# ROGRAMLAMA LABORATUVARI I PROJE 3:

# BAĞLI LİSTE İLE KELİME SAYMA

**SENA ÖKTEM**

Kocaeli Üniversitesi

Mühendislik Fakültesi

Bilgisayar Mühendisliği(İÖ)

190202054@kocaeli.edu.tr

**RUMEYSA ÜSTÜN**

Kocaeli Üniversitesi

Mühendislik Fakültesi

Bilgisayar Mühendisliği(İÖ)

190202011@kocaeli.edu.tr

## PROBLEM TANIMI

Projede bir .txt dosyasında verilen metin içerisinde kelimenin okunması ve okunan bu kelimenin txt dosyasında kaç kere geçtiğinin bulunması istenmiştir. Eğer okunan kelime linked listte yoksa, okunan kelimenin ve frekansının linked liste atılması istenmiştir. (Bu işlemler her kelime için tek tek yapılmaktadır. Tüm kelimeler ve frekanslar hesaplandıktan sonra linked liste atılma istenmemiştir.) Tüm kelimeler linked liste atıldıktan sonra frekanslarına(metinde geçme sıklıklıklarına) göre linked list büyükten küçüğe doğru sıralanmıştır.

## YAPILAN ARAŞTIRMALAR

.txt dosyasından kelimeleri tek tek okumak için dosya işlemleri konusunda araştırma yapıldı. getc() fonksiyonu ile metin içerisinden karakter karakter okuma yapılması sağlandı.*i[1]

.txt dosyasından okunan kelimenin linked listte olup olmadığını kontrol etmek için int VarMi() fonksiyonu geliştirildi. Bu fonksiyonun geliştirilmesi aşamasında linked list içinde nasıl arama yapılabileceği konusunda araştırmalar yapıldı. Veri yapıları ve Algoritmalar ders slaytından ve videosundan faydalanılarak ders içinde yazılan kodlar incelendi. *i[2]

Linked liste ekleme yapmak için BasaEkle(), ArayaEkle() ve SonaEkle() fonksiyonları Veri Yapıları ve Algoritmalar ders slaytından inceleme yapılarak yazıldı. *i[2]

Linked listte büyükten küçüğe sıralama yapmak için void SiraliEkle() fonksiyonu geliştirildi. Bu fonksiyon içinde yukarıda belirtilen BasaEkle(), ArayaEkle() ve SonaEkle() fonksiyoları kullanıldı.*i[2] ,*i[3],*i[4],*i[5] Linked listi çıktı olarak ekrana vermek için void listele() fonksiyonu geliştirildi.* İ[2]

## GENEL YAPI

Bu kısımda projenin genel yapısı anlatılacaktır.

### 4.1. Kullanılanlar

Bu projede programlama dili olarak C dili ve geliştirme ortamı olarak da CodeBlocks geliştirme ortamı kullanılmıştır. Artı olarak stdio.h ve string.h kütüphaneleri kullanılmıştır.

4.1.1. String.h Kütüphanesi

String.h kütüphanesindeki kullanılan fonksiyonlar aşağıdaki gibidir.

strcmp(s,t) : s ile t stringleri aynı ise 0 döner.

strcpy(s,t) : t'yi s'ye kopyalar.

memset(Dizi,NULL,size) : Dizi'nin içini boşaltır.

### 4.2. Dosyadaki Verileri Çekme

Dosyadan veri çekerken öncelikle, proje kodunun bulunduğu klasörün içinde "dosya" adında dosyanın olup olmadığını kontrol eder.

Eğer yoksa “Dosya bulunamadı!” hatasını verir. Eğer böyle bir dosya var ise dosyayı sonuna kadar okur ve metni char dizisine atar. Daha sonrasında metin için bu char dizisi kullanılır.

### 4.3. SiraliEkle() Fonksiyonu

SiraliEkle() fonksiyonu içinde ArayaEkle(), BasaEkle(), SonaEkle() fonksiyonları kullanılmıştır. Eğer bağlantılı liste boş ise BasaEkle() fonksiyonu kullanılır. Boş değil ise listenin başından başlayarak sonuna kadar kelime adetlerini karşılaştırarak gider. Adet sayıları büyük olduğu sürece listenin bir sonraki elemanına geçerek devam eder. En sonunda bulduğu yer eğer NULL ise listenin en küçük elemanı olduğu anlaşılır ve SonaEkle() fonksiyonu çalışır. Yer, ilk eleman ise bu demektir ki listenin en büyük elemanı bu elemandır. Bu nedenle BasaEkle() fonksiyonu çalışır. Eğer yerimiz arada bir yerde ise, bulunduğu yeri ArayaEkle() fonksiyonuna gönderir. Böylece ArayaEkle() fonksiyonu çalışır.

### 4.4. BasaEkle() Fonksiyonu

Eğer listemiz boş ise gelen eleman hem ilk eleman hem de son eleman olur. Ama boş değilse gelen elemanın sonrakisi ilk eleman olur ve ilk eleman gelen eleman olarak atanır. Böylece baştaki eleman, gelenin sonrasına bağlanır ve gelen eleman başa geçirilir.

### 4.5. SonaEkle() Fonksiyonu

Eğer listemiz boş ise gelen eleman hem ilk eleman hem de son eleman olur. Ama boş değilse son elemanın sonrakisi gelen eleman olur ve gelen eleman son eleman olarak atanır. Böylece sondaki eleman artık gelen eleman olur.

### 4.6. ArayaEkle() Fonksiyonu

Bu fonksiyon parametre olarak hem yer değişkenini hem de önceki değişkenini alır. Gelen elemanımızın sonrakisi, öncekisinin sonrakisi olur. Öncekinin sonrakisini de gelen eleman yapılır. Böylece önceki elemandan sonra, gelen eleman gelir.

### 4.7. VarMi() Fonksiyonu

Bu fonksiyonda ilk elemandan başlayarak son elemana kadar, bütün elemanların kelimelerini, gelen eleman ile karşılaştırır. Eğer var ise 1, yok ise döngüden çıkar ve 0 döndürür.

### 4.8. Listele() Fonksiyonu

Bu fonksiyonda ilk elemandan son elemana kadar dolaşarak hepsini ekrana bastırır.

### 4.9. Yapılan İşlemler

Öncelikle metinin içindeki bir kelimeyi alır ve bu kelimenin bağlantılı listede geçip geçmediğini VarMi() fonksiyonu ile kontrol eder. Eğer varsa diğer kelimeye geçer. Ama eğer yoksa metinin başındaki kelimeden başlayarak bütün kelimeleri tarar ve aynı kelimeye denk geldiği zaman adet sayısını bir arttırır. Böylece, kelimenin metinde geçtiği adet belli olur. Bu adedi ve kelimeyi SiraliEkle() fonksiyonuna atar.

## YAZILIM MİMARİSİ

```
struct dugum
{
    char kelimeler[50];
    int adet;
    struct dugum *sonraki;
};

struct dugum *ilk=NULL, *son=NULL;
```

**Struct içinde kelime ve kelimelerin metin içerisinde kaç kere geçtiğini belirtmek için char kelimeler[], int adet, bir sonraki düğümün işaretçisi olarak *sonraki kullanılmıştır.**

```
int VarMi(char kelime[])
{
     struct dugum *ara=ilk;
     char kelime2[50];
     while(ara!=NULL)
     {
          strcpy(kelime2,ara->kelimeler);
          if(strcmp(kelime2,kelime)==0)
               return 1;
          ara=ara->sonraki;
     }
     return 0;
}
```

**Metin içerisinde okunan kelimenin linked listte olup olmadığını kontrol etmek için geliştirilmiştir.**

---

```
void SonaEkle(int adetsayisi,char* kelime)
{
    struct dugum *yeni = (struct dugum*) malloc(sizeof(struct dugum));
    yeni->adet=adetsayisi;
    for(int j=0; j<50; j++)
        yeni->kelimeler[j]=kelime[j];

    if(ilk==NULL)
    {
        ilk=yeni;
        son=yeni;
        son->sonraki=NULL;
    }
    else
    {
        son->sonraki=yeni;
        son=yeni;
        son->sonraki=NULL;
    }
}
```

**metinden okunan kelime ve adedinin linked list'in sonuna eklenmesi için geliştirilmiştir. SiraliEkle() fonksiyonu içinde büyükten küçüğe sıralama yaparken kullanılmıştır.**

---

```
void BasaEkle(int adetsayisi,char kelime[])
{

    struct dugum *yeni = (struct dugum*) malloc(sizeof(struct dugum));
    yeni->adet=adetsayisi;

    strcpy(yeni->kelimeler,kelime);

    if(ilk==NULL)
    {
        ilk=yeni;
        son=yeni;
        son->sonraki=NULL;
    }
    else
    {
        yeni->sonraki=ilk;
        ilk=yeni;
    }
}
```

**Metinden okunan kelime ve adedinin linked list'in başına eklenmesi için geliştirilmiştir. SiraliEkle() fonksiyonu içinde büyükten küçüğe sıralama yaparken kullanılmıştır.**

---

```
void ArayaEkle(int adetsayisi,char* kelime,struct dugum *yer,struct dugum *onceki)
{
    struct dugum *yeni=(struct dugum*) malloc(sizeof(struct dugum));
    yeni->adet=adetsayisi;
    strcpy(yeni->kelimeler,kelime);
    yeni->sonraki=onceki->sonraki;
    onceki->sonraki=yeni;

}
```

**Metinden okunan kelime ve adedinin linked list'te araya eklenmesi için geliştirilmiştir. SiraliEkle() fonksiyonu içinde büyükten küçüğe sıralama yaparken kullanılmıştır.**

---

```
void SiraliEkle(int adetsayisi,char* kelime)
{
    if(ilk==NULL)
    {
        BasaEkle(adetsayisi,kelime);
    }
    else
    {
        struct dugum *yer=ilk;
        struct dugum *onceki=(struct dugum*) malloc(sizeof(struct dugum));
        while(yer!=NULL && yer->adet>adetsayisi)
        {
            onceki=yer;
            yer=yer->sonraki;
        }

        if(yer==NULL){
            SonaEkle(adetsayisi,kelime);
        }

        else if(yer==ilk){
            BasaEkle(adetsayisi,kelime);
        }

        else{
            ArayaEkle(adetsayisi,kelime,yer,onceki);
        }

    }
```

**Linked list'in büyükten küçüğe doğru sıralanmasını sağlamak için geliştirilmiştir. İçerisinde SonaEkle(),BasaEkle(),ArayaEkle() fonksiyonları kullanılmıştır.**

---

```
void listele()
{
    int say=1;
    struct dugum *liste=ilk;
    while(liste!=NULL)
    {
        printf("%d. %s:%d\n",say,liste->kelimeler,liste->adet);
        liste=liste->sonraki;
        say++;
    }
}
```

**Linked listin ekrana çıktı vermesi için geliştirilmiştir.**

*****Fonksiyonların işlevleri 'GENEL YAPI' kısmında detaylı anlatılmıştır.**

## YAKLAŞIMLAR

C programlama yapısal bir programlama dilidir. Yapısal programlamada amaç problemi alt parçalara bölerek bu parçaların çözümlerinin birleştirilmesidir. C

programlamada yukarıdan aşağı tasarım(top down design) vardır. Yapılacak görevlere yoğunlaşır. Main() fonksiyonu içinde diğer fonksiyonlar çağırılır.

Yapısal programlamaya sahip bir dilde kontrol işlemleri (şartlar) aşağıdaki şekilde üçe ayrılır:
**1.** Akış (sequence) bir alt programdan diğerine geçiş işlemi: Bu işlemde fonksiyonlar kullanılır. Main() içinde fonksiyonların çağırılmasıdır.
**2.** iki alt programdan birisini bir bool mantık işlemine göre çalıştırmak: Bu işlemde if else if else yapıları kullanılır.
**3.** bir şart sağlanana kadar bir alt programın çalıştırılması: Bu işlemde while do while for döngüleri kullanılır.

Yukarıdaki şartları destekleyen bir dil yapısal programlama mantığına sahiptir denilebilir.

## KAZANIMLAR

Linked list mantığını, bir projede linked listin nasıl kullanılabileceğini öğrendik.

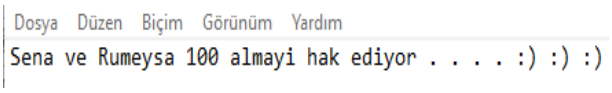
Linked list in fonksiyonlara nasıl aktarılabileceğini öğrendik.

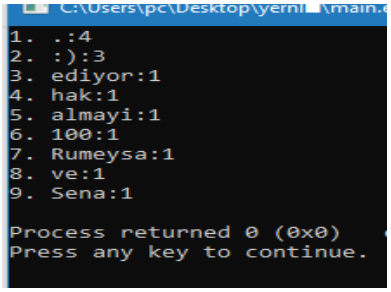
C programlamada dosyadan veri okuması yaparak ve linked list yapısı kullanarak algoritma geliştirmeyi öğrendik.

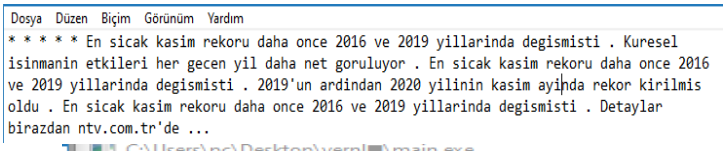
## EKSİKLİKLER

Programımız .txt dosyasından metni okurken Türkçe karakter okuyamamaktadır. Dolayısıyla Türkçe karakter kullanılarak yazılan metinde kelimeyi okurken ve bu kelimenin frekansını bulurken sorun yaşanmaktadır. Programımız doğru bir çıktı vermemektedir.

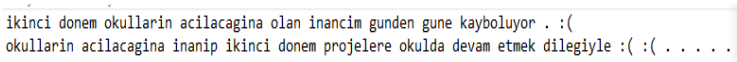
## PROGRAM ÇIKTILARI

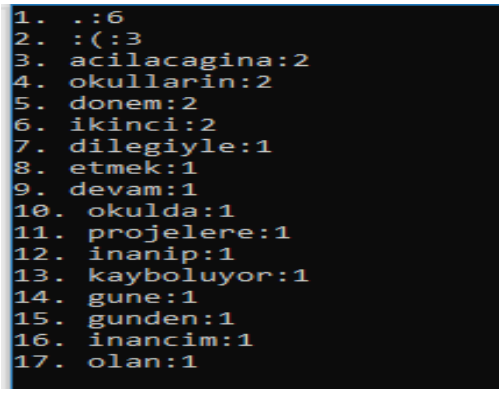







## REFERANSLAR


i[1]:https://www.bilgigunlugum.net/prog/cprog/c_dosya

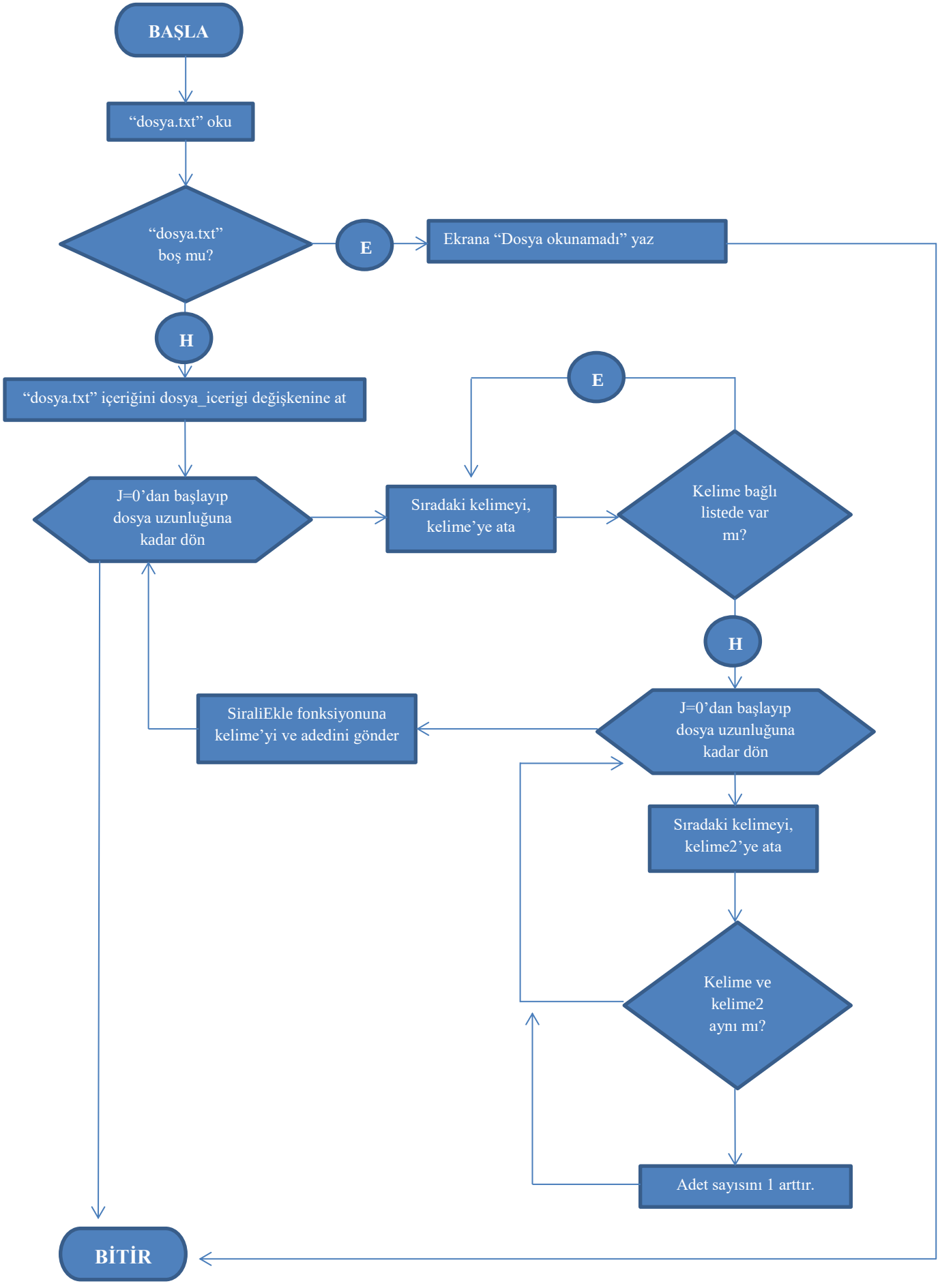
i[2]:http://edestek2.kocaeli.edu.tr/pluginfile.php/121181/mod_resource/content/0/5_VeriYapilariVeAlgoritmalar.pdf

i[3]:https://www.youtube.com/watch?v=r3uOBb3BM-0

i[4]:https://www.youtube.com/watch?v=DGi_gSfYfKo

i[5]:https://www.youtube.com/watch?v=wDAf9Er6Qq8

# AKIŞ ŞEMASI


BAŞLA
"dosya.txt" oku
"dosya.txt" boş mu?
E
Ekrana "Dosya okunamadı" yaz
H
"dosya.txt" içeriğini dosya_icerigi değişkenine at
E
J=0'dan başlayıp dosya uzunluğuna kadar dön
Sıradaki kelimeyi, kelime'ye ata
Kelime bağlı listede var mı?
H
SiraliEkle fonksiyonuna kelime'yi ve adedini gönder
J=0'dan başlayıp dosya uzunluğuna kadar dön
Sıradaki kelimeyi, kelime2'ye ata
Kelime ve kelime2 aynı mı?
Adet sayısını 1 arttır.
BİTİR