

Bilgisayar Bilimlerine Giriş-II

-6-

BIL 1002

**Dokuz Eylül Üniversitesi, Fen Fakültesi,
Bilgisayar Bilimleri Bölümü**

Bu günkü dersimizin içeriği

- ▶ Karakter Dizilerini (Strings) **Okumak** ve **Yazmak**
- ▶ Karakter Dizilerinin Uzunluğu Bulmak (**strlen**)
- ▶ Karakter Dizilerini Birleştirmek (**strcat**)
- ▶ Karakter Dizisi Kopyalama (**strcpy**)
- ▶ Karakter Dizilerini Karşılaştırmak **strcmp()**
- ▶ Karakter Kütüphanesi **<ctype.h>**
- ▶ String Dönüşüm Fonksiyonları
- ▶ Standart Giriş/Çıkış Kütüphane Fonksiyonları
- ▶ String Kütüphanesindeki String İşleme Fonksiyonları
 - ▶ **bilgilerini içermektedir**

Karakter Dizileri (Strings)

- ▶ Bazı programlama dillerinde karakter dizilerini tutmak için özel veri türleri (*string*) bulunmaktadır.
 - ▶ Ancak C programlama dilinde böyle bir veri türü olmadığı için yerine **karakterlerden oluşan bir boyutlu diziler** kullanılır.
 - ▶ Karakter dizilerine özel olarak, karakter dizilerinin sonuna **sonlandırıcı karakter** olarak adlandırılan bir simge eklenir.
 - ▶ Sonlandırıcı karakter:
 - ▶ Dizin bittiği yeri gösterir,
 - ▶ ASCII tablosunun sıfır numaralı (**'\0'**) karakteridir.
- char** **katar_adı**[**elemansayısı**]

Karakter Dizilerini Okumak ve Yazmak

- ▶ **printf()** ve **scanf()** fonksiyonları diğer tiplerde olduğu gibi formatlı okuma/yazma amaçlı kullanılır.
- ▶ Katar formatı **%s** dir.

```
char str[20];  
...  
scanf("%s",str);  
printf("%s\n",str);
```

Katar okutulurken
& operatörünün
kullanılmadığına
dikkat ediniz.

Karakter Dizilerini Okumak ve Yazmak

- ▶ Bir karakter dizisini klavyeden okumak için C'nin standart `gets()` fonksiyonu kullanılır.
- ▶ `stdio.h` dışında yeni bir kitaplığı C programına dahil etmeye gerek yoktur.
- ▶ Bu fonksiyon her hangi bir indeks tanımlamadan karakter dizilerinin okunmasını sağlar.
- ▶ Okuduğu karakter dizisinin sonuna satır sonu işaretini değil, NULL değerini yerleştirir.

```
gets(string_name);  
puts(string_name);
```

Örnek: Karakter Dizisi Okuma ve Yazma

- Maksimum 50 karakter okuyabilecek bir karakter dizisi tanımlayın. Klavyeden karakter dizisinin okuyun ve ekrana karakterleri yazdırın.
- Tek tek karakterleri yazdırın.

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
int main()
{
    char dizi[50];
    int i;

    printf("Metni giriniz:");
    gets(dizi);
```

```
printf("\nTek tek karakterleri yazdir\n");
    for(i=0;dizi[i];i++)
    {
        printf("%c \n",dizi[i]);
    }
    getch();
    return 0;
}
```

Karakter Dizilerinin Uzunluğu Bulmak (**strlen**)

- ▶ Bazı uygulamalarda bir karakter dizisinin uzunluğunu bulmak gerekebilir.
- ▶ Bir karakter dizisinin uzunluğunu, yani kaç karakter içerdiğini bulmak için C'nin standart **strlen()** fonksiyonu kullanılır.
- ▶ Uzunluk bulunurken, içerdiği en son karakter olan **NULL** karakteri göz önüne alınmaz.
- ▶ Örneğin, karakter dizisi "abc" değerlerini içeriyorsa, strlen() fonksiyonu bu uzunluk olarak "3" değerini döndürür.

strlen(str)

Örnek: Girilen Karakter Dizisinin Uzunluğunu Bulmak

```
#include<stdio.h>
```

```
#include<conio.h>
```

```
#include<string.h>
```

```
int main()
```

```
{
```

```
    char dizi[50];
```

```
    printf("Metni giriniz:");
```

```
    gets(dizi);
```

```
    printf("\nUzunluk:%d",strlen(dizi));
```

```
    getch();
```

```
    return 0;
```

```
}
```


Karakter Dizilerini Birleştirmek (**strcat**)

- ▶ İki karakter dizisini birleştirilerek tek bir karakter dizisi haline dönüştürmek için C'nin **strcat()** fonksiyonu kullanılır.
- ▶ Bu fonksiyon, var olan bir karakter dizisinin sonuna bir başka karakter dizisini ekleyecektir.
- ▶ Örneğin **"abc"** karakter dizisinin sonuna **"def"** karakter dizisi **strcat()** fonksiyonu kullanılarak eklenebilir.


strcat(string1,string2)

Örnek: Girilen karakter dizilerini birleştirme

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
int main()
{
    char m1[50],m2[50];
    printf("1. Metni giriniz:");
    gets(m1);
    printf("\n2. Metni giriniz:");
    gets(m2);
    strcat(m1,m2);
    printf(m1);
    getch();
    return 0;
}
```

Karakter Dizisi Kopyalama (strcpy)

- Atamanın bir karakter dizisine yapılabilmesi için, C'nin standart **strcpy()** fonksiyonu kullanılır.



Bu fonksiyon program çalıştırıldığında, *string2*'nin *string1*'e kopyalanmasını sağlar.

strcpy(*string1*,*string2*);

Örnek: Karakter dizilerini kopyalama

```
#include<stdio.h>
```

```
#include<conio.h>
```

```
#include<string.h>
```

```
int main()
```

```
{
```

```
    char m1[50],m2[50];
```

```
    int i;
```

```
    strcpy(m1,"Algorithms ");
```

```
    strcpy(m2,"and Programming");
```

```
    strcat(m1,m2);
```

```
    printf(m1);
```

```
    getch();
```

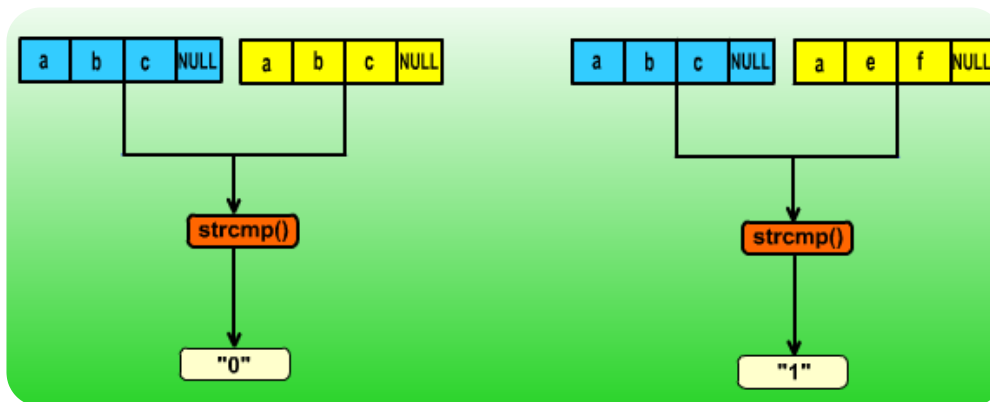
```
    return 0;
```

```
}
```

Karakter Dizilerini Karşılaştırmak

- ▶ İki karakter dizisinin birbirleriyle karşılaştırılarak, içerdiği karakterlerin aynı olup olmadıkları test edilebilir. Bu amaçla **strcmp()** fonksiyonu kullanılır.
- ▶ Karşılaştırma sonucunda, her iki karakter dizisi birbirinin aynı ise "0"; birbirinden farklı ise "1" değeri üretilir. Elde edilen bu değer kullanılarak programın akışı yönlendirilebilir.

strcmp(string1,string2);



Örnek: Karakter dizilerini karşılaştırma

```
#include<stdio.h>
```

```
#include<conio.h>
```

```
#include<string.h>
```

```
int main()
```

```
{
```

```
    char m1[50],m2[50];
```

```
    strcpy(m1,"Algorithms ");
```

```
    strcpy(m2,"and Programming");
```

```
    if (strcmp(m1,m2)!=0)
```

```
        printf("Metinler birbirinden farkli...");
```

```
    else
```

```
        printf("Metinler birbirleriyle ayni...");
```

```
    getch();
```

```
    return 0;
```

```
}
```

Karakter Kütüphanesi

- ▶ Karakter kütüphanesi, karakter verilerini işlemek ve test etmek için kullanışlı bir çok fonksiyon içermektedir. Her fonksiyon argüman olarak, **int** ile temsil edilen bir karakter ya da **EOF** alır.
- ▶ **Not:** Karakter kütüphanesinden fonksiyonlar kullanırken **<ctype.h>** öncü dosyasını programınıza dahil edin.

Prototip	Fonksiyon Tanımı
int isdigit (int c);	c bir rakam ise doğru bir değer, değilse 0 (yanlış) döndürür.
int isalpha (int c);	c bir harf ise doğru bir değer, değilse 0 döndürür.
int isalnum (int c);	c bir harf ya da rakamsa doğru bir değer, değilse 0 döndürür.
int isxdigit (int c);	c onaltılık sistemde bir rakam değeri karakteri ise doğru bir değer, değilse 0 döndürür.
int islower (int c);	c küçük bir harf ise doğru bir değer, değilse 0 döndürür
int isupper (int c);	c büyük bir harf ise doğru bir değer, değilse 0 döndürür.

Prototip	Fonksiyon Tanımı
int tolower (int c);	c bir büyük harf ise, tolower c 'yi küçük harfe çevirerek döndürür, değilse tolower argümanı değiştirmeden döndürür
int toupper (int c);	c küçük harf ise, toupper c 'yi büyük harfe çevirip döndürür, değilse toupper argümanı değiştirmeden döndürür.
int isspace (int c);	c yeni satır('\n'),boşluk(' '),form besleme('\f ')-,satır başı('\r'),yatay tab('\t') ya da dikey tab('\v') karakterlerinden biriyse doğru bir değer, değilse 0 döndürür.
int iscntrl (int c);	c bir kontrol değişkeni ise doğru bir değer, değilse 0 döndürür.
int ispunct (int c);	c boşluk, rakam ya da harften başka bir yazdırma karakteri ise doğru bir değer, değilse 0 döndürür.
int isprint (int c);	c boşluk (==) karakteri de dahil olmak üzere bir yazdırma karakteri ise doğru bir değer, değilse 0 döndürür.
int isgraph (int c);	c boşluk karakteri haricinde bir yazdırma değeri ise doğru bir değer, değilse 0 döndürür.

C'de \f escape

- ▶ https://en.wikipedia.org/wiki/Escape_sequences_in_C
- ▶ [Escape sequences in C - Wikipedia](#)

Not: (?:) koşullu operatörünü kullanarak, test edilen her karakter için operatörle birlikte kullanılan stringlerden hangisinin yazdırılacağına karar verir.

```
#include <stdio.h>
#include <ctype.h>
#include <conio.h>
```

```
int main()
{
    printf( "%s\n%s\n%s\n\n", "isdigit için: ",
        isdigit( '8' )? "8 bir rakamdir" : "8 bir rakam degildir",
        isdigit( '#' )? "# bir rakamdir" :
            "# bir rakam degildir." );
    printf( "%s\n%s\n%s\n%s\n%s\n\n",
        "isalpha için:",
        isalpha( 'A' ) ? "A bir harftir " : "A bir harf degildir",
        isalpha( 'b' ) ? "b bir harftir " : "b bir harf degildir",
        isalpha( '&' ) ? "& bir harftir" : "& bir harf degildir",
        isalpha( '4' ) ? "4 bir harftir " : "4 bir harf degildir" );
    printf( "%s\n%s\n%s\n%s\n%s\n\n", "isalnum için:",
        isalnum( 'A' ) ? "A bir rakam yada harftir" : "A bir rakam yada harf degildir",
        isalnum( '8' ) ? "8 bir rakam yada harftir" : "8 bir rakam yada harf degildir",
        isalnum( '#' ) ? "# bir rakam yada harftir" : "# bir rakam yada harf degildir");
    printf( "%s\n%s\n%s\n%s\n%s\n\n", "isxdigit için:",
        isxdigit( 'F' ) ? "F bir heksadesimal rakamdir" : "F bir heksadesimal rakamdir degildir",
        isxdigit( 'J' ) ? "J bir heksadesimal rakamdir" : "J bir heksadesimal rakamdir degildir",
        isxdigit( '7' ) ? "7 bir heksadesimal rakamdir" : "7 bir heksadesimal rakamdir degildir",
        isxdigit( '$' ) ? "$ bir heksadesimal rakamdir" : "$ bir heksadesimal rakamdir degildir",
        isxdigit( 'f' ) ? "f bir heksadesimal rakamdir" : "f bir heksadesimal rakamdir degildir");
    getch();
    return 0;
}
```

```
isdigit için:
8 bir rakamdir
# bir rakam degildir.

isalpha için:
A bir harftir
b bir harftir
& bir harf degildir
4 bir harf degildir

isalnum için:
A bir rakam yada harftir
8 bir rakam yada harftir
# bir rakam yada harf degildir

isxdigit için:
F bir heksadesimal rakamdir
J bir heksadesimal rakamdir degildir
7 bir heksadesimal rakamdir
$ bir heksadesimal rakamdir degildir
f bir heksadesimal rakamdir
```

```

#include <stdio.h>
#include <ctype.h>
#include <conio.h>
int main()
{
    printf( "%s\n%s\n%s\n%s\n%s\n\n",
        "islower:",
        islower( 'p' ) ? "p bir küçük harftir" : "p bir küçük harf degildir",
        islower( 'P' ) ? "P bir küçük harftir" : "P bir küçük harft degildir",
        islower( '5' ) ? "5 bir küçük harftir" : "5 bir küçük harf degildir",
        islower( '!' ) ? "!! bir küçük harftir" : "!! bir küçük harf degildir");
    printf( "%s\n%s\n%s\n%s\n%s\n\n",
        "isupper:",
        isupper( 'D' ) ? "D bir büyük harftir" : "D bir büyük harf degildir",
        isupper( 'd' ) ? "d bir büyük harftir" : "d bir büyük harf degildir",
        isupper( '8' ) ? "8 bir büyük harftir" : "8 bir büyük harf degildir",
        isupper( '$' ) ? "$ bir büyük harftir" : "$ bir büyük harf degildir");
    printf( "%s%c\n%s%c\n%s%c\n%s%c\n",
        "u nun büyük harfi: ", toupper( 'u' ),
        "7 nun büyük harfi: ", toupper( '7' ),
        "$ nun büyük harfi: ", toupper( '$' ),
        "L nun küçük harfi: ", tolower( 'L' ) );
    getch();
    return 0;
}

```

```

islower:
p bir küçük harftir
P bir küçük harft degildir
5 bir küçük harf degildir
! bir küçük harf degildir

isupper:
D bir büyük harftir
d bir büyük harf degildir
8 bir büyük harf degildir
$ bir büyük harf degildir

u nun büyük harfi: U
7 nun büyük harfi: 7
$ nun büyük harfi: $
L nun küçük harfi: l

```

```

#include <stdio.h>
#include <ctype.h>
#include <conio.h>
int main()
{
    printf( "%s\n%s%s\n%s%s\n%s\n\n",
        "isspace:",
        "Yeni satir", isspace( '\n' ) ? " bosluk karakteridir" : " bosluk karakteri degildir",
        "Sekme karakteri", isspace( '\t' ) ? " bosluk karakteridir" : " bosluk karakteri degildir",
        isspace( '%' ) ? "% bosluk karakteridir" : "% bosluk karakteri degildir");
    printf( "%s\n%s%s%\n%s%\n\n", "iscntrl:",
        "Yeni satir", iscntrl( '\n' ) ? " kontrol karakteridir" : " kontrol karakteri degildir",
        iscntrl( '$' ) ? "$ kontrol karakteri degildir" : "$ kontrol karakteri degildir");
    printf( "%s\n%s\n%s\n%s\n\n",
        "ispunct:",
        ispunct( ';' ) ? "; noktalama isaretidir" : "; noktalama isareti degildir",
        ispunct( 'Y' ) ? "Y noktalama isaretidir" : "Y noktalama isareti degildir",
        ispunct( '#' ) ? "# noktalama isaretidir" : "# noktalama isareti degildir");
    printf( "%s\n%s\n%s%s\n\n", "isprint:",
        isprint( '$' ) ? "$ yazi karakteridir" : "$ yazi karakteri degildir",
        "Alarm", isprint( '\a' ) ? " yazi karakteridir" : " yazi karakteri degildir");
    printf( "%s\n%s\n%s%s\n", "isgraph:",
        isgraph( 'Q' ) ? "Q bosluktan farkli bir yazi karakteridir" :
        "Q bosluktan farkli bir yazi karakteri degildir",
        "Bosluk", isgraph( ' ' ) ? " bosluktan farkli bir yazi karakteridir" :
        " bosluktan farkli bir yazi karakteri degildir");
    getch();
    return 0;
}

```

```

isspace:
Yeni satir bosluk karakteridir
Sekme karakteri bosluk karakteridir
% bosluk karakteri degildir

```

```

iscntrl:
Yeni satir kontrol karakteridir
$ kontrol karakteri degildir

```

```

ispunct:
; noktalama isaretidir
Y noktalama isareti degildir
# noktalama isaretidir

```

```

isprint:
$ yazi karakteridir
Alarm yazi karakteri degildir

```

```

isgraph:
Q bosluktan farkli bir yazi karakteridir
Bosluk bosluktan farkli bir yazi karakteri degildir

```

String Dönüşüm Fonksiyonları

Genel amaçlı kütüphanedeki (**stdlib**) string dönüşüm fonksiyonlarını göstereceğiz. Bu fonksiyonlar, rakam stringlerini tamsayı ve ondalıklı sayı değerlerine dönüştürür.

Prototip	Fonksiyon Tanımı
double atof (const char *nPtr) ;	nPtr stringini double 'a dönüştürür.
int atoi (const char *nPtr);	nPtr stringini int 'e dönüştürür.
long atol (const char *nPtr);	nPtr stringini long int 'e dönüştürür.
double strtod (const char *nPtr, char ** endPtr);	nPtr stringini double 'a dönüştürür.
long strtol (const char *nPtr, char **endPtr, int base);	nPtr stringini long 'a dönüştürür.
unsigned long strtoul(const char *nPtr, char **endPtr, int base)	nPtr stringini unsigned long 'a dönüştürür.

```
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
int main()
{
    double d;
    d = atof( "99.0" );
    printf( "%s%.3f\n%s%.3f\n",
        "\"99.0\" stringi double tipine donusturuldu ", d,
        "Donusturulen sayi ikiye bolundu: ", d / 2.0 );
    getch();
    return 0;
}
```

```
"99.0" stringi double tipine donusturuldu 99.000
Donusturulen sayi ikiye bolundu: 49.500
```

```
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
int main()
{
    int i;

    i = atoi( "2593" );
    printf( "%s%d\n%s%d\n",
        "\"2593\" stringi int tipine donusturuldu ", i,
        "Donusturulen sayi eksi 593: ", i - 593 );
    getch();
    return 0;
}
```

```
"2593" stringi int tipine donusturuldu 2593
Donusturulen sayi eksi 593: 2000
```

```

#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
int main()
{
    long l;
    l = atol( "1000000" );
    printf( "%s%ld\n%s%ld\n",
        "\"1000000\" stringi long int donusuturuldu ", l,
        "Donusturulen deger 2'ye bolundu: ", l / 2 );
    getch();
    return 0;
}

```

```

"1000000" stringi long int donusuturuldu 1000000
Donusturulen deger 2'ye bolundu: 500000

```

```

#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
int main()
{
    double d;
    const char *string = "51.2% kabul edildi.";
    char *stringPtr;
    d = strtod( string, &stringPtr );
    printf( "String: \"%s\\n\",
        string );
    printf( "double %.2f ve string \"%s\\n 'ye
donusuturuldu.\\n",
        d, stringPtr );
    getch();
    return 0;
}

```

```

String: "51.2% kabul edildi."
double 51.20 ve string "% kabul edildi." 'ye donusuturuldu.

```

```

#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
int main()
{
    long x;
    const char *string = "-1234567abc";
    char *remainderPtr;

    x = strtol( string, &remainderPtr, 0 );
    printf( "%s\\\"%s\\\"\\n%s%ld\\n%s\\\"%s\\\"\\n%s%ld\\n",
        "Orjinal string: ", string,
        "Donusturlen deger ", x,
        "Orjinal string'den geriye kalanlar:",
        remainderPtr,
        "Donusturulen deger arti 567: ", x + 567 );
    getch();
    return 0;
}

```

```

Orjinal string: "-1234567abc"
Donusturlen deger -1234567
Orjinal string'den geriye kalanlar:"abc"
Donusturulen deger arti 567: -1234000

```

```

#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
int main()
{
    unsigned long x;
    const char *string = "1234567abc";
    char *remainderPtr;

    x = strtoul( string, &remainderPtr, 0 );
    printf( "%s\\\"%s\\\"\\n%s%ld\\n%s\\\"%s\\\"\\n%s%ld\\n",
        "Orjinal string: ", string,
        "Donusturlen deger ", x,
        "Orjinal string'den geriye kalanlar:",
        remainderPtr,
        "Donusturulen deger eksi 567: ", x - 567 );
    getch();
    return 0;
}

```

```

Orjinal string: "1234567abc"
Donusturlen deger 1234567
Orjinal string'den geriye kalanlar:"abc"
Donusturulen deger eksi 567: 1234000

```


Standart Giriş/Çıkış Kütüphane Fonksiyonları

Prototip	Fonksiyon Tanımı
int getchar(void);	Standart girişteki karakteri alır ve tamsayı olarak döndürür.
char *gets(char *s);	Standart girişten aldığı karakterleri, yeni satır ya da dosya sonu belirteciyle karşılaşınca s dizisine alır. Sonlandırıcı null karakter diziye eklenir.
int putchar(int c);	c içindeki karakteri yazdırır.
int puts(const char *s);	yeni satır karakteri ile devam edilen bir stringi yazdırır.
int sprintf(char *s,const char *format,...);	printf ile denktir, ancak çıktılar ekran yerine s dizisine gönderilir.
int sscanf (char *s,const char *format,...);	scanf ile denktir, ancak girdiler klavye yerine s dizisinden okunur.

```
#include <stdio.h>
#include <conio.h>
int main()
{
    char sentence[ 80 ];
    printf( "Metin giriniz:\n" );
    gets( sentence );
    printf( "\nGirdiginiz metin\n" );
    puts( sentence );
    getch();
    return 0;
}
```

```
Metin giriniz:
Merhaba

Girdiginiz metin
Merhaba
```

```
#include <stdio.h>
#include <conio.h>
int main()
{
    char c, sentence[ 80 ];
    int i = 0;
    puts( "Bir satir metin giriniz:" );
    while ( ( c = getchar() ) != '\n' )
        sentence[ i++ ] = c;
    sentence[ i ] = '\0';
    puts( "\nGirdiginiz satir:" );
    puts( sentence );
    getch();
    return 0;
}
```

```
Bir satir metin giriniz:
Deneme deneme

Girdiginiz satir:
Deneme deneme
```

```

#include <stdio.h>
#include <conio.h>
int main()
{
    char s[ 80 ];
    int x;
    double y;
    printf( "int ve double tipte deger girin:\n" );
    scanf( "%d%lf", &x, &y );
    sprintf( s, "int:%6d\ndouble:%8.2f", x, y );
    printf( "%s\n%s\n",
        "s dizisinde saklanan cikti:", s );
    getch();
    return 0;
}

```

```

int ve double tipte deger girin:
298 71.1887
s dizisinde saklanan cikti:
int:    298
double:  71.19

```

```

#include <stdio.h>
#include <conio.h>
int main()
{
    char s[] = "31298 87.375";
    int x;
    double y;
    sscanf( s, "%d%lf", &x, &y );
    printf( "%s\n%s%6d\n%s%8.3f\n",
        "s karakter dizisinde saklanan degerler:",
        "integer:", x, "double:", y );
    getch();
    return 0;
}

```

```

s karakter dizisinde saklanan degerler:
integer: 31298
double:  87.375

```

String Kütüphanesindeki String İşleme Fonksiyonları

String kütüphanesi, string verilerini ele almak, stringleri karşılaştırmak, stringlerde karakterler ya da başka stringler aramak, stringleri atomlara (stringi mantıklı parçalara bölmek) ayırmak ve stringlerin uzunluğuna karar vermek gibi bir çok kullanışlı fonksiyon sunar. Bu kısım, string kütüphanesindeki string işleme fonksiyonlarını ele almaktadır. Her fonksiyon (**strncpy** hariç), sonucunun sonuna null karakter ekler.

Not: String kütüphanesindeki fonksiyonları kullanırken, <string.h> öncü dosyasını eklemeyi unutmayın.

Prototip	Fonksiyon Tanımı
char *strcpy (char *s1,const char *s2)	s2 stringini s1 dizisi içine kopyalar,s1'in değeri döndürülür.
char *strncpy(char *s1,const char *s2,size_t n)	s2 stringinin en fazla n karakterini s1 dizisi içine kopyalar.s1'in değeri döndürülür.
char *strcat(char *s1,const char *s2)	s2 stringini s1 dizisine ekler.s2'nin ilk karakteri s1 dizisinin null karakteri üzerine yazılır.s1'in değeri döndürülür.
char *strncat(char *s1,const char *s2,size_t n)	s2 stringinin en fazla n karakterini s1 dizisine ekler.s2'nin ilk karakteri s1 dizisinin null karakteri üzerine yazılır.s1'in değeri döndürülür.

```

#include <stdio.h>
#include <string.h>
#include <conio.h>
int main()
{
    char x[] = "Mutlu yillar
sana";
    char y[ 25 ], z[ 14 ];

    printf( "%s%s\n%s%s\n",
        "x : ", x,
        "y : ", strcpy( y, x ) );

    strncpy( z, x, 13 );
    z[ 13 ] = '\0';
    printf( "z : %s\n", z );
    getch();
    return 0;
}

```

```

x : Mutlu yillar sana
y : Mutlu yillar sana
z : Mutlu yillar

```

```

#include <stdio.h>
#include <string.h>
#include <conio.h>
int main()
{
    char s1[ 20 ] = "Mutlu ";
    char s2[] = "Yeni Yillar ";
    char s3[ 40 ] = "";

    printf( "s1 = %s\ns2 = %s\n", s1, s2 );
    printf( "strcat( s1, s2 ) = %s\n", strcat( s1, s2 ) );
    printf( "strncat( s3, s1, 6 ) = %s\n", strncat( s3, s1, 6 ) );
    printf( "strcat( s3, s1 ) = %s\n", strcat( s3, s1 ) );
    getch();
    return 0;
}

```

```

s1 = Mutlu
s2 = Yeni Yillar
strcat( s1, s2 ) = Mutlu Yeni Yillar
strncat( s3, s1, 6 ) = Mutlu
strcat( s3, s1 ) = Mutlu Mutlu Yeni Yillar

```

SON