdır. C programlama dilinde var olan temel veri türleri şunlardır:
  - char : karakter veriler
  - int : tamsayı veriler
  - float : tek duyarlıklı kayan noktalı sayılar
  - double : çift duyarlıklı kayan noktalı sayılar
  - void : değer içermeyen veriler
- Tanımlarken *veri_türü değişken_ismi;* şeklinde tanımlanır (*int sayi;*). Bir başlangıç değeri vereceksek atama işleci kullanılır: *int sayi=100;*

```
1 /*
2  * tamsayi degisken tanimlama ornegi
3  */
4 #include <stdio.h>
5
6 main (void) // Ana fonksiyon
7 {// ana fonksiyon bloku baslangici
8   // printf ile ekranda sayiyi gosteriyoruz
9   int i;
10  for (i=0; i<50; i++)
11        printf("%d\n",i);
12 }// ana fonksiyon blok bitisi
```

Değişken Türleri

- Farklı amaçlara göre farklı değişken türleri vardır:
  - Yerel değişkenler: bildirildiği blokta/fonksiyonda geçerli değişkenlerdir
  - Genel değişkenler: program içindeki tüm fonksiyonlar/bloklarda geçerli olan değişkenlerdir
  - **extern** değişkenler: farklı programlarda geçerli olabilmesi için dışa aktarılan değişkenlerdir
  - **static** değişkenler: değişken değeri blok/fonksiyon bitse de korunsun istiyorsak
  - **auto** değişkenler: gerektiğinde oluşturulur, işi bitince ortadan kaldırılırlar
  - **register** değişkenler: yazmaç alanlarını kullanan değişkenler

Sabitler

- Daha önce *#define* ile sembolik sabit tanımlamıştık, sabit değişken tanımlaması ise farklı şekilde yapılmaktadır: *veri_türü const değişken_adı = değeri*
- Şu sabit türleri vardır:
  - Tamsayı sabitler (int)
  - Kayan noktalı sabitler (float)
  - Karakter sabitler (char)
  - Karakter dizisi (String) sabitler (char[])

```
1 main (void)
2 {
3   char const erkek = 'E';
4   int const standart = 120;
5   float const pi = 3.14;
6 }
```

Temel Giriş/Çıkış

- Standart giriş ve çıkış işlemleri için kullanılması gereken kütüphane stdio.h kütüphanesidir
- Standart giriş birimi klavye, çıkış birimi ekrandır.
- Standart çıkış birimine çıkışlar için *printf()* fonksiyonu kullanılabilir: *printf(kontrol_karakterleri, argüman_listesi)*
- Kontrol karakterleri çıkışı yapılan verinin biçimlendirilmesinde kullanılır. Bu karakterler % işaretiyle başlar:
- %c (işaretsiz karakter), %s (karakter dizisi), %d veya %i (işaretli ondalık sayı), %u (işaretsiz ondalık sayı), %o (işaretsiz sekizli sayı), %x (işaretsiz onaltılı sayı), %e (çift duyarlıklı sayı), %f (tek duyarlıklı sayı)
- %n (satır atlama, yeni satır), %b (imleci sola kaydırma), %f (sayfa atlama), %r (satır başı), %t (tab boşluğu)
- Örnek: 9printf.c

Temel Giriş/Çıkış

- Temel giriş işlemi için - klavyeden veri alma - *scanf()* fonksiyonu kullanılabilir
- *scanf(kontrol_karakterleri, argüman_listesi);*
- Örnek: 10scanf.c

İşleçler

- Aritmetik, mantıksal ve karşılaştırma işlemleri için işleç simgeleri kullanılır
- C programlarında başlıca işleç türleri şunlardır:
  - Aritmetik işleçler (toplama, çıkarma, çarpma, bölme, arttırma, azaltma)
  - Mantıksal işleçler (ve, veya, değil)
  - Karşılaştırma işleçleri (eşit, eşit değil, büyük, küçük, büyük veya eşit, küçük veya eşit)
  - Basit atama işleci =

Aritmetik işleçler

- Toplama $+$, Çıkarma $-$, Çarpma $*$, Bölme / işleç simgeleriyle gerçekleştirilir
- Bölme işlemindeki kalan için % işleci kullanılır
- $++$ ve $--$ işleçleri sırasıyla bir arttırma ve bir eksiltme için kullanılır
- Aritmetik işlemli atamalar da var $+=$, $*=$, $-=$, $/=$
- Örnek: 11aritmetik.c

Mantıksal işleçler

- && ve
- || veya
- ! değil
- Örnek: 12mantiksal.c

Karşılaştırma işleçleri

- == eşit
- eşit değil (! =)
- > büyük
- < küçük
- >= büyük veya eşit
- <= küçük veya eşit
- Örnek: 12mantiksal.c

İşlem Önceliği

- İşleçlerin bir işlem önceliği vardır. Program geliştirirken bu işlem önceliğini bilmek önemlidir.
- Yanlış yapmamak için, umulmadık buglarla karşılaşmamak için her zaman parantez kullanın!
- Örnek: 13islemonceligi.c

OPERATÖR ÖNCELİK SIRASI

DÜŞÜK	|	^	&	<< >>	+ -	* / %	! ~ - ++ --	( )	YÜKSEK

İLİŞKİSEL ve MANTIKSAL OPERATÖR ÖNCELİK SIRASI

DÜŞÜK	||	&&	== !=	> >= < <=	!	YÜKSEK

Kaynakça

- Y. Özkan, C ile Programlama, Alfa Yayınları, 2003
- J. Elkner, A. B. Downey, C. Meyers, Bilgisayar Bilimcisi gibi Düşünmek: Python ile Öğrenme (2. baskı, Çeviren: T. E. Kalaycı), 2010, http://yzgrafik.ege.edu.tr/~tekrei/dersler/bbgd_p/
- Viki, Algoritma, http://tr.wikipedia.org/wiki/Algoritma
- A. Bingül, C Programlama Dili'ne Giriş, http://www1.gantep.edu.tr/~bingul/c/index.php?ders=1
- Ç. Çebi, C Programlama Dersi - I, http://www.cagataycebi.com/programming/c_programming/c_programm
- F. Kadifeli, A. C. C. Say, M. U. Çağlayan, C Programlama Dili, 2007, http://www.kadifeli.com/fedon/stdcprtr.php
- Ç. Çebi, C Programlama Dersi - VI, http://www.cagataycebi.com/programming/c_programming/c_programm