Sistem Programlama

Ders 9

Dr. Öğr. Üyesi Mehmet Dinçer Erbaş
Bolu Abant İzzet Baysal Üniversitesi
Mühendislik Fakültesi
Bilgisayar Mühendisliği Bölümü

Binary I/O

- Buraya kadar gördüğümüz fonksiyonlar satır veya karakter okuyabiliyor.
- Bazı durumlarda bütün bir yapıyı okumamız ve yazmamız gerekebilir.
  - Bunu sizce getc veya fputs ile yapabilir miyiz?
- Aşağıdaki fonksiyonlar ile binary I/O yapabiliriz.

```
#include <stdio.h>

size_t fread(void *restrict ptr, size_t size, size_t nobj, FILE
*restrict fp);

size_t fwrite(const void *restrict ptr, size_t size, size_t nobj,
FILE *restrict fp);
```

Dönüş: okunan veya yazılan nesne sayısı.

Binary I/O

- Bu fonksiyonların iki kullanımı vardır
  - Bir dizinin okunması ve yazılması için kullanılabilir. Örneğin 10 elemanlı bir dizinin 2 ile 5 arası elemanlarını yazdırabiliriz.

```
float data[10];
if(fwrite(&data[2], sizeof(float), 4, fp) != 4
    err_sys("fwrite error");
```

  - Bir yapıyı okuyabilir ve yazabiliriz.

```
struct {
    short count;
    long total;
    char name[NAMESIZE];
} item;
if(fwrite(&item, sizeof(item), 1, fp) != 1
    err_sys("fwrite error");
```

Binary I/O

- Binary I/O ile ilgili temel sorun sadece aynı sistem üzerinde yazma ve okuma işlemi yapılabilmesidir.
  - Bir yapı içerisindeki ofset değeri farklı derleyiciler ve sistemler üzerinde, hizalama yöntemi nedeniyle, farklılık gösterebilir.
  - Multibyte tam sayılar ile noktalıklı sayıların saklanma formatı farklı makine mimarilerinde farklılık gösterir.

Formatlı I/O

- Formatlı çıktı 4 printf fonksiyonu ile yapılabilir.

```
#include <stdio.h>

int printf(const char *restrict format, ...);

int fprintf(FILE *restrict fp, const char *restrict format, ...);
```

Dönüş: OK ise çıktı yapılan karakter sayısı, hata ise negatif değer.

```
int sprintf(char *restrict buf, const char *restrict format, ...);

int snprintf(char *restrict buf, size_t n, const char *restrict
format, ...);
```

Dönüş: OK ise diziye yazılan karakter sayısı, hata ise negatif değer.

  - Format kısmı geri kalan argümanların ne şekilde düzenleneceğini ve sonuç olarak gösterileceğini belirler.
  - Dönüştürme tanımı 4 opsiyonel parça içerir.
    - %[flags][fldwidth][precision][lenmodifier]convtype

Formatlı I/O

- Bayraklar

Bayrak	Tanım
‘	Binlik olarak tamsayıyı ayır.
-	Çıktıyı sola yapıştır.
+	+, - değerini yazdır
(boşluk)	+, - yok ise başa boşluk koy.
#	Alternatif forma çevir (örneğin 0x var ise onaltılı olarak ayarla)
0	Başa boşluk yerine 0’lar ekle.

Formatlı I/O

- fldwidth parametresi çevirim için minimum genişlik alanı belirler.
  - Çevirim sonrası daha az karakter var ise sona boşluk eklenir.
  - Alan genişliği negatif olmayan tam sayı veya * karakteridir.
- precision parametresi
  - Tam sayı çevirimleri için minimum basamak sayısını belirler.
  - Noktalıklı sayılar için noktanın sağında kalan basamak sayısını belirler.
  - Metin çevirimleri için maksimum byte sayısını belirler.
  - Precision parametresi . ve onu takip eden negatif olmayan bir tamsayı veya * karaktedir.
- lenmodifier parametresi argümanın uzunluğunu belirler.
  - Muhtemel değerler sonraki slaytta gösterilmiştir.

Formatlı I/O

Uzunluk değişkeni	Tanım
hh	signed veya unsigned char
h	signed veya unsigned short
l	signed veya unsigned long veya wide character
ll	signed veya unsigned long long
j	intmax_t veya uintmax_t
z	size_t
t	ptrdiff_t
l	long double

Formatlı I/O

- convtype değişkeni opsiyonel değildir. Bu değişkenin argümanın ne şekilde yorumlanacağını belirler.

Conversion type	Description
`d,i`	signed decimal
`o`	unsigned octal
`u`	unsigned decimal
`x,X`	unsigned hexadecimal
`f,F`	`double` floating-point number
`e,E`	`double` floating-point number in exponential format
`g,G`	interpreted as f, F, e, or E, depending on value converted
`a,A`	`double` floating-point number in hexadecimal exponential format
`c`	character (with `l` length modifier, wide character)
`s`	string (with `l` length modifier, wide character string)
`p`	pointer to a `void`
`n`	pointer to a signed integer into which is written the number of characters written so far
`%`	a % character
`C`	wide character (XSI option, equivalent to `lc`)
`S`	wide character string (XSI option, equivalent to `ls`)

Formatlı I/O

- Aşağıdaki dört farklı printf ailesine ait fonksiyon önceki dört fonksiyona benzer. Ancak argüman listesi yerine arg tipinde değişken alır.

```
#include <stdarg.h>

#include <stdio.h>

int vprintf(const char *restrict format, va_list arg);

int vfprintf(FILE *restrict fp, const char *restrict format,
va_list arg);
```

Dönüş: OK ise çıktı yapılan karakter sayısı, hata ise negatif tamsayı.

```
int vsprintf(char *restrict buf, const char *restrict format,
va_list arg);

int vsnprintf(char *restrict buf, size_t n, const char *restrict
format, va_list arg);
```

Dönüş: OK ise dizi içerine saklanan karakter sayısı, hata ise negatif tamsayı

Formatlı I/O

- Formatlı girdi üç scanf fonksiyonu ile yapılır.

```
#include <stdio.h>

int scanf(const char *restrict format, ...);

int fscanf(FILE *restrict fp, const char *restrict format, ...);

int sscanf(const char *restrict buf, const char *restrict
format, ...);
```

Dönüş: OK ise değer atanan değişken sayısı, hata veya değer atama öncesi dosya sonuna gelinirse EOF.

  - scanf ailesi fonksiyonları girdi metnini parçalar ve karakter dizilerini verilen tipte girdilere çevirir.
  - Formatı takip eden argüman ismi alınan girdinin nereye yazılacağını belirtir.
  - Aşağıda görüldüğü gibi opsiyonel olan üç farklı parametre vardır.

%[*][flwidth][lenmodifier]convtype

Formatlı I/O

- Baştaki * karakteri çevirimi engeller. Girdi geri kalan parametrelere göre çevrilir ancak sonuç bir argümanda saklanmaz.
- fldwidth parametresi maksimum alan büyüklüğünü karakter sayısı olarak tanımlar.
- lenmodifier parametresi dönüştürme sonucu oluşturulan argümanın büyüklüğünü belirler.
  - printf deki uzunluk değişkenleri scanf fonksiyonu için geçerlidir.
- Convtype değişkeni printf'deki değişkene benzer. Ancak bazı farklılıklar vardır.

Formatlı I/O

Conversion type	Description
`d`	signed decimal, base 10
`i`	signed decimal, base determined by format of input
`o`	unsigned octal (input optionally signed)
`u`	unsigned decimal, base 10 (input optionally signed)
`x,X`	unsigned hexadecimal (input optionally signed)
`a,A,e,E,f,F,g,G`	floating-point number
`c`	character (with `l` length modifier, wide character)
`s`	string (with `l` length modifier, wide character string)
`[`	matches a sequence of listed characters, ending with `]`
`[^`	matches all characters except the ones listed, ending with `]`
`p`	pointer to a `void`
`n`	pointer to a signed integer into which is written the number of characters read so far
`%`	a % character
`C`	wide character (XSI option, equivalent to `lc`)
`S`	wide character string (XSI option, equivalent to `ls`)

Formatlı I/O

- printf ailesinde olduğu gibi scanf fonksiyonlarının da argüman listesi yerine arg değişkeni alan versionları vardır.

```
#include <stdarg.h>

#include <stdio.h>

int vscanf(const char *restrict format, va_list arg);

int vfscanf(FILE *restrict fp, const char *restrict format,
va_list arg);

int vsscanf(const char *restrict buf, const char *restrict
format, va_list arg);
```

Dönüş: OK ise değer atanan değişken sayısı, hata veya değer atama öncesi dosya sonuna gelinirse EOF.