**VERİ YAPILARI UYGULAMA ÖDEVİ**

**YUNUS EMRE TOPÇU**

**1170505060**

**YAZILIM MÜHENDİSLİĞİ 1. SINIF**

**SON TESLİM TARİHİ: 21.03.2018 23:59**

**VERİ YAPILARI VE ALGORİTMALAR**

**DR. ÖĞR. Ü. EDİP SERDAR GÜNER**

**ARAŞ. GÖR. BURAK BEYNEK**

**İÇİNDEKİLER**

**Sf.**

Ödev kapağı 1

İçindekiler 2

Giriş 3

Bağlı Liste Nedir? 4-6

Bağlı Liste Ne Amaçla Kullanılır? 7

C Dilinde Bağlı Liste Nasıl Oluşturulur? 8

C Dilinde Bağlı Listeye Yeni Eleman Ekleme 9-10

Sonuç 11

Bibliyografya 12

Bu ödevde bağlı listenin ne olduğunu, nerelerde hangi amaçla kullanıldığını öğreneceğiz. C programlama dilinde bağlı listeyi kullanmayı öğreneceğiz.

1. **Bağlı Liste Nedir?**

Bağlı liste, herhangi bir tipten düğümlerin yine aynı tiplerinden düğümlere işaret etmesi(*pointer*) ile oluşan zincire verilen isimdir. Aşağıda örnek bir bağlı liste yapısını görebiliriz:

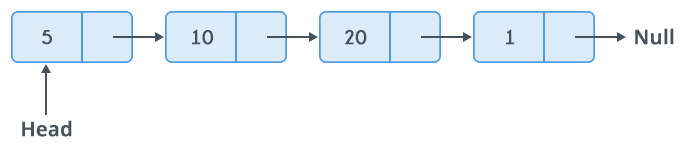


Bağlı listede hafızayı dinamik olarak kullanırız. Bu da bize istediğimiz kadar veri saklayabilme olanağı sağlar. Buna göre hafızada silinen veri o alanı boşaltır veya yeni eklenen veri için sadece o bilgiyi tutacak kadar yer ayrılacaktır.

Bağlı listeler, tek yönlü liste, çift yönlü liste ve dairesel listeler olmak üzere üçe ayrılır.

* + 1. **Tek Yönlü Bağlı Liste (*Single Linked List*)**

Bu listede her düğüm kendinden bir sonraki düğümü işaret eder. Bu listeleme türünde en son düğüm boş(*NULL*) değer gösterir. En baştaki düğüm baş(*head*) düğümüdür ve işlemler bu düğümden başlayarak yapılır. Bu listede sadece ileri yönlü hareket söz konusudur.



Şekil 1.1

* + 1. **Çift Yönlü Bağlı Liste (*Doubly Linked List*)**

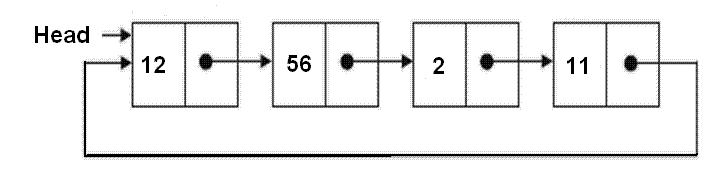
Bu listede her düğüm tek yönlü bağlı listedekilerden farklı olarak kendinden bir önceki ve bir sonraki düğümü gösterir. İlk düğümün öncesi boş(*null*), son düğümün sonrası boş(*null*) olur. Bu listede ileri veya geri yönlü hareket mümkündür.



Şekil 1.2

* + 1. **Tek Yönlü Dairesel Bağlı Liste (*Singly Circular Linked List*)**

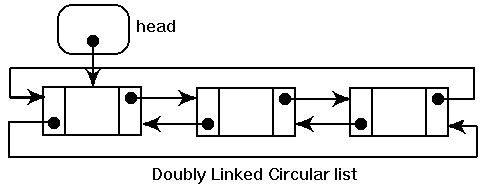
Bu listede her düğüm kendiliğinden bir sonraki düğümü gösterir. Bunun yanısıra son düğüm ilk düğümü ve ilk düğüm de son düğümü gösterir. Bunun nedeni listenin sonundan başına gitmeyi kolaylaştırmaktır.



Şekil 1.3

* + 1. **Çift Yönlü Dairesel Bağlı Liste (*Doubly Circular Linked List*)**

Bu listede her düğüm kendinden bir önceki ve bir sonraki düğümü gösterdiği gibi bu ilk düğüm son düğümü, son düğüm de ilk düğümü gösterir. İlk düğümde ileri geri hareket kabiliyetini sağlar.



Şekil 1.4

* 1. **Bağlı Liste ve Dizi(*Array*) Karşılaştırılması**

Bağlı liste ve dizilerin en temel farkı bağlı listede eleman ekleme ve çıkarma işleminin daha basit olmasıdır. Şöyle düşünelim, elimizde 1000 elemanlık bir dizi var ve biz 500. Elemanı silmek istiyoruz. Bu durumda 499 elemanı birer kaydırmalıyız. Bu da 499 tane işlem gerçekleştirmek demektir. Eğer milyonlarca veri ile uğraşıyorsak bu içinden çıkılamayacak bir hal almış olur. Bağlı listede böyle bir şeye gerek yoktur. İstediğimiz elemanı kolayca silip ekleyebiliriz. Bağlı listenin bir diğer avantajı da dinamik olmasıdır. Modern dillerde bu geçerli olmasa da C, C++ gibi dillerde dizi tanımladığımızda boyutunu da belirleriz. Bu gibi dillerde boyut aşımı gerçekleşmez. (Ram miktarını sınırsız düşünürsek.) Bağlı listede böyle bir şey söz konusu değildir.

* 1. **Bağlı Listelerde Yapılan İşlemler**

1. Başa, araya, sona eleman ekleme
2. Baştan, aradan, sondan eleman silme
3. Ters çevirme
   1. **Bağlı Listenin Dezavantajları**

Bağlı listelerin dizilere karşı birtakım dezavantajları bulunur. Bağlı listelerde mevcut bilgilere ek olarak bir de adres bilgisi tutulmaktadır. Bu da daha fazla alan kullanımı anlamına gelmektedir. Diğer bir dezavantaj ise rastgele erişimin maliyetli olmasıdır. Rastgele(*random*) erişim ile kast edilen şey elimizdeki herhangi bir veriye erişmek için geçen zaman gibi düşünebilirsiniz. Bir dizide 70. İndise erişmek istediğimiz zaman direkt olarak erişiriz. Ancak bağlı listede 70. İndise erişmek için aradaki tüm düğümleri dolaşmamız gerekmektedir. Bu da maliyeti artırmaktadır. Bu tip sorunları çözmek için geliştirilmiş pek çok yöntem bulunmaktadır. Ancak en yalın halleriyle incelediğimizde bağlı liste yapısında rastgele erişim zordur.

**2.0) Bağlı Liste Ne Amaçla Kullanılır?**

Bağlı liste elimizdeki çok sayıda veriyi kullanmak için kullanılır. Gerektiğinde her elemana erişilmesi ve veri ekleme ve silme işlemlerinin kolayca yapılabilmesi büyük bir avantajdır. Ayrıca elimizde belirli bir bellek alanı bulunmaz, alan daraltılabilir ya da genişletilebilir. Şu şekilde açıklayalım, 50 elemanlık bir dizi tanımlayalım. Kullanıcı buraya 51. Elemanı girmek isterse sıkıntılar ortaya çıkacaktır. 50’den az eleman girdiğinde ise geri kalan alan bellekte boşa yer kaplayacaktır. Bellekte boşa yer kaplamayı önlemek amacıyla bağlı liste kullanılır

**3.0) C Dilinde Bağlı Liste Nasıl Oluşturulur?**

C programlama dilinde bağlı liste oluştururken şu adımları takip ederiz:

1. Önce kayıtların tutulacağı struct veri yapısını tanımlarız.
2. Sonra bu struct yapısını baz alarak amacımıza uygun bir şekilde fonksiyon yazmaya başlarız.
3. Fonksiyonları yazdıktan sonra ana(*main*) fonksiyonu içinde amacımıza uygun olarak kullanırız.

#include<stdio.h>

#include<stdlib.h>

struct dugum{

int anahtar;

int veri;

struct dugum \*ileri;};

struct dugum \*bas = NULL; // Baş değere boş(*NULL*) atadık

// Şimdi birkaç fonksiyon yazalım

void insertFirst(int k, int d){ // k, anahtar için, d, veri için

struct dugum \*temp = malloc(sizeof(struct dugum));

temp->anahtar = k;

temp->veri = d;

temp->ileri = bas;

bas = temp;}

void printList(){

struct dugum \*ptr = bas;

while(!(ptr == NULL)){

printf("%d, %d\n",ptr->anahtar,ptr->veri);

ptr = ptr->ileri;

}}

// Fonksiyonlarımızı yazdık şimdi de ana fonksiyonumuz olan main'i yazalım

main(){ insertFirst(1,12);

insertFirst(2,23);

insertFirst(3,34);

insertFirst(4,45);

printList();

return 0;}

Metin Kutusu 1.0

C dilinde bağlı liste kullanımını daha iyi anlamak için aşağıdaki kodu inceleyelim:

**4.0) C Dilinde Bağlı Liste’ye Yeni Eleman Nasıl Eklenir?**

C dilinde bağlı listeye yeni eleman eklemeyi bir C kodu üzerinde anlatalım:

#include<stdio.h>

#include<stdlib.h>

struct bagliliste{

int anahtar;

int veri;

int \*ileri;}

struct bagliliste \*bas = NULL;

void ekle(int a, int v){

struct dugum \*geçici = malloc(sizeof(bagliliste));

gecici->anahtar = a;

gecici->veri = v;

gecici->ileri = bas;

bas = gecici;}

main(){

ekle(2,5);

return 0;

}

Metin Kutusu 2.0

Şimdi bu kodu açıklayalım:

Öncelikle ‘bagliliste’ adında bir struct tanımladık ve bu structın içinde ‘anahtar’, ‘veri’ değişkenlerini ve ‘ileri’ işaretçisini tanımladık. Daha sonra bu bağlı listemizin ‘bas’ kısmına NULL değer atadık.

Bu işlemlerden sonra ‘ekle’ fonksiyonunu açtık ve anahtar için ‘a’, veri için ‘v’ değişkenlerini oluşturduk. Daha sonra malloc ile yer ayırma işlemini gerçekleştirdik ve gecici üzerinden anahtara a değerini atadık. Benzer işlemleri aşağıda sıraladık ve en son bas değişkenine gecici değişkenini atadık. Ana fonksiyonumuzun içine oluşturduğumuz fonksiyonu yazdık ve parantez içine eklemek istediğimiz değerleri yazdık.

C programlama dilinde bağlı liste ile ekleme yapma fonksiyonu bu şekilde yapabiliriz

**Sonuç olarak;**

Bağlı liste, dizi yerine kullanılabilecek daha kullanışlı bir yapıdır. Rahatça ekleme çıkarma yapabilmemiz, istediğimiz elemana erişebilmemiz ve dinamik bir bellek alanı sağlaması bizlere büyük avantajlar ve kolaylıklar sağlar. Bağlı listenin ana mantığı bir struct içerisinde değişkenlerimizi belirleriz ve daha sonra bu değişkenlerin yerini tutacak bir işaretçi(*pointer*) belirleriz daha sonra gerekli şekilde işlemlerimizi gerçekleştirerek verilerimizi belleğe yazmış oluruz. Bağlı listelerin amacı bizleri ekstra maliyetten kurtarmak ve bellekten tasarruf etmemizi sağlayıp işlerimizi kolaylaştırmaktır. C dilinde bir değişken ve bir ileri adında pointer ile tek yönlü bir bağlı liste oluşturabiliriz. ‘malloc’ dediğimiz yapı ile bellekte alan ayırabiliriz ve alanımız doldukça bu alanı genişletebiliriz.

**Bibliyografya**

<http://bilgisayarkavramlari.sadievrenseker.com/2007/05/03/linked-list-linkli-liste-veya-bagli-liste/>

<http://www.algoritmauzmani.com/veri-yapilari/bagli-liste-nedir-linked-list-nedir/>

<https://mericberber.com/linked-list-bagli-liste-nedir/>

<https://omkarnathsingh.files.wordpress.com/2015/07/dcll.gif>

<http://2.bp.blogspot.com/-I3Kad-rBO2M/VpJ6fTFpv6I/AAAAAAAACV4/_oeMufKeIYc/s1600/cll_orig.gif>

<https://www.geeksforgeeks.org/wp-content/uploads/gq/2014/03/DLL1.png>

<https://he-s3.s3.amazonaws.com/media/uploads/1b76d10.png>

<http://turgayozgur.com/cde-bagli-liste-ile-kayit-fonksiyonlari.html>