Değişkenler, Veri Tipleri, Aritmetik İşlemler

Ders 2

İçerik

- C Programlamanın Temelleri
- Değişken Tanımlama (Variable declarations)
- Değişken başlatma (Initialization)
- Veri tipleri (Data Types)
- Formatlı Çıktı
- Aritmetik İşlemler

C Dili

- Anahtar kelimeler(keywords),
- Tanımlamalar (identifiers)
- Sabitler (constants)
- Metinler (string literals)
- veya semboller (symbols)
- Noktalı virgül
- Yorumlar

C Programının Yapısı

- Bir C programı bir veya daha fazla C dosyasından (.c uzantılı dosya), mesela
  - main.c, problem1.c, myprogram.c
- ve header dosyalarından (.h uzantılı) dosyalardan oluşur.
  - math.h, stdio.h, myheader.h

Bir C Dosyasının Yapısı

- **/* dosyanın içeriği ve işlevini anlatan bir yorumla başlanır. */**
- **Opsiyonel: #include** cümlecikleri ve önişleme tanımlamaları sıralanır **(preprocessor definitions)**.
- Fonksiyon prototipi ve onu değişken beyanları takip eder.

```
fonksiyon  tanımlaması
    {
      Function body
    }
Baska bir fonksiyon tanımlaması
    {
     Function body
    }
```

C Programının Parçaları: Yorumlar

- yorumlar: /* … */

```
/* bu
  bir
  çok satırlı
yorumdur. Blok yorum */
```

```
/* Bu
 * bir
 * çok
 * satırlı
 * yorumdur */
```

- Compiler tarafından tamamen gözardı edilirler.
- Dosya içerisinde her yere konulabilirler.
- //tek satır değiştirme yorumları,

```
// puts("A string to print");
```

  - Genelde satırdaki komutların yoruma dönüştürülerek programdan çıkarılması için kullanılır.

C Programının Parçaları:
Önişeleme (preprocessor) Komutları

- hash sembolü (#) ile başlarlar:
  - #include, #define, #ifdef, #undef, #if, vs.
  - #include: kütüphaneden header dosyası eklemek için kullanılır.
    - Header dosyalar: (sabitler) constants, fonksiyonlar (functions), ve diğer beyanları içerirler.
    - #include <stdio.h>
    - stdio.h – C Standard Kütüphanesinin bir parçası.
    - Diğer header dosyaları: ctype.h, math.h, stdlib.h, string.h, time.h

C Programının Parçaları:
main fonksiyonu

- Bir C programında bir tane main() fonksiyonu olmak zorundadır.

```
int main()
{
   Function body (fonksiyonun akisi icin kodlar)

}
```

C Programının Parçaları:
Diğer Fonksiyonlar

- Mainle aynı c dosyasına koyulabileceği gibi başka dosyayada koyulabilirler.

```
int main()
{
   Function body (fonksiyonun akisi icin kodlar)
}

baskaBirFonksiyon()
{
   Function body (fonksiyonun akisi icin kodlar)
}

fonksiyon2()
{
    Function body (fonksiyonun akisi icin kodlar)
}
```

C Programının Parçaları: Fonksiyonlar

- Her fonksiyonun bir ismi var: **main, printf, puts, scanf,** gibi.

- Fonksiyon isminden sonra gelen parantezler argüman listelerini çevirirler.

```
printf("Hi!");          /* Burada "Hi!" bir arguman */
```

- Bir fonksiyon diğer fonksiyondan isminin yazılması ile çağrılır:

```
main() {
    …
    printf("Hi!"); /* "Hi!" is the argument*/
    ...
}
```

- İki fonksiyon birbirleriyle argümanları kullanarak veri haberleşmesi yaparlar.
- Çağıran fonksiyonun verdiği argüman listesi çağrılan fonksiyonda kullanılır.

C Programının Parçaları: Fonksiyonlar

- printf(“Hi! \n”);
- “Hi! \n” karakter string (metin) veya string constant (sabit) olarak adlandırılır.
- \n yeni bir satır için C notasyonudur (escape sequence).
- Diğer kaçış dizileri \n veya \t gibi yazılması zor olan karakterleri ifade edebilmemiz için bir mekanizma sunarlar.

Alıştırma

- Aşağıda bu karakterlerin tam listesi var bunların nasıl çalıştığını deneyebilirsiniz

`\a`	alert (bell) character	`\\`	backslash
`\b`	backspace	`\?`	question mark
`\f`	formfeed	`\'`	single quote
`\n`	newline	`\"`	double quote
`\r`	carriage return	`\`*ooo*	octal number
`\t`	horizontal tab	`\x`*hh*	hexadecimal number
`\v`	vertical tab		

C Programının Parçaları: değişkenler

- Programın giriş verilerini saklamak için kullanılan hafıza hücrelerini temsil ederler.
  - int x = 4;
  - int y = 5;

- Bu değişkenlerde saklı veriler programın çalışması esnasında değişebilirler:
  - x = x*y + 25;
  - + ve * işlemlerdir (operators).

C Programının Parçaları: Cümlecikler (statements)

- Cümlecikler (statements)
  - Bir satırda bulunan ve belirli bir işlev gören koddur.
  - Noktalı virgül bir cümleciği bitirir (yeni bir satır değil)

```
printf(“Hello\n”);
scanf(“enter an integer value for x %d“, x);
printf(“The value of x is %d”, x);
x = x + 12;
```

C Programının Parçaları: İfadeler (expressions)

- Bir değerle sonuçlanan, sembollerin (sabitlerin, değişkenlerin, işlemlerin) bir usüle göre birleşimine denir.

  x*2
  x == 5
  x + y - 12

C Programının Parçaları: Farklı Cümleler(statements)

- İfade cümlecikleri (Expression Statements)
  - x = x + 12;
  - Noktalı virgülle takip edilir

- Birleştirilmiş Cümlecikler (compound statements)
  - {} içerisinde birden fazla cümleciğin birleştirilmesi:

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

- Control Statements

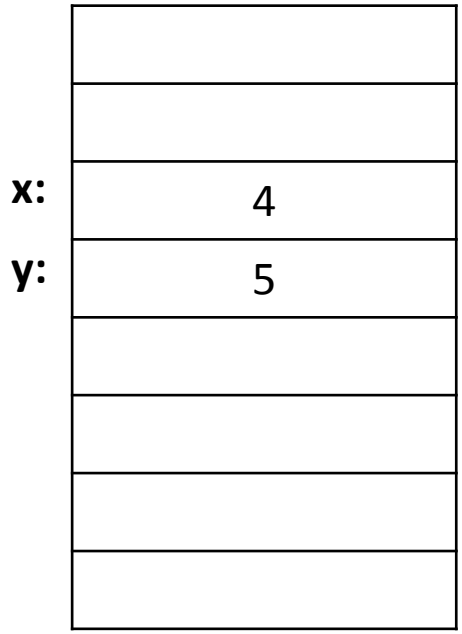
Değişkenler(Variables)

- Değişkenler hafıza hücreleri (memory cells) için verilen referans isimlerdir.
- Program datasını saklamak için kullanılırlar.

```
int x = 4;  int y = 5;
```

  - Burada x ve y değişken isimleri
  - ve = sembolü atama işlemidir.

- Memory de görünüm



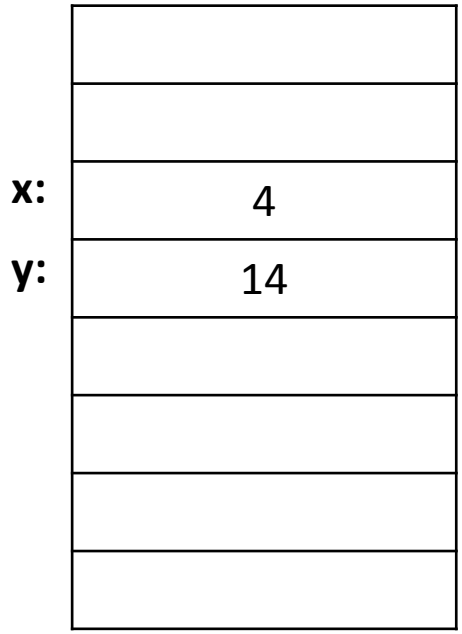
Değişkenler(Variables)

- Değişkenlerde saklı değerler programın çalışması esnasında değişebilirler.

y = y + x + 5;

  - Burada + toplama
  - ve **=** sembolü atama işlemidir

- Memory de görünüm



Değişken Tipi

```
int x = 4;  int y = 5;
```

- Değişkenler üzerinde yapabileceğimiz işlemlerin türlerini değişken tipleri belirler:
  - int sıralanan değişkenlerin tam sayı (integer) olduğunu ifade eder.
  - 32 bit bilgisayarlarda, integer değişkeni -32768 ile +32768 arasında bir değer alır.

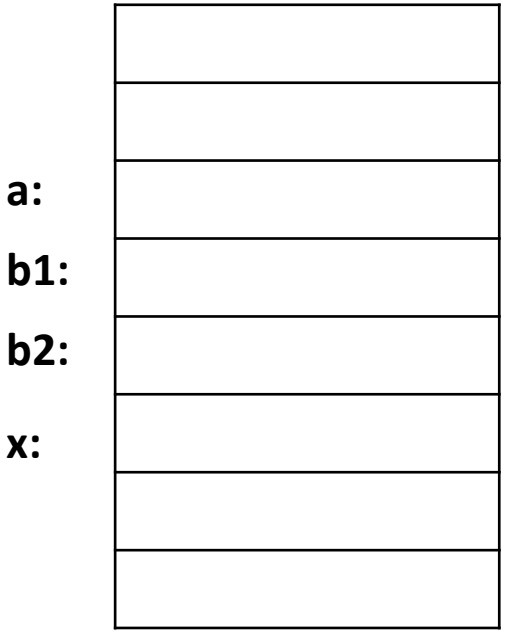
- Değişkenin tipine bağlı olarak derleyici (compiler) hafızada yer ayırır.

  32-bit bilgisayarda int x için, 32 bit hafıza hücresi ayırır.

Değişken Beyanları (Declarations)

- Değişkenin beyan edilmesi
  - tipi ismi;
    - int x; /*int değişken tipi, x değişkenin ismini ifade eder*/

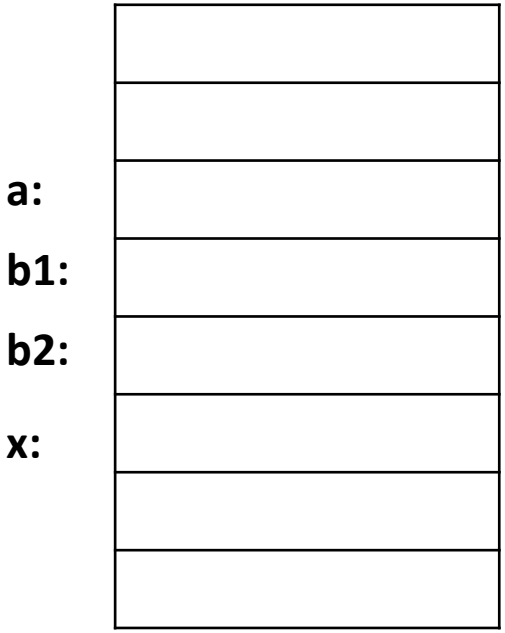
```
int a;
int b1;
int b2;
int x;
```



Değişken Beyanları (Declarations)

- Birden fazla deklarasyon virgülle birleştirilebilir.
  - tipi isim1, isim2, isim3;

```
int a, b1, b2, x;
```



Değişkenin Başalatılması (Initialization)

- Sabitler (**Constants**) değişkenlere verilen değerleri ifade ederler.
  - 3, 3.14, 718111, gibi

- Değişkenler atama işlemi ile başalatılırlar:

```
int a;
a = 3;
```

- **=** sembolü atama işlemi için kullanılır.
- Değişkenin kullanılması için önce beyan edilmiş olması zorunludur.
- C de başlatılmayan veriler varsayım (default) değer alırlar:
  - int a;

Değişkenin Başalatılması (Initialization)

- Değişkeni beyan esnasında da başlatabilirsiniz:

```
int a = 3;
```

- Virgül kullanarak birden fazla değişkeni aynı anda beyan edip başlatabilirsiniz:

```
int a = 3;

int x, b = 5, c = 50;
```

Diğer Değişken Tipleri: Tam Sayılar

Type	Storage size	Value range
char	1 byte	-128 ile 127 veya 0 ile255
unsigned char	1 byte	0 ile 255
signed char	1 byte	-128 ile 127
int	2 veya 4 bytes	-32,768 ile 32,767 veya- 2,147,483,648 ile 2,147,483,647
unsigned int	2 veya 4 bytes	0 ile 65,535 veya 0 ile 4,294,967,295
short	2 bytes	-32,768 ile 32,767
unsigned short	2 bytes	0 ile 65,535
long	4 bytes	-2,147,483,648 ile 2,147,483,647
unsigned long	4 bytes	0 ile 4,294,967,295

Diğer Değişken Tipleri:
Noktalı Sayılar (floating-point numbers)

Type	**Storage size**	**Value range**	**Precision**
float	4 byte	1.2E-38 ile 3.4E+38	6 ondalık basamak
double	8 byte	2.3E-308 ile 1.7E+308	15 ondalık basamak
long double	10 byte	3.4E-4932 ile 1.1E+4932	19 ondalık basamak

Printf ile Formatlı Çıktı
Place Holders

- \n ve \t nin yeni bir satır ve tab boşluğu için olduğunu öğrendik
- printf ile bir değişkenin yazdırılabilmesi için, "%" işareti ile argümanın tipini belirtmemiz gerekir:
- Mesela, "%d" printf'in integer tipte bir argüman beklediğini belirtir:

```
int a = 5;
printf("a nin degeri %d",a);
```

Printf ile Formatlı Çıktı
Place Holders

- “%d” ile printf integer tipte bir argüman bekler:

```
int a = 5, b = 4;
printf("a nin degeri %d ve b nin degeri %d", a, b);
```

- Karakterler için “%c”

```
char c = 'a';
printf("c is %c\n", c);
```

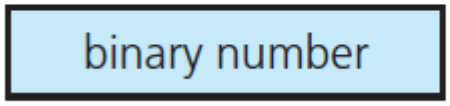
- float ve double için “%f” ve “%lf”

```
double d1 = 0.1; float d2 = 0.5;
printf("d1 is %f and d2 is %f", d1, d2);
```

- Metin için “%s”

int ve double farkı

type `int` format



type `double` format

$$real\ number = mantissa \times 2^{exponent}$$

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

Printf de basamak sayısı

- Çıkışta kaç basamak görüleceği belirtilebilir:

```
printf("%0.5f",0.123456789);
```

- “%0.5f” en önemli 5 basamağın gösterileceğini belirtir.
- Kalan basamaklar son basamağa yuvarlanır.

Karakterler

- C de char -128 ile 127 veya 0 ile 255 arasında bir tam sayı tipidir.
- char kullanılarak karakterlerisaklayabilirsiniz.

```
char a = 'a';
char b = '*';
```

- Aşağıdaki kodun çıktısı nedir?

```
char a1 = 'A';
char a2 = 65;
printf("\n %c %c\n", a1, a2);
```

Karakterlerin ASCII Kodları

Character	ASCII Code
' '	32
'*'	42
'A'	65
'B'	66
'Z'	90
'a'	97
'b'	98
'z'	122
'0'	48
'9'	57

```
printf("\n %c %c\n", a1, a2);/*prints A A */
```

Daha fazla printf

- “%s” kullandığınızda asgari veya maksimum karakter sayısınıda verebilirsiniz.
- Mesela 5 ve 10 karakter yer ayırmak için

```
printf("%5s \t %10s \n","Hello", "Medeniyet");
```

- Eğer karakter sayısı az ise kalan kısım için boşluk koyulur. Fazla karakterler yine yazdırılır.

```
printf("%15s \t %5s \n","Hello", "Medeniyet");
```

Daha fazla printf

- Maksimum karakter sayısı noktadan sonra belirtilir.

```
printf("%.5s \t %.10s \n","Hello", "Medeniyet");
```

- Eğer karakter sayısı az ise kalan kısım için bir şey yapılmaz. Fazla karakterler yazdırılmaz.

```
printf("%.15s \t %.5s \n","Hello", "Medeniyet");
```

UYARI

- Kullandığınız format işareti “%” kesinlikle argüman sayısından fazla olmamalıdır. Program derlenmesine rağmen güvenli değildir.

```
printf("%s \t %9s \n","Hello");
```

scanf

- Klavyeden karakter, string, ve sayısal veri alımı scanf kullanılarak yapılabilir.
- Syntax yapısı printf'e benzemektedir.

```
int age;
scanf("%d",&age);
printf("You are %d years old!\n", age);
```

- Ve karakteri "&" age değişkenin hafıza adresini ifade eder.

scanf

- Keyfi miktarda giriş alınabilir.

```
int year, month, day;

printf("Enter the year: ");
scanf("%d",&year);

printf("Enter the month: ");
scanf("%d",&month);

printf("Enter the day: ");
scanf("%d",&day);

printf("Today is %d\\%d\\%d\n", day, month, year);
```

Aritmetik İşlemler

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

Mathematical Formula	C Expression
1. $b^2 - 4ac$	`b * b - 4 * a * c`
2. $a + b - c$	`a + b - c`
3. $\frac{a + b}{c + d}$	`(a + b) / (c + d)`
4. $\frac{1}{1 + x^2}$	`1 / (1 + x * x)`
5. $a \times -(b + c)$	`a * -(b + c)`

Aritmetik İfadeler

- x ve y nin değişkenler olduğunu farz edelim
- (x + y), (x – y), (x*y), (x/y), (x%y)

- Basit bir atama cümleciği (assignment statement):

  y = x + 3 ∗ x / (y − 4);

- Sayılar(Numeric literals), 3 ve 4 gibi, aritmetik ifadelerde kullanılabilirler

- Aritmetik ve atama
  - x = x + y, x = x - y, x = x*y, x = x/y, x = x% y
  - x += y, x -= y, x *= y, x /= y, x %= y
  - += atama ve işlem (assignment operator)

İşlem Sırası

- İşlem yönü

+, - (sign)	sağdan sola
*, /, %	soldan sağa
+, -	soldan sağa
=, +=, -=, *=, /=, %=	sağdan sola

- Parentezler kullanılarak işlem sırası değiştirilebilir

```
float x = 6, y = 3;
float z1 = 2 + x/3+y;
float z2 = 2 + x/(3 + y);
printf("z1 = %f \t z2 = %f \n",z1, z2);
```

Tam Sayı Bölme

`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

Aritmetik İfadelerin Veri Tipi

- Aritmetik işlemler farklı türden verileri içerdiğinde; değişkenler en büyük veri tipine çevrilirler.

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

  - Yukarıda integer değer floata yükseltilirken verisinin değeri değişmiyor.
- 16 / 3 = ?
- 16.0 / 3 = ?
- 16 / 3.0 = ?

Tip Çevrimi (Type Casting)

- Bir ifadenin tipini, parentezle dönüştürmek istediğiniz tipi yazarak başka bir tipe dönütürebilirsiniz.

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

Vaka Çalışması

- Dönem sonu notunuzu hesaplayarak baş harflerinizle birlikte yazdıran bir program yazınız.
- Dönem sonu notu?
  - **Derecelendirme**
  - Ödev: 10%
  - Lablar: 10%
  - Projeler: 20%
  - Vize: 20%
  - Final: 40%

Vaka Çalışması

- Program girişleri neler?
- Çıkışlar neler?
- Dönem sonu notunu nasıl hesaplıyoruz?
  - Algortimamız ne?