Değişken Tipleri, Tip Çevirme ve İşlemler

Ders 3

Tekrar

- Sabitler (Constants)
  - Sayısal (numerical literals): 3, 33, 435, 0.4858, -5, -0.5
  - Metinler (string literals): "Hello", "Medeniyet"
  - Karakter sabitler: 'A', '3' ,'t', '£', 'H', 'e', 'l', 'l', 'o'
- Değişken – hafızada saklı bir değere verilen referans
- Veri tipi
  - Hafızadaki değişkenin hacmini belirler.
  - Değişkenin hangi değerleri alabileceğini ve değişkenin üzerinde hangi işlemlerin yapılabileceğini belirler.

Değişken Beyanı (deklarasyon)

- degiskenin_tipi degiskenin_ismi;
- Değişken isimlendirmede ki kurallar:
  - Harf, sayı veya _ sembolü kullanılabilir.
  - Harfle veya _ ile başlamalıdır.
  - (int, char) gibi anahtar kelimeler değişken ismi olarak kullanılamaz.
  - printf tanımlayıcı (identifier)dır, anahtar kelime değildir.
  - Tanımlayıcı (an identifier): değişken, fonksiyon vb. ne verilen isimlerdir.
  - C fonksiyon isimleri ve sabitler değişken ismi olarak kullanılmaz.
  - Değişken isimlerinde büyük küçük harflere dikkat ediniz:
    - int x; int X; farklı iki değişken tanımlar.

Değişken İsimlendirme

- int my$dollar;
- int total_count;
- int s2;
- int 2nd;
- int char;
- int _2nd;

Değişken İsimlendirme

- int my$dollar; (yanlış: $ sembolü kullanılamaz)
- int total_count; (doğru)
- int s2; (doğru)
- int 2nd; (yanlış)
- int char; (yanlış)
- int _2nd; (doğru)

Veri Tipleri ve Hacmi

- Sayısal (int, float, double)
- Karakter (char)
- Sayısal tipler işaretli (signed int) veya işaretsiz (unsigned int) olabilirler

```
printf("\t%d\t%d\t%d\t%d\n",
       sizeof(char), sizeof(int), sizeof(float), sizeof(double));
```

Değişken Başlatma

```
int a;
char c;
a = 3;
c = 'w';
```

```
float pi = 3.14;
int angle = 360;
char ismimin_ilk_karakteri = 'A';
```

```
int x, b = 5, c = 50;
float f1 = 2.35, f2 = 1.223;
x = 6;
```

Printf (Print Formatted)

- Printf (format metni, argument1, argument2,….)

```
printf("Renk %s, sayi1 %d, sayi2 %d,  float %5.2f, karakter %c.\n",
      "red", 123456, 89, 3.1415, 'A');
```

- Format metni:
  - "Renk %s, sayi1 %d, sayi2 %d, float %5.2f, karakter %c.\n"
- Argümanlar
  - "red"
  - 123456,
  - 89
  - 3.1415
  - 'A'

Değişken Değerlerini Yazdırma

```
char c1 = 'h';
char c2 = 'e';
char c3 = 'l';
char c4 = 'l';
char c5 = 'o';
int angle = 230;
float pi = 3.14;

printf("%c%c%c%c%c sayisal değişkenlerimin değerleri %d ve %f.\n",
c1, c2, c3, c4, c5, angle, pi);
```

Değişkenlere Değer Okuma

```
char m1, m2, m3, m4, m5;

printf("Lutfen mesajinizin ilk karakterini giriniz:");
scanf("%c", &m1);

printf("Lutfen mesajinizin ikinci karakterini giriniz:");
scanf("%c",&m2);

printf("Lutfen mesajinizin ucuncu karakterini giriniz:");
scanf("%c", &m3);

printf("Lutfen mesajinizin dorduncu karakterini giriniz:");
scanf("%c", &m4);



printf("Lutfen mesajinizin besinci karakterini giriniz:");
scanf("%c", &m5);

printf("Mesaj alindi. %c%c%c%c%c  diyorsunuz.\n", m1, m2, m3, m4, m5);
```

Aritmetik İşlemler

- İşlem – bir işlem, 1-3 tane değişken veya sabitler kullanılarak yapılır. Örneğin, toplama, çıkarma vs.

Arithmetic Operator	Meaning	Examples
+	addition	5 + 2 is 7 5.0 + 2.0 is 7.0
-	subtraction	5 - 2 is 3 5.0 - 2.0 is 3.0
*	multiplication	5 * 2 is 10 5.0 * 2.0 is 10.0
/	division	5.0 / 2.0 is 2.5 5 / 2 is 2
%	remainder	5 % 2 is 1

Aritmetik İşlemler

- + toplama
  - x = 3 + 2; /*sabitler*/
  - y + z; /*değişkenler*/
  - x + y + 2; /*her ikisi*/
- - çıkarma
  - 3 - 2; /*sabitler*/
  - x = y - z; /*değişkenler*/
  - y = 2 - z; /*her ikisi*/
- * çarpma
  - x = 3*2; /*sabitler*/
  - x = y*z; /*değişkenler*/
  - x*y*2; /*her ikisi*/

Aritmetik İşlemler

- C de kullanılan kısaltmalar
  - x += y +2;
  - x -= y +2;
  - x *= y +2;
  - x /= y +2;
- Azaltma ve artırma işlemleri
  - int x, y, z, f, k,
  - y = ++x; alttaki ile aynıdır:
    - x = x + 1;
    - y = x;
  - z = --x; alttaki ile aynıdır:
    - x = x-1;
    - z = x;
- f = x++; alttaki ile aynıdır:
  - f = x;
  - x = x + 1;
- k = x--; alttaki ile aynıdır:
  - k = x;
  - x = x – 1;

Aritmetik İşlemler

- Aşağıdaki programın çıktısı nedir?

```c
#include <stdio.h>

int main(){
  int x = 1, y = x++, z = ++x;
  printf("x = %d, y = %d, z = %d\n", x, y, z);

  return 0;
}
```

Tam Sayılarla Bölme İşlemi

`3 / 15 = 0`	`18 / 3 = 6`
`15 / 3 = 5`	`16 / -3` varies
`16 / 3 = 5`	`0 / 4 = 0`
`17 / 3 = 5`	`4 / 0` is undefined

Bir İfadenin Tipi

- Aritmetik işlemlerde farklı tipten veriler olduğunda; küçük tipli veri büyük tipli veriye çevrilir.

```
int i = 1;
float f = 3.25 + i;
```

  - İşlemde tam sayı i float yapıldı. Bu i nin değerini işlem dışında değiştirmez.

- 16 / 3 = ?
- 16.0 / 3 = ?
- 16 / 3.0 = ?

Tip Çevirme

- Bir ifadenin tipini, parantez içinde çevirmek istediğiniz tipi önüne yazarak değiştirebilirsiniz.

```
int x = 5, y = 6;
double z0, z1, z2, z3;
z0 = y/x;
z1 = (double) y/x;
z2 = (double) (y/x);
z3 =  y/(double) x;

printf ("z0 = %f, z1 = %f, z2 = %f, z3 = %f.\n", z0, z1, z2, z3);
```

Tip Çevirme

- Bir ifadenin tipini önüne parantez için çevirmek istediğiniz tipi yazarak değiştirebilirsiniz.

```
float y = 16.054;
int x;
x = (int) (y/3);
```

```
int c = 65;
printf("%c", (char)c);
```

İlişkisel İşlemler

- >, >=, <, <=
- Misaller
  - (5 > 4)
    - 5, 4 den büyük mü? Evet, o zaman (5 > 4) = 1.
  - (5 < 4)
    - 5, 4 den küçük mü? Hayır, o zaman (5 < 4) = 0.

```
int x = 5, y = 4;
printf("%d %d %d %d\n", (x<y), (x<=y), (x>y), (x>=y));
```

İlişkisel İşlemler

- == and !=
  - Eşitlik işlemleri
- Misaller
  - (5 == 4)
    - 5, 4'e eşit mi? Hayır, o zaman (5 == 4) = 0.
  - (5 ! = 4)
    - 5, 4 den faklı mı? Evet, o zaman (5 != 4) = 1.

```
int x = 5, y = 4;
printf("%d %d %d %d\n", (x==5), (x!=5), (x>y), (x>=5));
```

Mantiki (logical) İşlemler

- &&
  - Ve İşlemi
  - (a && b) = ?

a	b	(a && b)
0	0	0
0	1	0
1	0	0
1	1	1

- ||
  - Veya işlemi
  - (a || b) = ?

a	b	(a || b)
0	0	0
0	1	1
1	0	1
1	1	1

Mantiki (logical) İşlemler

- !
  - Not (Değil) işlemi
  - !(0), 1 dir.
  - !(1), 0 dır.
  - !(5 < 4)
    - 5, 4 den küçük mü? Hayır, o zaman. !(5 < 4), !(0) olur.
    - !(0): sıfır değil de 1 olur.
- Parentezleri kullanarak farklı işlemleri bir arada kullanabilirsiniz.

```
int x = 5, y = 4;
printf("%d %d\n",
      !((x == y) && (x != y)), ((x != 5) || (x > y + 4)));
```

Girinti (Indentation) ve Birleşik Cümlecikler

- Birden fazla cümlecik {...} kullanılarak birleştirilebilir:
- Buna ayrıca kod bloğu denir.

```
{
    int x = 0;
    printf("computing x\n");
    x = x + 1;
    printf("the new x value is %d \n",x);
}
```

- Girintiler kodumuzun okunmasını kolaylaştırır.

```
{
int x = 0;
printf("computing x\n");
x = x + 1;
printf("the new x value is %d \n",x);
}
```

Değişken Kapsamı

```c
#include <stdio.h>

int main(){
    int x = 1;/* blogun disinda*/

    /* birlesik cumlecikler */
    {
        x = x + 1;
        printf("1.printf: %d\n",x);/* prints x*/
    }

    printf("2.printf: %d\n", x);/* prints x*/
    return 0;
}
```

Değişken Kapsamı

```c
#include <stdio.h>

int main(){

    /* birlesik cumlecikler */
    {
        int x = 1;/* blogun icinde */
        x = x + 1;
        printf("1.printf: %d\n", x);/* prints x*/
    }
    printf("2.printf: %d\n", x);/* prints x*/
    return 0;

}
```

Değişken Kapsamı

```c
#include <stdio.h>

int main(){
    int x = 1;/* blogun disinda */

    /* birlesik cumlecikler */
    {
        int x = 1;/* blogun icinde*/
        x = x + 1;
        printf("1.printf: %d\n", x);/* prints x*/
    }
    printf("2.printf: %d\n", x);/* prints x*/
    return 0;

}
```

Akış Kontrol

- Haftaya
  - Sıralı Çalıştırma
  - Seçmeli Çalıştırma