Kocaeli Üniversitesi

Bilgisayar Mühendisliği Bölümü

Programlama Laboratuvarı I

Tunay Baştürk – Umut Kılıç

190201032@kocaeli.edu.tr-190201028@kocaeli.edu.tr

1.PROBLEMİN TANIMI VE GENEL YAPISI:

Bizden txt ' ye girilen metinin kelimelerini ayırıp bağlantılı listeye kelime sayılarına göre sıralı eklenmesi istendi . Biz de problemde struct kelimeler adında bir yapı oluşturduk . İçinde bir tane char tipinde pointer oluşturduk . İnteger tipinde kelime sayısını tutan bir değişken oluşturduk . Struct kelimeler tipinde next pointerımızı oluşturduk . Biz projede tek yönlü bağlantılı liste kullandık . Struct kelimeler tipinde ilk adında bir pointer oluşturduk ve buna ilk başta NULL atadık . Bir tane dosya oluşturduk . Burdan X txt sindeki kelimeleri okutmayı sağladık . Struct kelimeler tipinde yeni isimli bir pointer oluşturduk ve buna başta NULL atadık . İnteger tipinde sayac , kelimesayisi ve kontrol adında değişkenler oluşturduk. Bir tane while döngüsüyle bu dosyanın içinde gezinmeyi ve gezinirken char tipinde x adlı bir pointer oluşturduk . Her gezişimizde kelimeleri bu pointer fscanf ile okuyarak attık . Bir tane daha dosya oluşturduk . Bunlada aynı txt ' yi okutmayı sağladık . Daha sonra bu okuttuğumuz txt de strcmp fonksiyonumuz ile x pointerımıza atadığımız kelimeleri txt yi gezerek o andaki kelimeden kaç tane olduğunu buldurduk . Struct kelimeler tipinde iter adlı bir pointer oluşturduk. Bu bağlantılı listeyi gezerek strcmp fonksiyonu[4] ile bağlantılı listede bu kelimeden olup olmadığını kontrol ettirdik . Eğer yoksa siraliekle fonksiyonu ile ekledik .

Siraliekle :

Bu fonksiyonumuz içine struct kelimeler tipinde bir pointer , char tipinde bir pointer , integer tipinde bir değişken alır . Bu fonksiyonumuz 3 koşul içerir . İlk koşulda bağlatılı listemiz boş ya da o andaki kelime sayısı göndereceğimiz kelime sayısından az ise basaEkle fonksiyonu çağırılır ve bu bağlantılı listemiz basaEkle fonksiyonuna eşitlenir ve döndürülür. Diğer koşulumuzda bağlantılı listemizin sonraki elemanı boş olmazsa arayaEkle fonksiyonu çağırılır ve bağlantılı listemize arayaEkle fonksiyonumuza eşitlenir ve döndürülür. En sondaki koşulumuzda bağlantılı listenin sonraki pointerı boş ise yani bağlantılı listemiz sona geldiyse sonaEkle fonksiyonu çağırılır ve bağlantılı listemize sonaEkle fonksiyonumuza eşitlenir ve döndürülür.[1]

basaEkle :

Bu fonksiyonumuz içine struct kelimeler tipinde bir pointer , char tipinde bir pointer , integer tipinde bir değişken alır . Eğer bağlantılı listemiz boş ise yeni bir bağlantılı liste elemanı oluşturulur ve bu bağlantılı listenin başına eklenir . Diğer koşulumuz ise bağlantılı listemizdeki kelime sayısından gönderdiğimiz kelime sayısı fazla ise yeni struct kelime tipinde bir pointer oluşturduk . Daha sonra buna gönderdiğimiz kelimeyi ve kelime sayısını eşitledik ve listenin başına eklenir. Daha sonra döndürülür . [1]

arayaEkle :

Bu fonksiyonumuz içine struct kelimeler tipinde bir pointer , char tipinde bir pointer , integer tipinde bir değişken alır . Struct kelimeler tipinde bir pointer oluşturulur ve bu oluşturduğumuz pointera bağlantılı listemiz atılır . While döngüsü ile bağlantılı listede bir sonraki elemanın boş olmadığı ve bir sonraki elemanın kelime sayısının gönderdiğimiz kelime sayısından fazla olduğumuz sürece bağlantılı listemiz ileriye doğru devam eder. Yeni struct kelime tipinde bir pointer oluşturduk . Daha sonra buna gönderdiğimiz kelimeyi ve kelime sayısını eşitledik ve listenin arasına eklenir . Daha sonra döndürülür . [3]

sonaEkle :

Bu fonksiyonumuz içine struct kelimeler tipinde bir pointer , char tipinde bir pointer , integer tipinde bir değişken alır. Struct kelimeler tipinde bir pointer oluşturulur ve bu oluşturduğumuz pointera bağlantılı listemiz atılır. While döngüsü ile bağlantılı listede bir sonraki elemanın boş olmadığı sürece bağlantılı listemiz ileriye doğru devam eder . Yeni struct kelime tipinde bir pointer oluşturduk. Daha sonra buna gönderdiğimiz kelimeyi ve kelime sayısını eşitledik ve listenin sonuna eklenir . Daha sonra döndürülür . [1]

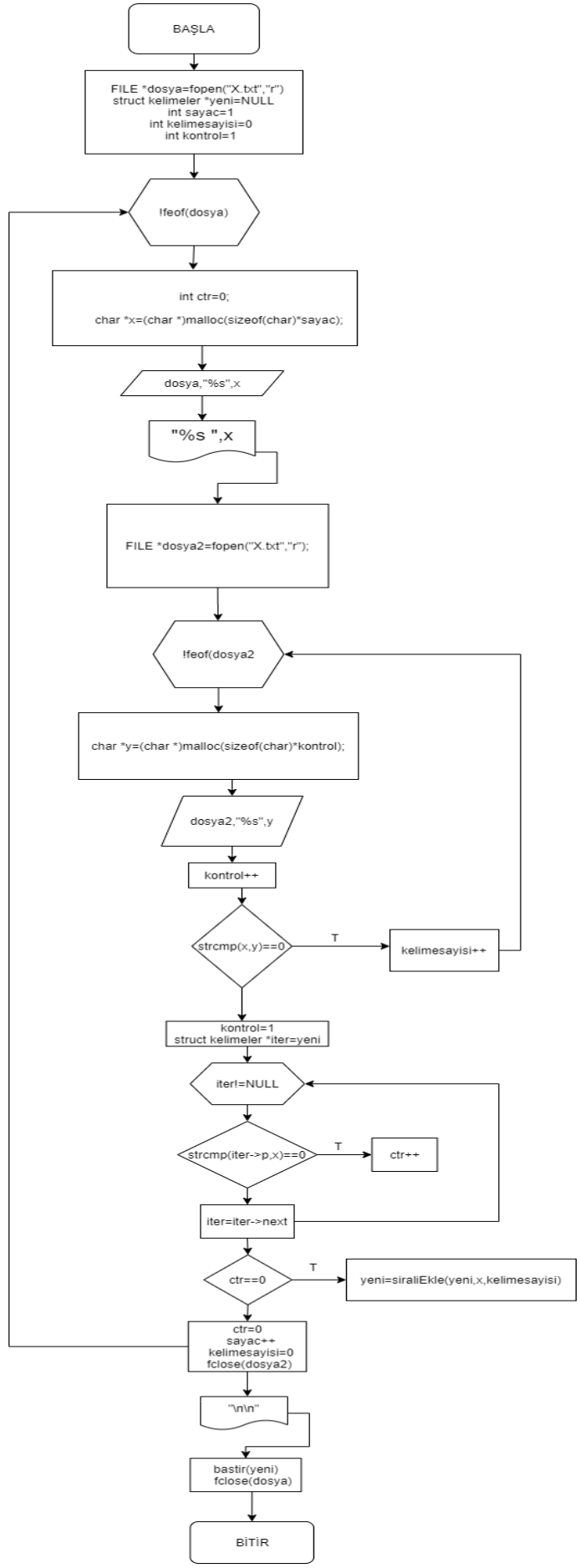
bastir :

Bu fonksiyonumuz içine struct kelimeler tipinde bir pointer alır . Ve bu pointer ilke eşitlenir . Sonra while bir while döngüsü ile bağlantılı liste NULL olmayana kadar döner ve kelimelerle kelime sayısı sıralı şekilde yazdırılır.[1]

2.YAPILAN ARAŞTIRMALAR:

Projeyi geliştirirken dizi kullanmadığımız için başlarda çok zorlandık . Bunu 2. Bir char tipinde pointer açarak bu açtığımız pointerla listemizdeki kelimeyi tutan pointerı karşılaştırmayı sağlayarak kelime sayılarını tutmayı sağladık .Bu sayede bu sorunu çözmüş olduk .

3.AKIŞ ŞEMASI:



4.YAZILIM MİMARİSİ:

```
struct kelimeler *siraliEkle(struct kelimeler *r,char *p,int x)
{
  if(r==NULL||r->kelimesayisi<x)
    {
        r=basaEkle(r,p,x);
        return r;
    }
    if(r->next!=NULL)
    {
        r=arayaEkle(r,p,x);
        return r;
    }
    if(r->next==NULL)
    {
        r=sonaEkle(r,p,x);
        return r;
    }
}
```

Siraliekle :

Bu fonksiyonumuz içine struct kelimeler tipinde bir pointer , char tipinde bir pointer , integer tipinde bir değişken alır . Bu fonksiyonumuz 3 koşul içerir . İlk koşulda bağlatılı listemiz boş ya da o andaki kelime sayısı göndereceğimiz kelime sayısından az ise basaEkle fonksiyonu çağırılır ve bu bağlantılı listemiz basaEkle fonksiyonuna eşitlenir ve döndürülür. Diğer koşulumuzda bağlantılı listemizin sonraki elemanı boş olmazsa arayaEkle fonksiyonu çağırılır ve bağlantılı listemize arayaEkle fonksiyonumuza eşitlenir ve döndürülür. En sondaki koşulumuzda bağlantılı listenin sonraki pointerı boş ise yani bağlantılı listemiz sona geldiyse sonaEkle fonksiyonu çağırılır ve bağlantılı listemize sonaEkle fonksiyonumuza eşitlenir ve döndürülür.[1]

```
struct kelimeler * basaEkle(struct kelimeler *r,char *p,int x)
{
    if(r==NULL)
    {
        r=(struct kelimeler *)malloc(sizeof(struct kelimeler));
    r->next==NULL;
    ilk=r;
    r->p=p;
    r->kelimesayisi=x;
    return r;
    }
    if(r->kelimesayisi<x)
    {
        struct kelimeler *y=(struct kelimeler*)malloc(sizeof(struct
kelimeler));
        y->p=p;
        y->kelimesayisi=x;
        y->next=r;
        ilk=y;
        return y;
    }
}
```

basaEkle :

Bu fonksiyonumuz içine struct kelimeler tipinde bir pointer , char tipinde bir pointer , integer tipinde bir değişken alır . Eğer bağlantılı listemiz boş ise yeni bir bağlantılı liste elemanı oluşturulur ve bu bağlantılı listenin başına eklenir . Diğer koşulumuz ise bağlantılı listemizdeki kelime sayısından gönderdiğimiz kelime sayısı fazla ise yeni struct kelime tipinde bir pointer oluşturduk . Daha sonra buna gönderdiğimiz kelimeyi ve kelime sayısını eşitledik ve listenin başına eklenir. Daha sonra döndürülür . [1]

```
struct kelimeler * arayaEkle(struct kelimeler *r,char *p,int x)

{

   struct kelimeler *iter=r;

    while(iter->next!=NULL&&iter->next->kelimesayisi>x)

    {

        iter=iter->next;

    }

    struct kelimeler *degisken=(struct kelimeler *)malloc(sizeof(struct kelimeler));

    degisken->next=iter->next;

    iter->next=degisken;

    degisken->kelimesayisi=x;

    degisken->p=p;

    return r;

}
```

arayaEkle :

Bu fonksiyonumuz içine struct kelimeler tipinde bir pointer , char tipinde bir pointer , integer tipinde bir değişken alır . Struct kelimeler tipinde bir pointer oluşturulur ve bu oluşturduğumuz pointera bağlantılı listemiz atılır . While döngüsü ile bağlantılı listede bir sonraki elemanın boş olmadığı ve bir sonraki elemanın kelime sayısının gönderdiğimiz kelime sayısından fazla olduğumuz sürece bağlantılı listemiz ileriye doğru devam eder. Yeni struct kelime tipinde bir pointer oluşturduk . Daha sonra buna gönderdiğimiz kelimeyi ve kelime sayısını eşitledik ve listenin arasına eklenir . Daha sonra döndürülür . [3]

```
struct kelimeler * sonaEkle(struct kelimeler *r,char *p,int x)

{

    struct kelimeler *iter=r;

    while(iter->next!=NULL)

    {

        iter=iter->next;

    }

    struct kelimeler *degisken=(struct kelimeler *)malloc(sizeof(struct kelimeler));

    degisken->next=iter->next;

    iter->next=degisken;

    degisken->kelimesayisi=x;

    degisken->p=p;

    return r;

}
```

sonaEkle :

Bu fonksiyonumuz içine struct kelimeler tipinde bir pointer , char tipinde bir pointer , integer tipinde bir değişken alır. Struct kelimeler tipinde bir pointer oluşturulur ve bu oluşturduğumuz pointera bağlantılı listemiz atılır. While döngüsü ile bağlantılı listede bir sonraki elemanın boş olmadığı sürece bağlantılı listemiz ileriye doğru devam eder . Yeni struct kelime tipinde bir pointer oluşturduk. Daha sonra buna gönderdiğimiz kelimeyi ve kelime sayısını eşitledik ve listenin sonuna eklenir . Daha sonra döndürülür . [1]

```
void bastir(struct kelimeler *p)
{
    p=ilk;
    int sayac=0;
    while(p!=NULL)
    {   printf("%d.  %s  %d",sayac+1,p->p,(p->kelimesayisi));
        printf("\n");
        p=p->next;
        sayac++;
    }
}
```

bastir :

Bu fonksiyonumuz içine struct kelimeler tipinde bir pointer alır . Ve bu pointer ilke eşitlenir . Sonra while bir while döngüsü ile bağlantılı liste NULL olmayana kadar döner ve kelimelerle kelime sayısı sıralı şekilde yazdırılır.[1]

```
struct kelimeler
{
    char *p;
    int kelimesayisi;
    struct kelimeler *next;
};
```

Bağlantılı Listemiz :

Struct kelimeler adında bir yapı oluşturduk . İçinde bir tane char tipinde pointer oluşturduk . İnteger tipinde kelime sayısını tutan bir değişken oluşturduk . Struct kelimeler tipinde next pointerımızı oluşturduk . [1][2]

5.REFERANSLAR:

[1]https://kocaeli.zoom.us/rec/play/5od9PibMVHCsDKu2Y3fGjQ0uyqFgtcIdr_CAYve_AzLwdh9Exn2EGqlDakOHfjXmGZUXmixsRbr38lzx.tvCykWj8afIaEWt0?continueMode=true&_x_zm_rtaid=DnOjve99REevC-g4iOrZlA.1609366711978.57376fc9f1752f6237e3ac62bb8f896e&_x_zm_rhtaid=138

(ONUR HOCA)

[2]https://www.youtube.com/watch?v=r3uOBb3BM-0&t=800s

[3]https://www.youtube.com/watch?v=wDAf9Er6Qg8

[4]https://www.bilgigunlugum.net/prog/cprog/2c_dizi

[5] https://www.yusufsezer.com.tr/c-turkce-karakter-kullanmak/