



BLM102

## PROGRAMLAMA DİLLERİ II

Yrd. Doç. Dr. Baha ŞEN

[baha.sen@karabuk.edu.tr](mailto:baha.sen@karabuk.edu.tr)

**KBUZEM**

Karabük Üniversitesi

Uzaktan Eğitim Araştırma ve Uygulama Merkezi

## 1. DAİRESEL LİSTELER

Dairesel tek bağılı listeleri doğrusal listelerden ayıran özellik; son bağı alanında NULL yerine listebaşının adresinin saklanmasıdır. Buradan hareketle çift bağılı dairesele listelerde bu duruma ek olarak ilk düğümün ilk bağı alanında da son düğümün adresi saklanmaktadır.

### 1.1. Tek Bağılı Dairesel Listeler

Bu veri yapısı iki ayrı alandan oluşur. Veri alanı ve bağı alanı. Şekilden de anlaşılacağı üzere listenin son düğümü, listenin ilk düğümünü işaret etmektedir.



Şekil 1: Tek Bağılı Dairesel Listelerin Genel Yapısı

### 1.2. Çift Bağılı Dairesel Listeler

Bu veri yapısının tek bağılı dairesele listeden farkı, önceki düğümün de adresini saklayan bir işaretçi bulunmasıdır. Bu sayede listede iki yönlü ilerleme yeteneğı sağlanmış olur.



Şekil 2: İki bağılı dairesele listenin yapısı

## Örnek:

```
#include <stdlib.h>
#include <string.h>
#include <stdio.h>
#include <conio.h>
struct listyapi
{
    struct listyapi *onceki;
    char    adi[21]; //veri alanı
    struct listyapi *sonraki;
};
typedef struct listyapi listnode; //artık listyapi
yerine listnode kullanılacak
listnode *listebasi; /* Herzaman listenin basini
gosteren bir işaretçi */
void basaekle(char *s)
{
    listnode *yeni_dugum;
    yeni_dugum = (listnode*) malloc(sizeof(listnode));
    /* yeni kayida yer al */
    strcpy(yeni_dugum->adi, s); /*bilgiyi
yeni kayida yaz */
    if(listebasi==NULL)
    {
        listebasi=yeni_dugum;
        yeni_dugum->onceki=yeni_dugum;
        yeni_dugum->sonraki=yeni_dugum;
    }
    else
    {
        listnode *son_node=listebasi->onceki;
        yeni_dugum->sonraki = listebasi;
        yeni_dugum->onceki  = son_node;
        listebasi->onceki   = yeni_dugum;
        son_node->sonraki   = yeni_dugum;
        listebasi=yeni_dugum;
    }
}
void sonaekle(char *s)
{
    listnode *yeni_dugum;
    yeni_dugum = (listnode*) malloc(sizeof(listnode));
    /* yeni kayida yer al */
```

```

    strcpy(yeni_dugum->adi, s);           /*bilgiyi
yeni kayida yaz */
    if(listebasi==NULL)
    {
        listebasi=yeni_dugum;
        yeni_dugum->onceki=yeni_dugum;
        yeni_dugum->sonraki=yeni_dugum;
    }
    else
    {
        listnode *son_node=listebasi->onceki;
        yeni_dugum->sonraki = listebasi;
        yeni_dugum->onceki  = son_node;
        listebasi->onceki   = yeni_dugum;
        son_node->sonraki   = yeni_dugum;
    }
}
void listlist(void)
{
    listnode *aktif_node = listebasi;
    while (aktif_node != NULL)
    {
        printf("%s ",aktif_node->adi);
        aktif_node = aktif_node->sonraki;
        if(aktif_node->onceki == listebasi->onceki)
break;
    }
    printf("\n");
}
int main()
{
    char      sec;
    char      *s;
    do
    {
        system("cls");
        listlist();
        printf("\n\n1 - Basa Ekle\n2 - Sona Ekle\n3 -
Son\n\nSec :");
        sec = getche();
        switch (sec)
        {
            case '1':
                printf("\nAdi :");

```

```
        gets(s);
        basaekle(s);
        break;
    case '2':
        printf("\nAdi :");
        gets(s);
        sonaekle(s);
        break;
    case '3':
        return(0);
        break;
    }
}
while (1);
}
```