**BLGM 112 – DENEY 8[[1]](#footnote-2)\***

# Dizgiler

## Amaçlar

1. Dizgiler ve temel kullanım alanlarının tanıtılması.
2. Dizgiler ile ilgili temel fonksiyonların kullanımları

## Ön bilgi

Dizgiler temelde karakter (*char*) dizileridir, fakat kullanım alanları ve yararları oldukça fazla olduğundan, C standard kütüphanesi ile birlikte birtakım özel dizgi fonksiyonlar gelmektedir. Dizgiler, dizi oldukları için, işaretçilerle de kullanılabilirler. Dizgilerin, karakterler ile karıştırılmaması gereken bir özelliği, tanımında çift tırnak kullanılmasıdır. Karakterler ise tek tırnak ile tanımlanır. Çift tırnak arasında sadece bir karakter konulsa bile, derleyici o karakteri bir dizgi olarak algılayacaktır. Aşağıda, örnek dizgi tanımlamaları ve girdi/çıktı işlemleri verilmiştir.

#include <stdio.h>

#include <string.h>

int main() {

 char \*dizgi1 = “bu bir dizgi”;

 char dizgi2[] = “bu da bir dizgi”;

 printf(“%s\n”, dizgi1);

 printf(“%s\n”, dizgi2);

 return 0;

}

Dizgiler, sıradan dizilerden farklı olarak, en sonda özel bir karakter bulundururlar. Bu karakter '\0' dır. Yukarıdaki örnekte, iki dizginin sonuna derleyici tarafından '\0' koyulmuştur. Bu karakter, çıktı alınırken gösterilmez (örneğin *printf* fonksiyonu ile) fakat hafıza yapısına bakıldığında, her dizgiye, ek olarak '\0' karakteri için bir byte açılır. Bu duruma örnek aşağıda verilmiştir.

printf(“hafızadaki boyut: %d\n”, sizeof(dizgi2)); // sonuç 16

printf(“karakter sayısı: %d\n”, strlen(dizgi2)); // sonuç 15

Tırnak içerisine yazılan karakterlerde küçük kısıtlamalar vardır. Bunlardan ilki iki çift tırnak içinde bir başka çift tırnak kullanımıdır. Bu tarz bir durum derleyicinin dizginin nerede sonlandığını bulmasını engelleyeceği için, özel kelimelerden önce '\' karakteri kullanılır. İkinci durum ise tırnak içinde '\' kullanılmasıdır. Bu durum ise derleyicinin hangi karakterin özel olduğunu anlamasını engeller. O yüzden her kullanılmak istenen '\' için, bir '\' daha kullanılmalıdır. Aşağıda bir takım özel karakter kullanımları verilmiştir.

printf(“%s\n”, “\”bu dizgi cift tirnak icinde\””);

printf(“%s\n”, “3 adet \\:\\\\\\”);

C dilinin standard dizgi fonksiyonları *<string.h>* dosyası içinde bulunur. Bu fonksiyonlardan belkide en önemlileri şunlardır:

* *strcmp(dizgi1, dizgi2)*: iki dizgiyi karşılaştırır, eğer ilk ikinci dizgiden alfabetik olarak büyük ise 1, iki dizgi alfabetik olarak aynı ise 0, ikinci dizgi alfabetik olarak büyük ise -1 döndürür.
* *strcpy(dizgi1, dizgi2)*: ikinci dizgiyi, ilk dizgiye kopyalar.
* *strlen(dizgi)*: verilen dizginin uzunluğunu döndürür, '\0' karakteri sayılmaz.
* *strcat(dizgi1, dizgi2)*: ikinci dizgiyi, ilk dizginin sonuna kopyalar ve iki dizgiyi birleştirir.

## Deneyler

1. Verilen dizgi fonksiyonlarını, farklı parapmetreler ile kullanınız.
2. Kullanıcıdan okunan bir dizgi içerisnde, yine kullanıcıdan okunan bir karakterin kaç kere tekrar ettiğini bulan bir fonksiyon yazınız. Bunun için aşağıdaki fonksiyon prototipini kullanabilirsiniz.

int tekrar(char dizgi[], char aranan, int boyut);

1. Kullanıcıdan okuduğunuz dizginin bir *palindrom* olup olmadığını bulan bir fonksiyon yazınız. Bir kelime hem soldan hem de sağdan aynı şekilde yazılıyorsa palindromdur. Örneğin, “radar”, “para hazır ama rıza harap”. Bu fonksiyon eğer verilen dizgi palindrom ise 1; değislse 0 döndürmeli. Bunun için aşağıdaki fonksiyon prototipini kullanabilirsiniz.

int palindromMu(char \*dizgi);

1. \* BLGM 112 dersi için Bahar 2014/15 döneminde Cengiz Kandemir ve Cem Kalyoncu tarafından hazırlanmıştır. [↑](#footnote-ref-2)