

YILDIZ TEKNİK ÜNİVERSİTESİ

ELEKTRİK-ELEKTRONİK FAKÜLTESİ

BİLGİSAYAR MÜHENDİSLİĞİ

2014-2015 ÖĞRETİM YILI GÜZ YARIYILI

VERİ YAPILARI VE ALGORİTMALAR

ÖDEV-3

( BLM-2512/ GRUP:1)

**Hazırlanan Anabilim Dalı**

**Bilgisayar Bilimleri Anabilim Dalı**

**Hazırlayan**

Mert Sevil

09013057

Bilgisayar Mühendisliği Lisans Programı

**Öğretim Üyesi**

Prof. M. Yahya KARSLIGİL

**İSTANBUL, 2014**

**İçindekiler**

1. Ödevin amacı, tanıtımı ve giriş………………………………………………………
2. Linkli listeler…………………………………………………………………………...
   1. Linkli listelerin oluşturulması………………………………………………..
   2. Linkli listelerde eleman ekleme…...………………………………………….
   3. Linkli listelerde arama………..………………………………………………
   4. Linkli listelerde silme……...……………………………………………….
   5. Dairesel linkli listeler…...……………………………………………………...
   6. Çift bağlı linkli listeler……..……………………………………………...
   7. Linkli listelerin avantaj ve dezavantajları……………………………………..
   8. Linkli listelerin kullanıldıkları yerler…………………………………………..
3. Ödevin gerçekleştirilmesi…………………………………………………………….
   1. Kod için algoritmanın çizilmesi…………………………………....
   2. Kodun yazılması……………………………………………..
   3. Çıktıların elde edilmesi ve algoritma analizi…………………………..
   4. Sonuçların yorumlanması…………………………………………………….
4. Kaynakça……………………………………………………………………………...

**1. Ödevin amacı, tanıtımı ve giriş**

**Ödevin amacı:**

* Ödev önemli bir veri yapısı sayılan linkli listeleri anlamayı ve özümsemeyi hedeflemektedir.
* Öğrenilen bilgiler C programlama dilinde kodlanacağı için çizilen bir akış diyagramının C dilinde kodlayabilme beceresi geliştirilir.
* Önemli bir C programlama dili konusu olan pointerlar tekrar edilecek ve uygulanacaktır.
* Linkli listelerin avantajları, dezavantajları ile kullanıldığı örnek uygulamalar incelenecektir.
* Algoritma çizebilme, çizilen algoritmanın analizini yapabilme, çıktıları yorumlayabilme gibi bilgisayar mühendisliğine ait önemli konular ödev sayesinden tekrar edilebilecektir
* Dairesel linkli liste, çift yönlü linkli liste gibi linkli listeye ait özel konular ele alınarak değerlendirilecektir
* Linkli listede temel işlemler kodlanarak gözlemlenecek ve analiz edilecektir
* Dönem boyunca öğrenilecek pekçok veri yapısına temel teşkil edilecektir

**Tanıtım ve Giriş:**

Linkli listeler veri yapıları arasında büyük öneme sahiptir. Bağlı liste herhangi bir tipten node’ların (düğümlerin) yine kendi tiplerinden düğümlere işaret etmesi (point) ile oluşan zincire verilen isimdir. Buna göre her düğümde kendi tipinden bir pointer olacak ve bu pointerlar ile düğümler birbirine aşağıdaki şekilde bağlanacaktır. [1]

[bagli liste](http://bilgisayarkavramlari.com/wp-content/uploads/2007/05/singly_linked_list.png)

Şekil- Linkli listelerin şematize edilmesi

Linked List’in avantajı, hafızayı dinamik olarak kullanmasıdır. Buna göre hafızadan silinen bir bilgi için hafıza alanı boşaltılacak veya yeni eklenen bir bilgi için sadece o bilgiyi tutmaya yetecek kadar hafıza alanı ayrılacaktır. [1]

Yukarıdaki figürde görülen bağlı listeye çok benzeyen ve yine çok kullanılan bir bağlı liste uygulaması da çift bağlı liste (doubly linked list) uygulamasıdır. [1]

[Çift bağlı liste](http://bilgisayarkavramlari.com/wp-content/uploads/2007/05/doublylinkedlist.png)

Şekil- Çift yönlü linkli listeler

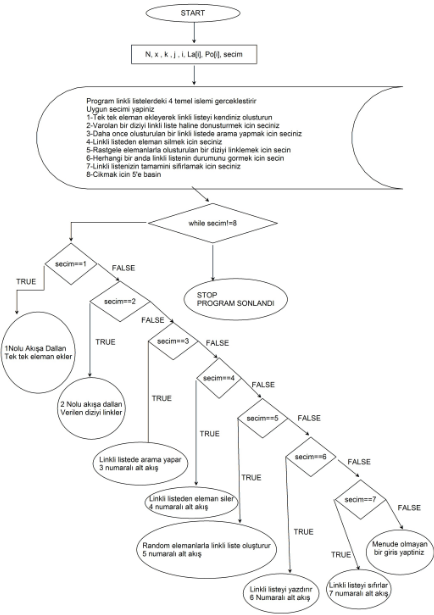
Buna göre her düğüm, hem kendinden öncekine hem de kendinden sonrakine bağlanır, bu sayede liste üzerinde ileri ve geri ilerlemek mümkündür. [1]

**3. Ödevin Gerçekleştirilmesi**

Ödev C programlama dilinde Dev C++ derleyicisinde kodlanmıştır. Ancak ödevin gerçekleştirilmesinde öncelikli olarak algoritmanın çizilmesi, akabinde çizilen bu algoritmanın analizi ve kodlanması sırayla birbirini takip etmiştir. Bu anlamda 3.1 bölümünde algoritmik akış açıkça çizilmiştir.

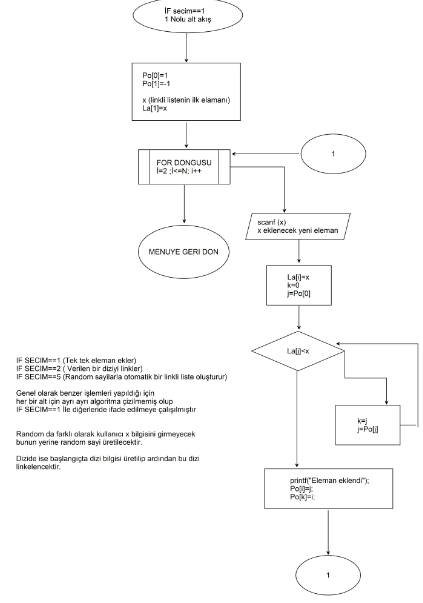
**3.1. Algoritmanın Çizilmesi**

Algoritmik akış aşağıda açıkça ifade edilmiştir. Buna göre başta kullanıcıyı bir Menü (kullanıcı arayüzü) karşılar. Bu arayüze göre seçim yapan kullanıcının seçimine göre linkli liste üzerinde temel işlemler gerçekleştirilir. Ekleme, silme, arama ve oluşturma bu işlemlerden temel olanlarıdır. Şekil-‘de bu durum gözlemlenmektedir.



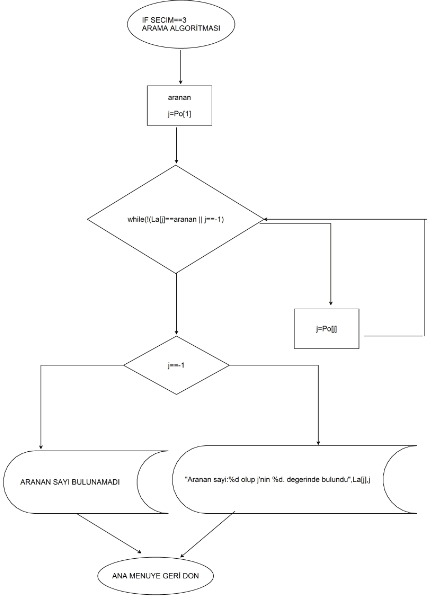
Şekil- Program menüsü ve alt ana akışların kabaca gösterilmesi

Şekil- deki algoritmada if secim==1 olması durumu ele alınmıştır. Bu algoritmik akış diziye tek tek eleman ekleyerek dizinin oluşturulmasına ve diğer bir mantıkla bakıldığında ise diziye eleman eklenmesi işlemlerine denk düşer.



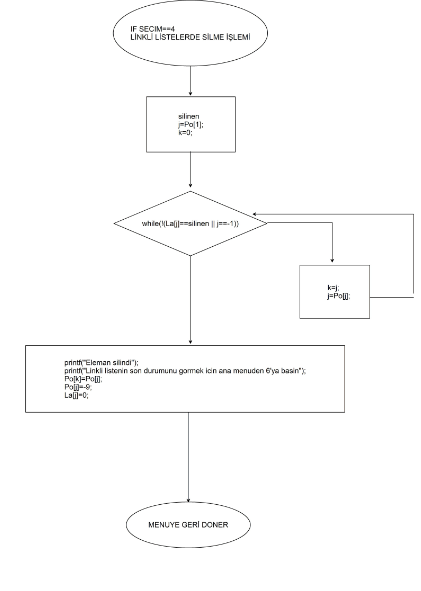
Şekil- IF SECIM==1’e ait alt ana akış (Linkli listeye eleman ekleye ekleye oluşturma)

Linkli listelerde diğer önemli bir konu arama işlemedir. Arama işlemi bütün veri yapıları için önem arzeder. Bir arama algoritmasının o veri yapısındaki karmaşıklığı o veri yapısının gücünü ve kullanım yerini belirler. Bu açıdan linkli listelerde arama işlemi binary ve dizi olarak yapılabilmektedir. Uygulamamızda linkli liste üzerinde bir dizi üzerindeki arama işlemi gibi bir yapıya denk düşen arama algoritması gerçekleştirilmiştir. Bu algoritmaya ait alt akış Şekil- de belirtilmiştir.



Şekil- IF SECIM==3 ‘e ait alt ana akış (Linkli listede arama algoritması)

Linkli listelerde ekleme, oluşturma ve arama kadar önemli bir diğer temel işlem ise silme algoritmalarıdır. Listeler üzerinde ekleme çıkarma işlemleri veri yapıları üzerinde gerçekleştirilen ve sıkça ihtiyaç duyulan işlemlerdir. Bu açıdan uygulamamızda linkli liste üzerinde silme işlemi gerçekleştirilmiştir. Şekil- de bu algoritmik akış gözlenmektedir.



Şekil- IF SECIM==4 ‘e ait alt ana akış (Linkli listede silme algoritması)

Böylece linkli listeler için;

* Oluşturma
* Ekleme
* Silme
* Arama

Algoritmalarına ait kodlanması planlanan algoritmik tasarımlar şematize edilmiştir. Bundan sonraki adım bu algoritmaları C programlama dilinde Dev C++ derleyicisinde kodlamak olmuştur.

**3.2. Kodun yazılması**

Kod aşağıdaki gibi gerçekleştirilmiş ve derlenmiştir. Derleme sonucunda herhangi bir hata ile karşılaşılmamıştır. Kod çalıştırılıp denenmiş ve algoritma analizi yapılmıştır. Analiz sonuçları 3.3 bölümünde değerlendirilmiştir.

#include <stdio.h>

#include <conio.h>

#include <stdlib.h>

#include <time.h>

int main(){

int secim;

int N,x,k,j,i;

printf("Program boyunca kullanilacak dizi veya linkli liste icin eleman sayisini giriniz\n");

scanf("%d",&N);

int\* La;

La= (int \*)malloc (N\*sizeof(int));

int\* Po;

Po= (int \*)malloc (N\*sizeof(int));

while(secim!=8){

printf("Program linkli listelerdeki 4 temel islemi gerceklestirir\n");

printf("Uygun secimi yapiniz\n");

printf("1-Tek tek eleman ekleyerek linkli listeyi kendiniz olusturun\n");

printf("2-Varolan bir diziyi linkli liste haline donusturmek icin seciniz\n");

printf("3-Daha once olusturulan bir linkli listede arama yapmak icin seciniz\n");

printf("4-Linkli listeden eleman silmek icin seciniz\n");

printf("5-Rastgele elemanlarla olusturulan bir diziyi linklemek icin secin\n");

printf("6-Herhangi bir anda linkli listenin durumunu gormek icin secin\n");

printf("7-Linkli listenizin tamamini sifirlamak icin seciniz\n");

printf("8-Cikmak icin 5'e basin\n");

scanf("%d",&secim);

printf("Seciminiz:%d dir\n",secim);

if(secim==1){ //Tek tek eleman eklemek suretiyle linkli listeyi oluşturur

/\*int N,x,k,j,i;

printf("Eleman sayisini giriniz\n");

scanf("%d",&N);

int\* La;

La= (int \*)malloc (N\*sizeof(int));

int\* Po;

Po= (int \*)malloc (N\*sizeof(int));\*/

Po[0]=1;

Po[1]=-1;

printf("Linkli listelinin ilk elemanini girin\n");

scanf("%d",&x);

La[1]=x;

for(i=2;i<=N;i++){

printf("Linkli listelinin %d. elemanini girin\n",i);

scanf("%d",&x);

La[i]=x;

k=0;

j=Po[0];

while(La[j]<x){

k=j;

j=Po[j];

}

printf("Eleman eklendi\n");

Po[i]=j;

Po[k]=i;

}

// printf("Linkli listelinin %d. elemanini girin\n",i);

//scanf("%d",&x);

// La[i]=x;

// Po[N]=-9;

for(i=1;i<=N;i++)

printf("Indis:%d La:%d Po:%d\n",i,La[i],Po[i]);

}

else if(secim==2){ /// Verilen bir diziyi linkler

/\* int N,x,k,j,i;

printf("Eleman sayisini giriniz\n");

scanf("%d",&N);

int\* La;

La= (int \*)malloc (N\*sizeof(int));

int\* Po;

Po= (int \*)malloc (N\*sizeof(int));\*/

int\* temp;

temp= (int \*)malloc (N\*sizeof(int));

Po[0]=1;

Po[1]=-1;

printf("Dizinizin elemanlarini eleman sayisi kadar giriniz\n");

for(i=1;i<N;i++){

scanf("%d ",&temp[i]);

}

La[1]=temp[1];

for(i=2;i<=N;i++){

La[i]=temp[i];

k=0;

j=Po[0];

while(La[j]<temp[i]){

k=j;

j=Po[j];

}

Po[i]=j;

Po[k]=i;

}

// printf("Linkli listelinin %d. elemanini girin\n",i);

//scanf("%d",&x);

// La[i]=x;

// Po[N]=-9;

printf("Diziniz linklendi,linkli liste sonuculari su sekildedir\n");

for(i=1;i<=N;i++)

printf("Indis:%d La:%d Po:%d\n",i,La[i],Po[i]);

}

else if(secim==3){ //Linkli listede arama yapar

int aranan;

printf("Aranacak elemani giriniz\n");

scanf("%d",&aranan);

j=Po[1];

while(!(La[j]==aranan || j==-1)){

j=Po[j];

}

if(j==-1)

printf("Aranan bulunamadi\n");

else

printf("Aranan sayi:%d olup j'nin %d. degerinde bulundu",La[j],j);

}

else if(secim==4){ // Linkli listelerde silme işlemi yapar

int silinen;

printf("Silinecek elemanin linkli listede varoldugundan eminseniz sileceginiz elemani giriniz\n");

scanf("%d",&silinen);

j=Po[1];

k=0;

while(!(La[j]==silinen || j==-1)){

k=j;

j=Po[j];

}

printf("Eleman silindi\n");

printf("Linkli listenin son durumunu gormek icin ana menuden 6'ya basin\n");

Po[k]=Po[j];

Po[j]=-9;

La[j]=0;

}

else if(secim==5){ //Random sayılarla linkli liste oluştururu (Deneme kolaylığı için yapıldı)

int\* temp;

temp= (int \*)malloc (N\*sizeof(int));

Po[0]=1;

Po[1]=-1;

printf("Dizinizin elemanlarini eleman sayisi kadar giriniz\n");

for(i=1;i<N;i++){

temp[i]=rand()%250;

}

La[1]=temp[1];

for(i=2;i<=N;i++){

La[i]=temp[i];

k=0;

j=Po[0];

while(La[j]<temp[i]){

k=j;

j=Po[j];

}

Po[i]=j;

Po[k]=i;

}

// printf("Linkli listelinin %d. elemanini girin\n",i);

//scanf("%d",&x);

// La[i]=x;

// Po[N]=-9;

printf("Diziniz linklendi,linkli liste sonuculari su sekildedir\n");

for(i=1;i<=N;i++)

printf("Indis:%d La:%d Po:%d\n",i,La[i],Po[i]);

}

else if(secim==6){ //Linkli listeyi yazdırır

for(i=1;i<=N;i++)

printf("Indis:%d La:%d Po:%d\n",i,La[i],Po[i]);

}

else if(secim==7){ //Linkli listeyi sıfırlar

for(i=1;i<=N;i++){

Po[i]=0;

La[i]=0;

}

printf("Diziniz sifirlandi\n");

}

else{

printf("Yanlis bir secim yaptiniz yeniden menuden secim yapiniz\n");

}

}

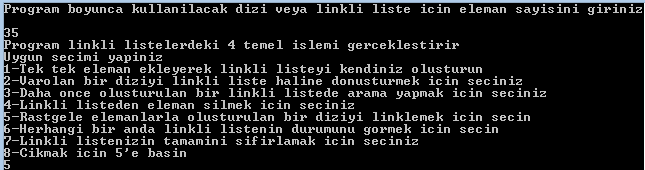
getch();

return 0;

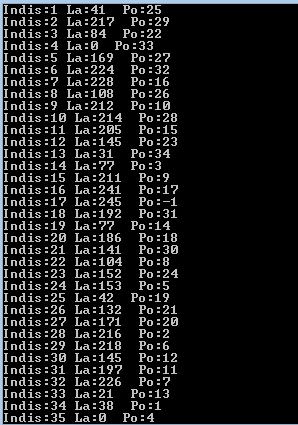
}

**3.3 Çıktıların elde edilmesi ve algoritma analizi**

Deneme kolaylığı olarak analizlerde random sayı olarak atanan dizilerin linklenmesi metodu kullanılmıştır. Bu açıdan ilk analizimiz 35 sayılık random olarak üretilen sayıların linkli liste metoduyla tutulması ve gözlemlenmesi şeklinde olmuştur. Bu açıdan Şekil- de görülen kullanıcı 5 seçimine basar. Böylece Şekil- de görülen linkli liste oluşur.

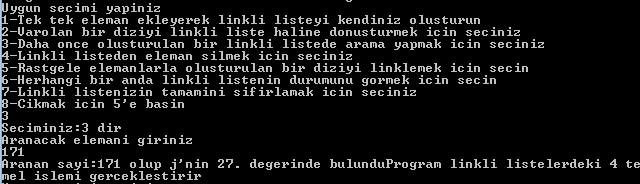


Şekil- Kullanıcı arayüzü ve seçimi (Programa ait menü)



Şekil- Random olarak üretilen 35 sayılık bir linkli liste yapısı

Yukarıda üretilen linkli liste üzerinde arama yapılmak istenilirse menü üzerinden 3’e basılmalıdır. Örnek olarak bu linkli listede 171 sayısı aranırsa buna ait ekran çıktısı Şekil- de gösterilmiştir. Ve yine örnek olarak listede bulunmayan 32 sayısı aranırsa bu duruma ekran çıktısı Şekil- de belirtilmiştir.

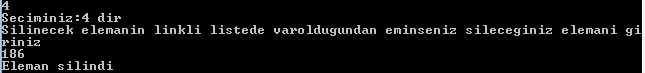


Şekil- 171 sayısının linkli listede aratılması ve bu duruma ait ekran çıktısı



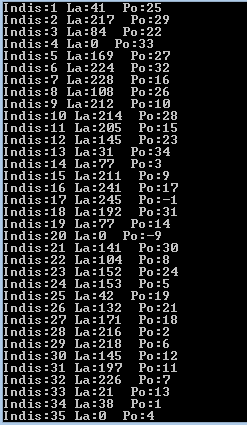
Şekil- 32 sayısının linkli listede aratılması ve bu duruma ait ekran çıktısı

Bu durumdayken linkli listeden eleman silinmek istenilebilir. Bu duruma örnek olarak 20. İndisteki 186 sayısı silinmek istenirse bu duruma denk düşecek ekran çıktısı Şekil- de belirtilmiştir.



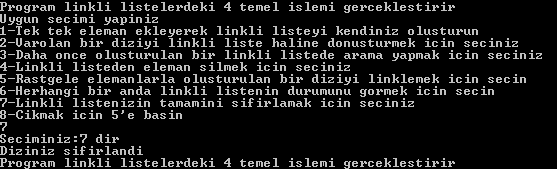
Şekil- 186 elemanının linkli listeden silinmesi durumu ve buna ait ekran çıktısı

Elemanın gerçekten silinip silinmediğini gözlemlemek için tekrar linkli liste yazdırılabilir. Böylece yazdırma işlemi de test edilmiş olacaktır. Bu duruma denk düşecek ekran çıktısı Şekil- de belirtilmiştir.

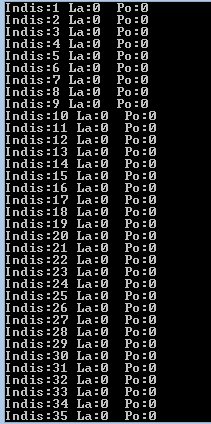


Şekil- 20. İndiste tutulan 186 verisinin silinmesi durumu ve buna ait ekran çıktısı

Görüldüğü gibi 20. İndisteki 186 verisi silinmiştir. Silinmeden önce 27. İndisteki verinin pointeri 20. İşaret ederken şimdi 18. İndisteki veriyi göstermektedir. 20. İndisteki göstericinin değeri ise silindi anlamına gelen -9 değerine eşitlenmiştir. Son olarak ekleme işlemini incelemek için önce linkli listemizi tamamen 0’layım bu durum Şekil- de gösterilmiştir.

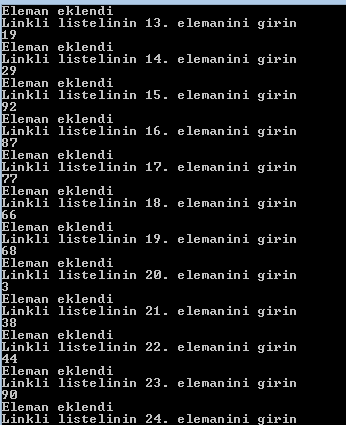


Şekil- Linkli listenin tüm verilerinin sıfırlanması durumu

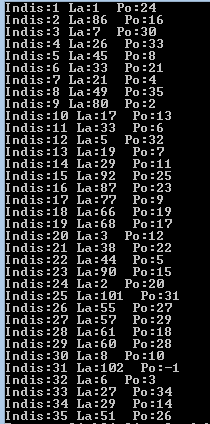


Şekil- Linkli listenin tüm verilerinin sıfırlanması durumu

Bu aşamada linkli listemizi tek tek eleman ekleyerek kendimiz oluşturalım böylece ekleme algoritmasının doğruluğu da tartışılacaktır. Örnek olarak 13-24 arası elemanların eklenmesi işlemi gösterilmiş olup, 1-35 arasındaki bütün verileri gösteren linkli liste dizisi Şekil- de gösterilmiştir.



Şekil- Tek tek elemanların eklenmesiyle elde edilen linkli liste



Şekil- Tek tek elemanların eklenmesiyle elde edilen linkli liste

Böylece rapora konulan bu analizlerin dışında gerçekleştirilen pekçok analiz ile kodumuza ait bütün fonksiyonlar test edilmiştir. Kodun doğru çalıştığı gözlemlenmiştir.

**3.4 Sonuçların yorumlanması**

Sonuçta ödevimizde önemli bir veri yapısı olan linkli listeler üzerinde 4 temel işlemi yerine getirebilecek algoritmik akış tasarlanmış, çizilmiş, analiz edilmiş ve bir programlama dilinde gerçekleştirilmiştir. Ödev önümüzdeki konular için önemli bir köşe taşı olmuş olup, aynı zamanda algoritma tasarlayabilme yeteneğinin geliştirilmesi ve programlama dillerine hâkimiyet gibi açılardan oldukça büyük katkı sağlamıştır.

**4. Kaynakça**

[1] <http://bilgisayarkavramlari.sadievrenseker.com/2007/05/03/linked-list-linkli-liste-veya-bagli-liste/> (İnternet Kaynağı, Erişim Tarihi: 30.09.2014)