BLM3021
ALGORİTMA ANALİZİ

ÖDEV-2

Hashing Algoritması nın Kullanımı

17011701 Onur KOÇKAN

Problem: Bu ödevde, bir kelimenin geçtiği dokümanları listeleyen bir sistem tasarlanacaktır. Bir kelimenin hangi dokümanlarda geçtiğini bulmak için her seferinde bütün dokümanlara bakmak çok zaman alıcıdır. Bunun yerine bu ödevde, yeni gelen bir dokümandaki kelimeler hashing ile bir sözlüğe yerleştirilecek ve bir kelime arandığında yine hashing yö ntemi ile sözlükte aranarak içinde yer aldığı dokümanlar bulunacaktır.

ÇÖZÜM

Fonksiyonlar ve yapılan işlemler:

int h (int key , int i) Fonksiyonu :

Bu fonksiyonda gelen key ve sayaç değerine göre double hash işlemi gerçekleştiriyoruz. Double hash'de iki farklı hash'den geçirerek hash table'da yerleştirilecek indis bulunur.

int insert (kv data) Fonksiyonu :

Hash table'a ekleme yaparken fonksiyona gelen key, kelime ve dosya adını hash table'a eklemek için gerekli işlemler yapılıyor. Öncelikle gelen ilk key değeri double hash yapılmak üzere h fonksiyonuna yollanıyor ve oradan gelen indis değerine göre eğer daha önce hash table'da o indis boş ise direk ekleyecek fakat o indiste daha önce bir key değeri eklenmişse o zaman i değeri tablo boyunu aşmayacak şekilde arttırılıp boş indis bulunana kadar arttırılır. Ayrıca burada Load Factor değerini de ekrana her aşamada yazdırmış olduk.

void search (int key) Fonksiyonu :

Parametre olarak gelen key değerini double hash fonksiyonundan indis sayısını döndürerek tekrar hash tablosunda o indise ait veriyi çekiyoruz bunun yanında kaç adımda işlemin tamamlandığını yazıdırıyoruz. Eğer eşleşen bir key değeri yok ise kelimenin hash tablosunda olmadığını kullanıcıya mesaj çıktısı vererek gösterir ayrıca bu işleminde kaç adımda tamamlandığı yazdırılır.

void printArray ( ) Fonksiyonu :

Hash tablosundaki elemanları ekrana yazdırıyoruz bu fonksiyon aracılığıyla. Tablodaki indislerde key değeri -1 olan tüm satırlarda key değeri bulunmuyor bu yüzden -1 olmayan tüm satırları dosyaya yazabiliriz.

void createHashTableFile ( ) Fonksiyonu :

Hash tablosu için file'ı burada oluşturuyoruz. Okul numarası.txt şeklinde w+ komutuyla dosya oluşturuluyor. Tablodaki indislerde key değeri -1 olan tüm satırlarda key değeri bulunmuyor bu yüzden -1 olmayan tüm satırları dosyaya yazabiliriz.

void resetHashTable ( ) Fonksiyonu :

Hash table'ı sıfırlamak için bu fonksiyonu kullanıyoruz. M olan dizi boyutu kadar döngüde ilerleyip ayrıca boş olan indisleri != -1 ler ile sorgulayarak ilgili keylere -1 atanır. Ayrıca hash tablosundaki kelime ve dosya ismi dizilerinin karakterlerini temizliyoruz.

void createExistTable (char dokumanAdi [50]) Fonksiyonu :

Önceki tabloda var olan hash table ' ı dizimize kaydediyoruz ve dökümanımızı açıyoruz. Hash fonksiyonundan geçirerek key (hash value) çıkarttıktan sonra bunu tutan keyGetir değişkenini yeniden hash table indis olarak kullanıyoruz. Dosyanın sonuna kadar okuma yapıp ilk operand key, ikinci operand kelime ve son operand ise bulunduğu doküman adı. Böylelikle var olan tabloyu dosyadan çekerek yeniden programa aktarmış olduk.

void main (int argc , char *argv [ ]) Fonksiyonu :

Öğrenci no değerini öncelikle dokumanAdi değişkenine aktarıyoruz. Daha sonra kullanıcıya arayüzde yapacağı işlem için seçim işlemi yaptırıyoruz. Eğer mevcut bir hash table var ise yeni doküman girmek için 2 eğer kelime arama yapmak istiyorsa 1 yi tercih edecek. 1 seçildiğinde var olan bilgileri hash table'a aktarıyoruz ve arama yapacağımız kelimeyi giriyor kullanıcı ve gelen kelimenin Horner Metoduna göre key değerini buluyoruz daha bulduğumuz key değerini search fonksiyonuna yollayıp orada double hash yaparak indis kontrolü ile kelimenin var olup olmadığını bulur.

Kullanıcı 2 yi tercih ettiğinde gerçekleşecek senaryolar;

- Mevcut hash table dosyasını kullanarak yeni doküman ekleyip kelime aratılır.
- Dökümandakiler ile yeni hash tablosu oluştur.
- Kullanıcıdan yeni doküman ismi alınıyor
- Hash tablosu güncelleniyor
- Arama Yapılıyor

Öncelikle doküman adı istenir kullanıcıdan daha sonra dosya adı girildiğinde dosyadan okuma işlemi yapılır. Dosyanın sonuna kadar okuma yapılır ve dosyadaki kelimeler için horner metodu kullanılarak bir key değeri oluşturulur. Daha sonra bu key değerini geçici bir struct yapısında tutup insert fonksiyonumuza bu bilgileri yolluyoruz. Ayrıca burada da arama işlemi yapılabiliyor kelimenin dökümanda var olup olmadığı sorgusunu yapıyoruz.

Karmaşıklık analizi:

1)Insert(Ekleme)işlemi:

- Döküamdaki sayi N ise,karmaşıklık=O(n)

2)Kelime arama işlemi:

- Best casede bu indisdeki adres,keyi verilen kelimeyle aynı adresdedir ve karmaşıklığı=O(1)
- Wors casede ise aramak için tüm tabloyu gezer,Tablo boyutu=997 => Karmaşıklık=O(997)

Kodlar:

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<math.h>
#define M 997

typedef struct kelimeVerileri{//tm kelimelerimiz icin olusturdugumuz struct yapimiz
    int key;    //unique olusacak keyimiz
    char value[30]; //kelimenin kendisi
    char fileName[30];  //bulundugu dokuman
}kv;

kv a[M];    //kelimeleri tutacak M(1000 sayisina en yakn asal say=997) boyutlu struct dizimiz.
int N;  //Load factor icin eleman sayisi

int h(int key,int i){//Istenlen formuldeki hash fonksiyonumuz.
    int h1=key%M;
    int h2=1+(key%(M-1));
    return (h1+i*(h2))%M;
}


int insert (kv data){   //Hash tablosuna ekleme islemi.
    char tempFileName[30];  //gecici olarak dosya ismini tutacak degiskenimiz.
    int i = 0;
    int keyGetir = h(data.key,i);   //keyGetir degiskeni ile kv tipinde data isimli verimizin keyi ve i degiskeni parametreleri ile hash key olusturulur.
    if(a[keyGetir].key==data.key && a[keyGetir].fileName!=data.fileName){   //olusan indiste bulunan elemann key degeri mevcut key degerine esit ve dokuman isimleri farkl ise
        //ikinci dokuman mevcut dokumann devamna ekle
        int tempL=strlen(a[keyGetir].fileName);
        tempL++;
        a[keyGetir].fileName[tempL]=',';//mevcut dokumann sonuna gel ve , koy
        tempL=strlen(data.fileName);
        int k,z=0;

for(k=strlen(a[keyGetir].fileName);k<tempL+strlen(a[keyGetir].fileName);k++){//ilk dokuman isminin boyutu kadar basla ve 2 kelimenin boyutu kadar git devamna
            a[keyGetir].fileName[k]=data.fileName[z];//ikinci dokuman yazdir.
            z++;
        }
    }
    if(a[keyGetir].key==-1){//eger bu data hic eklenmemis ise
        //tum degiskenleri ile olusan hash valuedeki indise ekle
        a[keyGetir].key=data.key;
        strcpy(a[keyGetir].value,data.value);
        strcpy(a[keyGetir].fileName,data.fileName);
        N++;//LF
        printf("Load factor=%f\n",(double)N/997);
    }
    else{//bunlardan hicbiri degil ise
        while((a[keyGetir].key!=-1)&&(a[keyGetir].key!=data.key)&&(i<M)){//bos yer bulana kadar iteratif bir sekilde ilerle ve
            i++;
            keyGetir = h(data.key,i);//en son olusturdugun keyi ve indisi tut
        }
        //bu indise eleman ekle.
        a[keyGetir].key=data.key;
        strcpy(a[keyGetir].value,data.value);
        strcpy(a[keyGetir].fileName,data.fileName);
        N++;//LF
        printf("Load factor=%f\n",(double)N/997);
    }
```

```
}

void search(int key){
    int i = 0,step=0;//step degiskenimiz adim sayaci olarak kullanilmasi icin
tanimlaniyor.
    int keyGetir = h(key,i);//search yapmamiz icin keyGetir isimli degiskeni tekrar
olusturuyoruz.
    if(a[keyGetir].key==key){//olusan keydeki hash valuemiz parametre olarak gelen key
degerine esit ise tek adimda bulunmustur.Direkt ekrana bastir.
        printf("Kelime:%s,Dokuman:%s    1 Adimda
bulundu\n",a[keyGetir].value,a[keyGetir].fileName);
    }
    else{//tekde bulamadysa eger
        int i,chA=0;//chA degiskenimiz switch gorevi gorucek.
        for(i=0;i<M;i++){
            step++; //her bulma isleminde 1 arttiriliyor.
            if(a[i].key==key){//mevcut i indisli eleman istedigimiz elemansa
                //kac adimda bulundugu gerekli bilgileri ile ekrana bastiriliyor
                printf("Kelime:%s,Dokuman:%s    %d adimda
bulundu\n",a[keyGetir].value,a[keyGetir].fileName,step);
                chA=1;//eger kelime bulunduysa bu switch gorevi goren degiskenimizi
1liyoruz
            }
        }
        if(chA==0)//eger bu kontrol degiskenimiz hic degismediyse bu kelime tabloda yok
demektir.
            printf("Bu kelime sozlukte(Hash tablosunda) yok\n");
    }
}

void printArray(){//tum elemanlari ekrana bastiran fonksiyonumuz
    int i=0;
    for(i=0;i<M;i++){//dizi boyutu kadar ilerler
        if(a[i].key!=-1){//eger burda bir eleman var ise yazdirir.
            printf("Key:%d , Value:%s,
Dokuman:%s\n",a[i].key,a[i].value,a[i].fileName);
        }
    }
}

void createHashTableFile(){//hash tableimizin file'ini olusturan fonksiyon.
    FILE* fp2;
    fp2 = fopen("17011701.txt","w");//istenilen formatda write formatinda dosya
aciliyor.
    int i=0;
    for(i=0;i<M;i++){
        if(a[i].key!=-1){//var olan tm elemanlar tm bilgileri ile yaziliyor.
            fprintf(fp2,"%d:%s:%s\n",a[i].key,a[i].value,a[i].fileName);
        }
    }
    //fscanf(fp2,"\nLoad factor=%f",(double)N/997);
    fclose(fp2);//dosya kapatiliyor.
}

void resetHashTable(){//hash tablein sifirlanmasi icin olusturulan fonksiyon.
    int i,j;
    for(i=0;i<M;i++){//dizi boyutu kadar ilerle
        if(a[i].key!=-1){//eger bu indisde eleman var ise
            a[i].key=-1;    //keyini -1le
            for(j=0;j<30;j++){  //kelime yani value degeri ve fileName yani dokuman
ismi degeri 30 boyutunda oldugu icin 30 adim ilerle ve tum char degerlerini sil.
                a[i].value[j]='\0';
                a[i].fileName[j]='\0';
            }
```

```
        }
    }
}

void createExistTable(char dokumanAdi[50]){//onceki dosyadan var olan hash table i
dizimize kaydediyoruz.
    kv data;//structimizi tanimliyoruz.
    FILE* fprc=fopen(dokumanAdi,"r");   //dokumanimizi aciyoruz(parametre olarak
dokumanAdi verilmis fakat 17011701.txt seklinde de direkt yazilabilinirdi ve
parametresiz calistirilabilirdi).
    int i=0;
    int keyGetir;//hash fonksiyonundan gecirerek bir key(hash value) cikarttiktan sonra
bunu tutmayi sagliyan degiskenimiz.
    while(!feof(fprc)){//dosya sonuna kadar ilerle
        fscanf(fprc,"%d:%s:%s",&(data.key),data.value,data.fileName);//ilk operand
key,ikinci operand vallue(kelime) ve son operand ise bulundugu dokuman adi
        N++;//tablodaki kelime sayisinin sayilmasi.
        keyGetir = h(data.key,i);//hash value olusturuyoruz
        //ve bu indisdeki elemana ekleme yapiyoruz(alttaki 3 satir)
        a[keyGetir].key=data.key;
        strcpy(a[keyGetir].value,data.value);
        strcpy(a[keyGetir].fileName,data.fileName);
    }
    fclose(fprc);
    printf("Dosya algilandi ve Hash Table Olusturuldu\n");
}

int main(int argc, char *argv[]) {
    FILE* fpHash;
    FILE* fp;
    char dokumanAdi[50];
    strcpy(dokumanAdi,"17011701.txt");//ogrenci no degerini dokuman adi olarak
dokumanAdi degiskenine atiyoruz.
    fpHash = fopen(dokumanAdi,"r");//kontrol icin bu dosya aciliyor
    int ch=0;//programin ana mensndeki secenekler icin kontrol degiskeni
    resetHashTable();//tum islemlerden once hash tableimiz sifirlaniyor.
    if(fpHash != NULL){//egerki mevcut bir hash table var ise
        printf("Mevcut bir Hash Table dosyasi var bu dokuman ile sadece kelime aramak
icin=1,yeniden dokuman ekleyerek arama yapmak icin=2 secimlerini yapin=");
        scanf("%d",&ch);
        createExistTable(dokumanAdi);//o hash table dosyasindan dizimizi olusturuyoruz
ve yeni hash tableimizi kuruyoruz.
        if(ch==1){//MEVCUT DOSYADAKI VERILERDEN HASH TABLOSU OLUSTURULUR VE SADECE
ARAMA YAPILIR.
            //DOKUMANDAKILER ILE YENI HASH TABLOSUNU OLUSTUR
            fclose(fpHash);//Kontrol icin acilan dosyayi kapatiyoruz
            //ARAMA ISLEMI YAPILACAK
            char kelime[30];//kullanicinin aratmak istedigi kelimeyi tutacak olan
degiskenimiz.
            printf("Lutfen aramak istediginiz kelimeyi girin:");
            scanf("%s",kelime);//gets(kelime);
            //HORNER METOD ILE KELIMENIN SAYIYA CEVRILME ISLEMI:
            int j=0;
            int i=0;
            kv data;//structimiz data ismiyle tanimlaniyor.
            unsigned int key=0;
            int L = strlen(kelime);//kelimenin boyutu
            int power = L-1;//us alimi icin horner metodu formulu geregi bir eksigi.
            char temp;//gecici karakter tutulmasi icin
            for(j=0;j<L;j++){//kelime boyutu kadar ilerle ve her karakteri kontrol et
                temp = kelime[j];
                if (temp >= 'a')//case sensitive probleminin duzeltilmesi islemi
                    temp = 'A' - 'a' + temp;

                key += (pow(2,power)*(temp-'A'+1));//dongu sonuna kadar key olusur
                power--;
            }
```

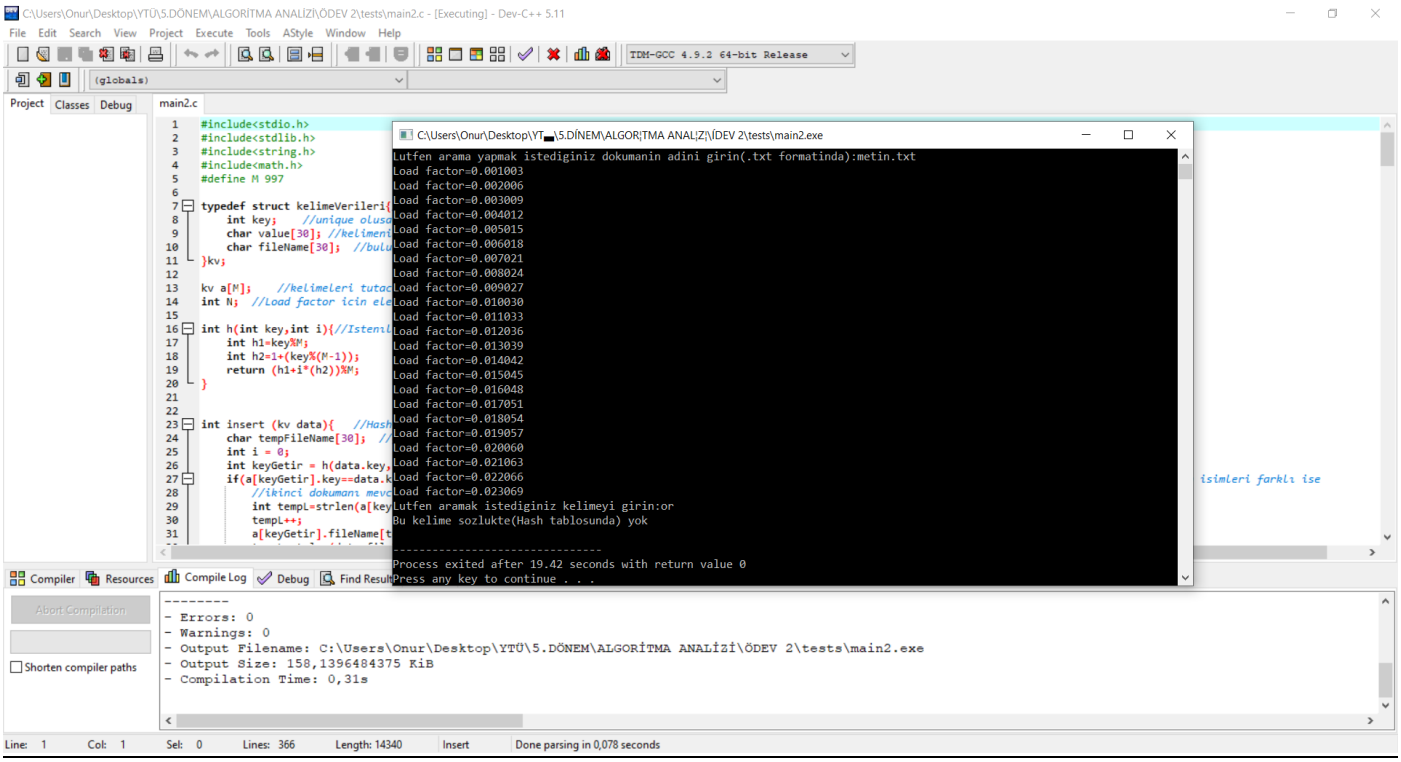
```
            if (key != 0){//cikan key 0dan farkli ise aranmasi icin search fonksiyonuna
parametre olarak gonderilir.
                search(key);
            }



        }
        else if(ch==2){//MEVCUT HASH TABLE DOSYASINI KULLANARAK YENI DOKUMAN EKLEYIP
KELIME ARATILIR.
            //DOKUMANDAKILER ILE YENI HASH TABLOSU OLUSTUR
            //KULLANICIDAN YENI DOKUMAN ISMI AL
            //HASH TABLOSUNU GUNCELLE
            //ARAMA YAP
            fclose(fpHash);//Kontrol icin acilan dosyayi kapatiyoruz

            char dokumanAdi[50];
            char kelime[30];
            //kullanicidan dokumani al.
            printf("Lutfen arama yapmak istediginiz dokumanin adini girin(.txt
formatinda):");
            scanf("%s",dokumanAdi);//gets(dokumanAdi);

            //dokumandaki kelimeleri al.
            //sozluge yani hash tablosuna yerlestir.
            //aranacak kelimeyi kullanicidan al.


            //dosya ismi yanlis verilmesi yada farkli bi hata olusmasinin kontrolu icin
acilan dosya NULL ise program calismaz.
            fp = fopen(dokumanAdi,"r");
            if(fp == NULL){
                printf("Dosya acilirken hata olustu!");
            }

            int j=0;
            int i=0;
            kv data;//data isimli structimizi olusturuyoruz.
            unsigned int key=0;
            char temp[50];
            char temp2[50];
            while(!feof(fp)){//Dosyanin sonuna kadar git
                temp[i] = fgetc(fp);//karakteri temp degiskenine at
                if(temp[i] == ' ' || temp[i] == '\n'){//bosluklar yada satir sonlarina
gelindigi durumlarda kelime olusmus demektir
                    int length = strlen(temp2);//boyutunu aliyoruz.
                    int power = length-1;//horner metoddaki formul geregi ussun bir
eksigini aliyoruz.
                    key = 0;
                    data.key=0;//key degiskeni ile birlikte dataninda key degiskenini
sifirliyoruz.
                    char temp3;//kelimedeki harfin case sensitive kontrolu icin olusan
3.temp degiskeni.
                    for(j=0;j<length;j++){
                        temp3 = temp2[j];//olusan kelimedeki karakterleri tek tek temp3
degiskenine atiyoruz.
                        if (temp3 >= 'a')//case sensitive kontrolu.
                            temp3 = 'A' - 'a' + temp3;

                        key += (pow(2,power)*(temp3-'A'+1));//horner metod formulu
                        power--;//dongu devam ettikce her adiminda us bir azaltilir.
                    }
                    if (key != 0){//key 0 degil ise olusmus demektir.
                        //dataya elde edilen degerler atilir
                        data.key=key;
                        strcpy(data.value,temp2);
                        strcpy(data.fileName,dokumanAdi);
```

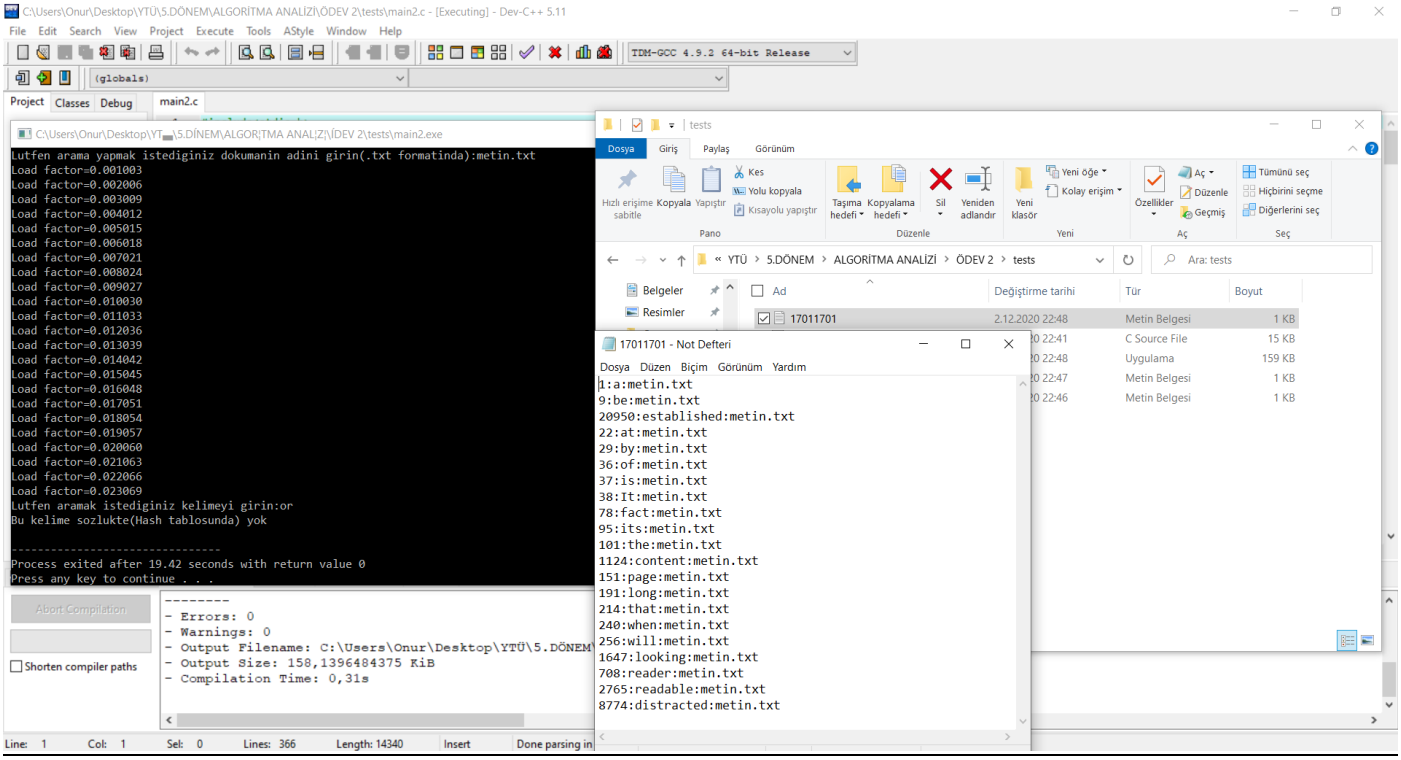
```
                        //diziye insert edilir.
                        insert(data);

                    }
                    for(j=0;j<50;j++){
                        //kelimeyi tutan temp2 dizisi temizlenir.Bununla birlikte
datanin value diziside temizlenir.
                        temp2[j] = '\0';
                        data.value[j]='\0';
                    }
                    i=0;

                }
                else{//BOSLUGA YADA SATIR SONUNA GELINMEDIYSE KELIME OKUNMAYA DEVAM
EDIYOR DEMEKTIR
                    temp2[i] = temp[i];//temp2 degiskeni kelimeyi tutar,temp degiskeni
karakteri tutar.Bu adimlar sayesinde temp2de bir kelime olusacaktir.
                    i++;
                }
            }
            fclose(fp);
            //Hash tablosunu dosya olarak olustur;
            createHashTableFile();



            printf("Lutfen aramak istediginiz kelimeyi girin:");
            scanf("%s",kelime);//gets(kelime);
            //HORNER METOD ILE KELIMENIN SAYIYA CEVRILME ISLEMI:
            int length = strlen(kelime);
            int power = length-1;
            key = 0;
            char gecici;
            for(j=0;j<length;j++){
                gecici = kelime[j];
                if (gecici >= 'a')
                    gecici = 'A' - 'a' + gecici;

                key += (pow(2,power)*(gecici-'A'+1));
                power--;
            }
            if (key != 0){
                search(key);

            }

        }

        else{//MEVCUT DOSYA OLMASINA RAGMEN KULLANICI VERMESI GEREKEN SECENEKLER
DISINDA BIR GIRIS YAPARSA EGER PROGRAM SONLANDIRILIR.
            printf("Lutfen sadece 1 veya 2 seceneklerini kullaniniz!");
        }
    }
    else {//(fpHash == NULL)MEVCUT OLAN BIR DOKUMAN YOK VE DOKUMAN ISMI KULLANICIDAN
ALINIP KELIMELERDEN HASH TABLOSU VE DOSYASI OLUSTURULUP ARAMA YAPILIR.
        fclose(fpHash);//Kontrol icin acilan dosyayi kapatiyoruz
        resetHashTable();//hash tableimizi temizliyoruz.

        char dokumanAdi[50];
        char kelime[30];
        //kullanicidan dokumani al.
        printf("Lutfen arama yapmak istediginiz dokumanin adini girin(.txt
formatinda):");
        scanf("%s",dokumanAdi);//gets(dokumanAdi);

        //dokumandaki kelimeleri al.
        //sozluge yani hash tablosuna yerlestir.
```

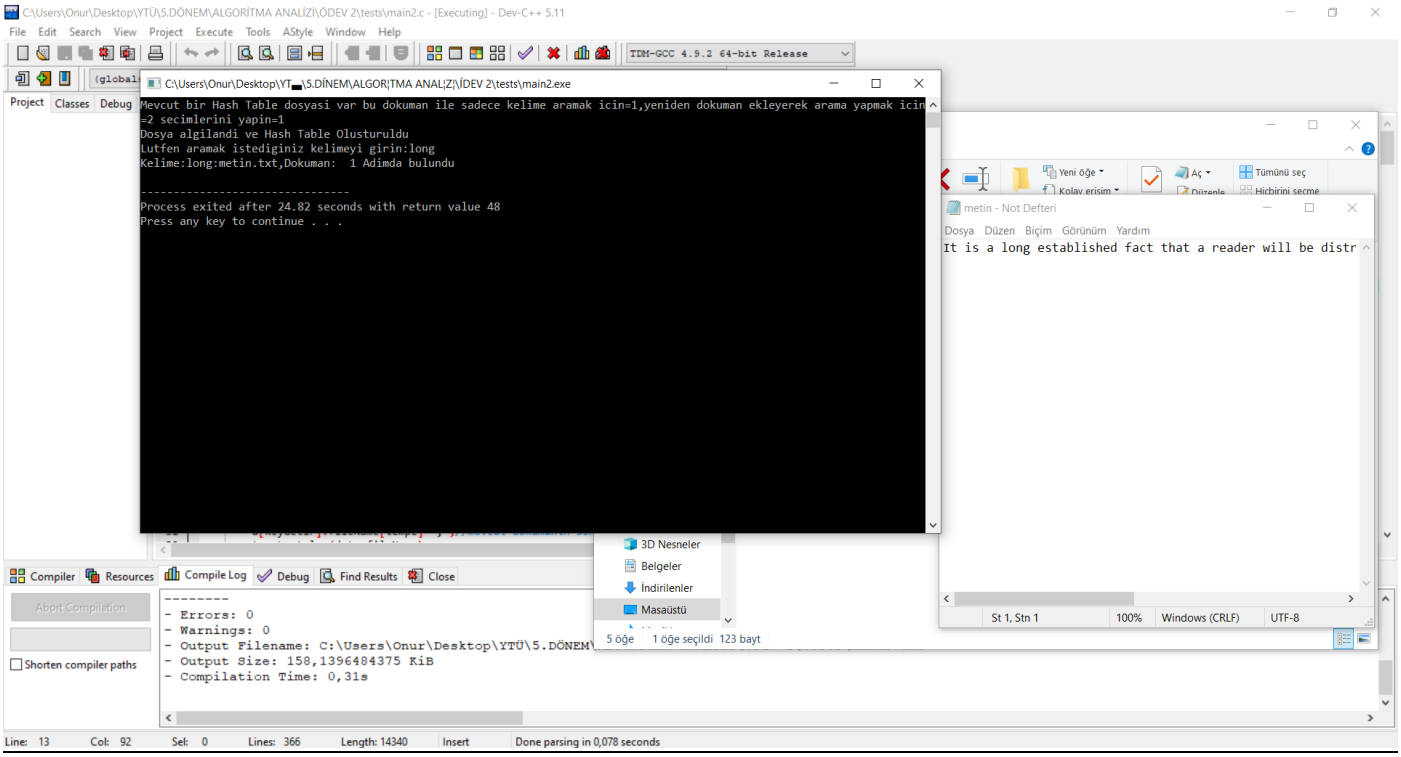
```
        //aranacak kelimeyi kullanicidan al.


        fp = fopen(dokumanAdi,"r");
        if(fp == NULL){
            printf("Dosya acilirken hata olustu!");
        }

        int j=0,i=0;
        kv data;//294.satirdan 335.satira kadarki kod blogu = 197.satirdan 243u satira
kadar anlatilan ayn kod blogudur.
        unsigned int key=0;
        char temp[50];
        char temp2[50];
        while(!feof(fp)){
            temp[i] = fgetc(fp);
            if(temp[i] == ' ' || temp[i] == '\n'){
                int length = strlen(temp2);
                int power = length-1;
                key = 0;
                data.key=0;
                char temp3;
                for(j=0;j<length;j++){
                    temp3 = temp2[j];
                    if (temp3 >= 'a')
                        temp3 = 'A' - 'a' + temp3;

                    key += (pow(2,power)*(temp3-'A'+1));
                    power--;
                }
                if (key != 0){
                    data.key=key;
                    strcpy(data.value,temp2);
                    strcpy(data.fileName,dokumanAdi);
                    insert(data);

                }
                for(j=0;j<50;j++){
                    temp2[j] = '\0';
                    data.value[j]='\0';
                }
                i=0;

            }
            else{
                temp2[i] = temp[i];
                i++;
            }
        }
        fclose(fp);
        //Hash tablosunu dosya olarak olustur;
        createHashTableFile();



        printf("Lutfen aramak istediginiz kelimeyi girin:");
        scanf("%s",kelime);//gets(kelime);
        //HORNER METOD ILE KELIMENIN SAYIYA CEVRILME ISLEMI:
        int length = strlen(kelime);
        int power = length-1;
        key = 0;
        char gecici;
        for(j=0;j<length;j++){
            gecici = kelime[j];
            if (gecici >= 'a')
                gecici = 'A' - 'a' + gecici;
```

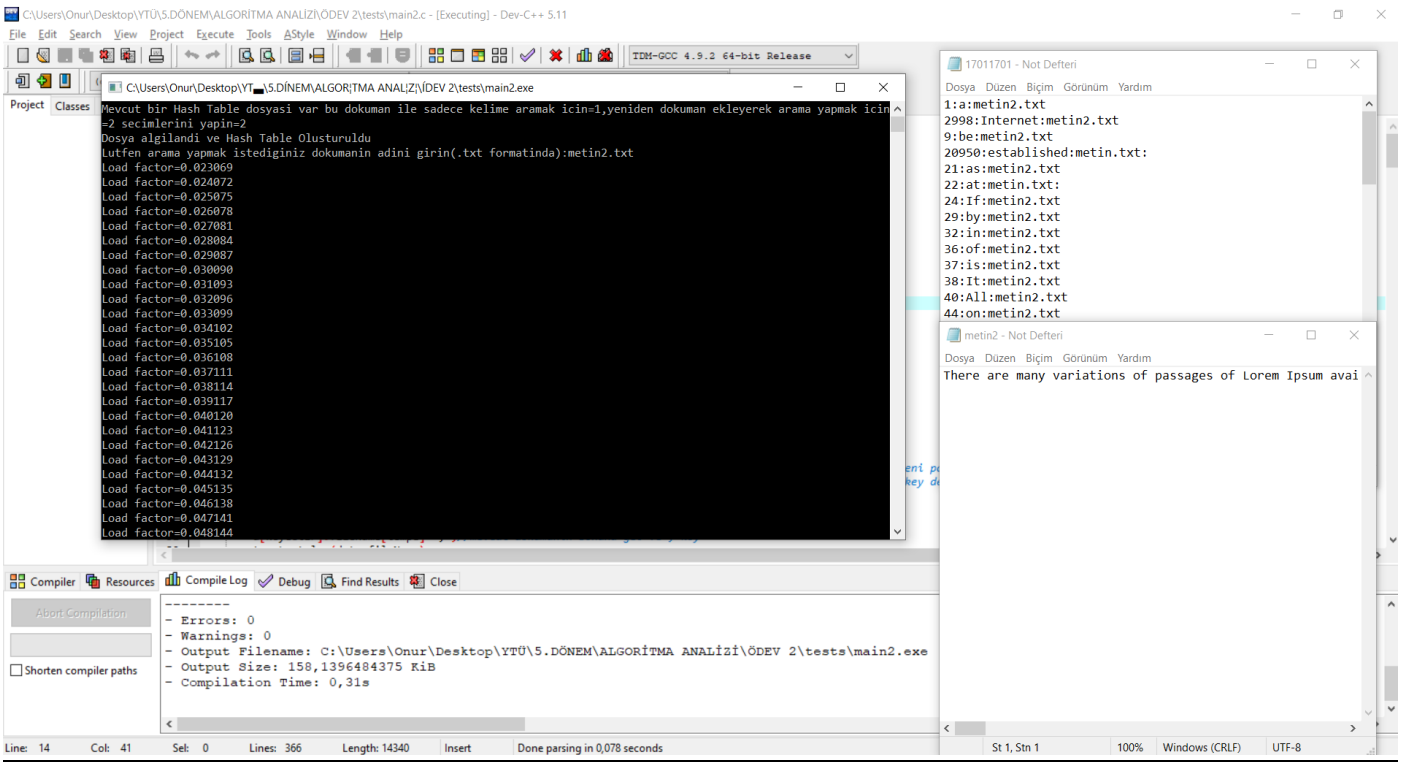
```
                key += (pow(2,power)*(gecici-'A'+1));
                power--;
            }
            if (key != 0){
                search(key);

            }
        }
}
```

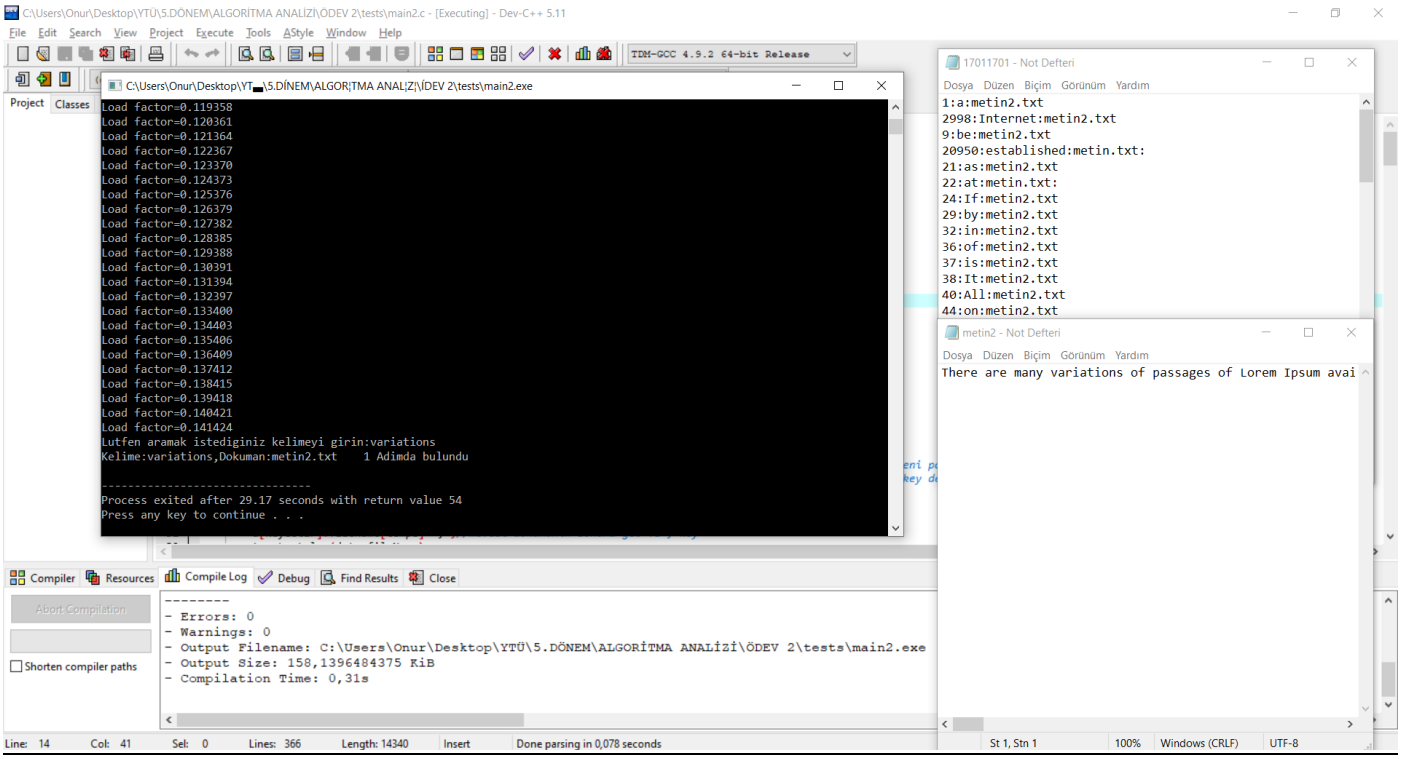
Ekran çıktıları:



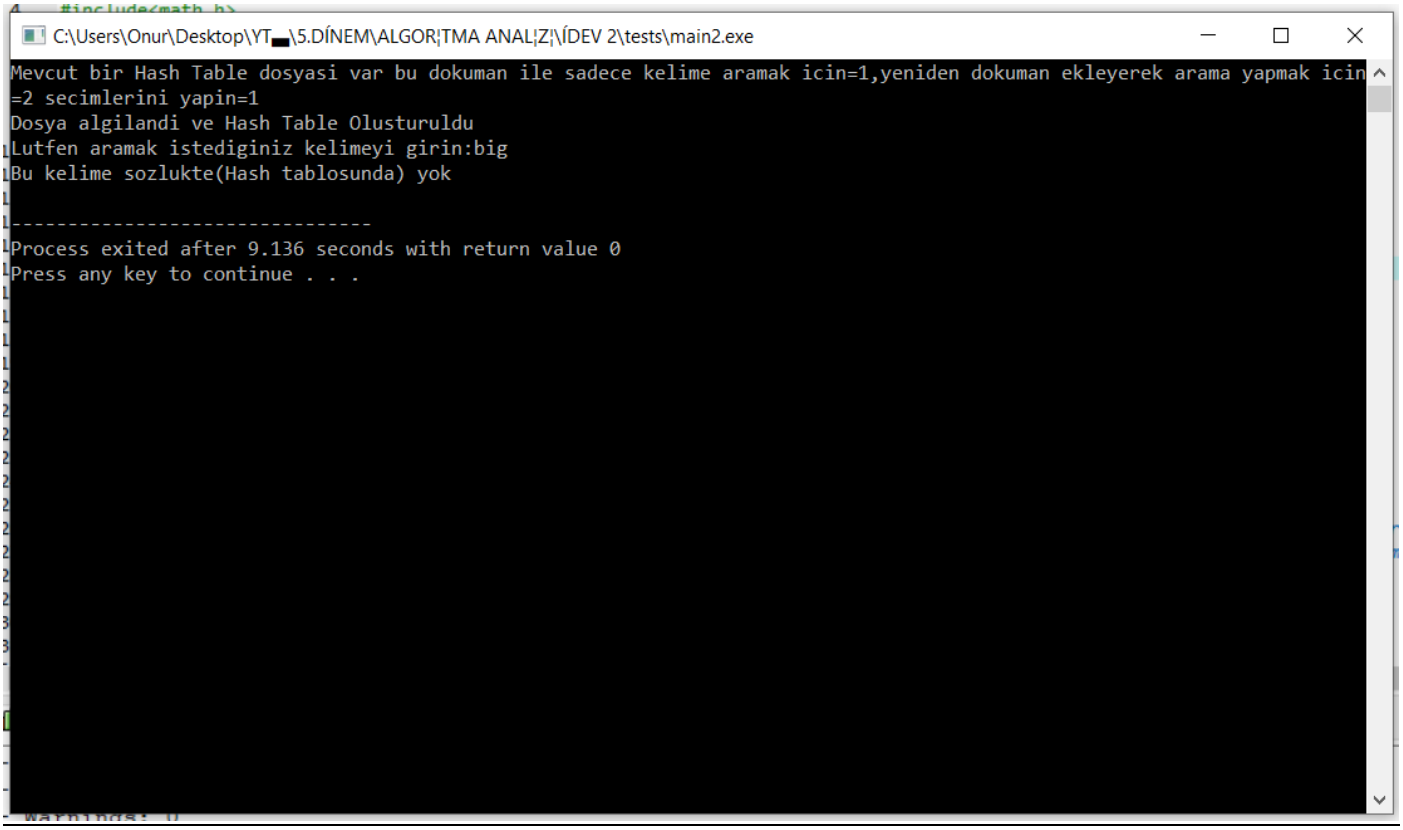


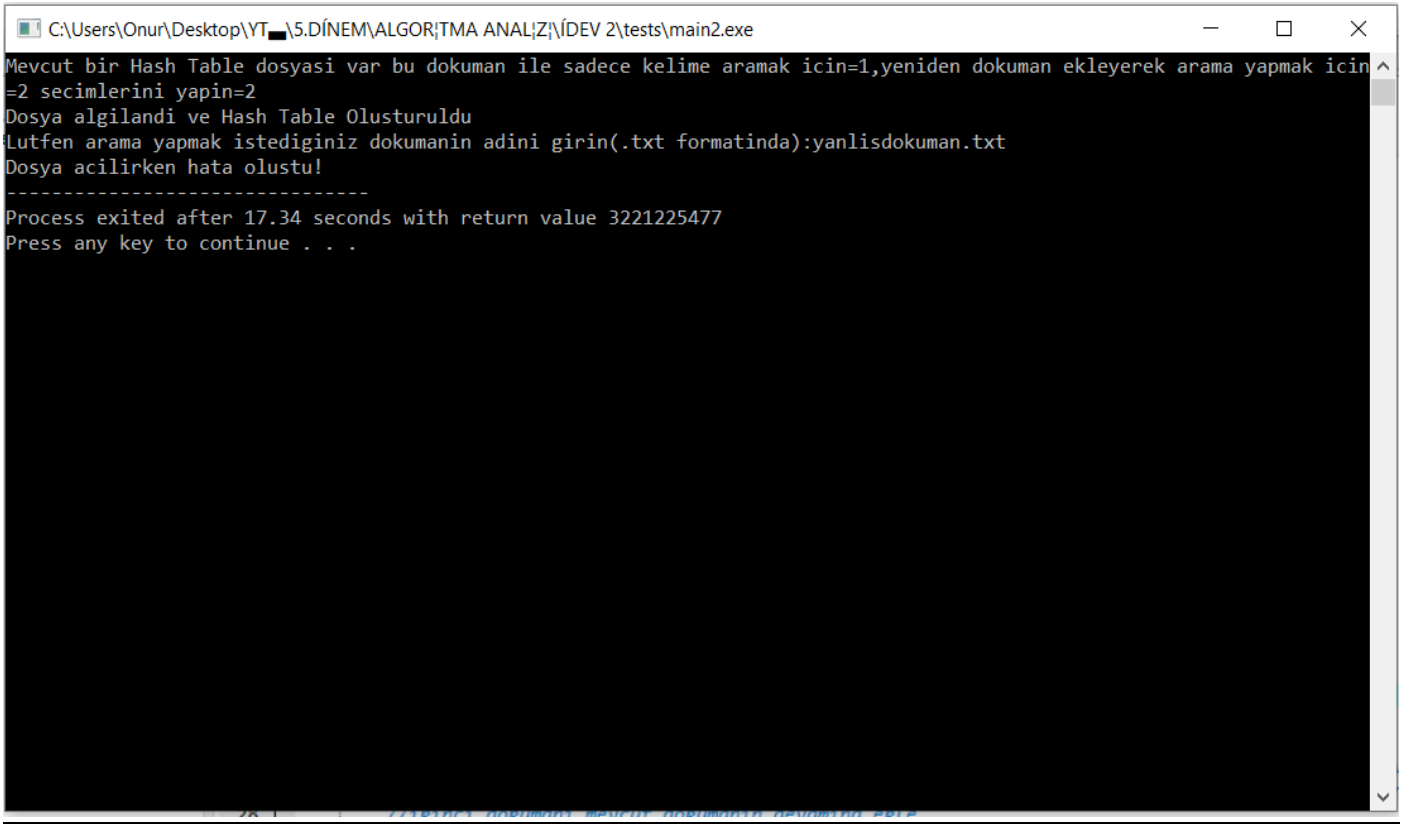

C:\Users\Onur\Desktop\YTÜ\5.DÖNEM\ALGORİTMA ANALİZİ\ÖDEV 2\tests\main2.c - [Executing] - Dev-C++ 5.11
C:\Users\Onur\Desktop\YT▂\5.DİNEM\ALGOR¦TMA ANAL¦Z¦\ÍDEV 2\tests\main2.exe
Mevcut bir Hash Table dosyasi var bu dokuman ile sadece kelime aramak icin=1,yeniden dokuman ekleyerek arama yapmak icin=2 secimlerini yapin=1
Dosya algilandi ve Hash Table Olusturuldu
Lutfen aramak istediginiz kelimeyi girin:long
Kelime:long:metin.txt,Dokuman: 1 Adimda bulundu
Process exited after 24.82 seconds with return value 48
Press any key to continue . . .
metin - Not Defteri
It is a long established fact that a reader will be distr
- Errors: 0
- Warnings: 0
- Output Filename: C:\Users\Onur\Desktop\YTÜ\5.DÖNEM
- Output Size: 158,1396484375 KiB
- Compilation Time: 0,31s


C:\Users\Onur\Desktop\YT▂\5.DİNEM\ALGOR¦TMA ANAL¦Z¦\ÍDEV 2\tests\main2.exe
Mevcut bir Hash Table dosyasi var bu dokuman ile sadece kelime aramak icin=1,yeniden dokuman ekleyerek arama yapmak icin=2 secimlerini yapin=2
Dosya algilandi ve Hash Table Olusturuldu
Lutfen arama yapmak istediginiz dokumanin adini girin(.txt formatinda):metin2.txt
Load factor=0.023069
Load factor=0.024072
Load factor=0.025075
Load factor=0.026078
Load factor=0.027081
Load factor=0.028084
Load factor=0.029087
Load factor=0.030090
Load factor=0.031093
Load factor=0.032096
Load factor=0.033099
Load factor=0.034102
Load factor=0.035105
Load factor=0.036108
Load factor=0.037111
Load factor=0.038114
Load factor=0.039117
Load factor=0.040120
Load factor=0.041123
Load factor=0.042126
Load factor=0.043129
Load factor=0.044132
Load factor=0.045135
Load factor=0.046138
Load factor=0.047141
Load factor=0.048144
17011701 - Not Defteri
1:a:metin2.txt
2998:Internet:metin2.txt
9:be:metin2.txt
20950:established:metin.txt:
21:as:metin2.txt
22:at:metin.txt:
24:If:metin2.txt
29:by:metin2.txt
32:in:metin2.txt
36:of:metin2.txt
37:is:metin2.txt
38:It:metin2.txt
40:All:metin2.txt
44:on:metin2.txt
metin2 - Not Defteri
There are many variations of passages of Lorem Ipsum avai
- Errors: 0
- Warnings: 0
- Output Filename: C:\Users\Onur\Desktop\YTÜ\5.DÖNEM\ALGORİTMA ANALİZİ\ÖDEV 2\tests\main2.exe
- Output Size: 158,1396484375 KiB
- Compilation Time: 0,31s


C:\Users\Onur\Desktop\YTÜ\5.DÖNEM\ALGORİTMA ANALİZİ\ÖDEV 2\tests\main2.c - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
C:\Users\Onur\Desktop\YT▄\5.DİNEM\ALGOR¦TMA ANAL¦Z¦\İDEV 2\tests\main2.exe
Load factor=0.119358
Load factor=0.120361
Load factor=0.121364
Load factor=0.122367
Load factor=0.123370
Load factor=0.124373
Load factor=0.125376
Load factor=0.126379
Load factor=0.127382
Load factor=0.128385
Load factor=0.129388
Load factor=0.130391
Load factor=0.131394
Load factor=0.132397
Load factor=0.133400
Load factor=0.134403
Load factor=0.135406
Load factor=0.136409
Load factor=0.137412
Load factor=0.138415
Load factor=0.139418
Load factor=0.140421
Load factor=0.141424
Lutfen aramak istediginiz kelimeyi girin:variations
Kelime:variations,Dokuman:metin2.txt 1 Adimda bulundu
Process exited after 29.17 seconds with return value 54
Press any key to continue . . .
17011701 - Not Defteri
Dosya Düzen Biçim Görünüm Yardım
1:a:metin2.txt
2998:Internet:metin2.txt
9:be:metin2.txt
20950:established:metin.txt:
21:as:metin2.txt
22:at:metin.txt:
24:If:metin2.txt
29:by:metin2.txt
32:in:metin2.txt
36:of:metin2.txt
37:is:metin2.txt
38:It:metin2.txt
40:All:metin2.txt
44:on:metin2.txt
metin2 - Not Defteri
Dosya Düzen Biçim Görünüm Yardım
There are many variations of passages of Lorem Ipsum avai
Compiler Resources Compile Log Debug Find Results Close
Abort Compilation
Shorten compiler paths

- Errors: 0
- Warnings: 0
- Output Filename: C:\Users\Onur\Desktop\YTÜ\5.DÖNEM\ALGORİTMA ANALİZİ\ÖDEV 2\tests\main2.exe
- Output Size: 158,1396484375 KiB
- Compilation Time: 0,31s
Line: 14 Col: 41 Sel: 0 Lines: 366 Length: 14340 Insert Done parsing in 0,078 seconds
St 1, Stn 1 100% Windows (CRLF) UTF-8


C:\Users\Onur\Desktop\YT▄\5.DİNEM\ALGOR¦TMA ANAL¦Z¦\İDEV 2\tests\main2.exe
Mevcut bir Hash Table dosyasi var bu dokuman ile sadece kelime aramak icin=1,yeniden dokuman ekleyerek arama yapmak icin
=2 secimlerini yapin=1
Dosya algilandi ve Hash Table Olusturuldu
Lutfen aramak istediginiz kelimeyi girin:big
Bu kelime sozlukte(Hash tablosunda) yok
Process exited after 9.136 seconds with return value 0
Press any key to continue . . .


C:\Users\Onur\Desktop\YT▄\5.DÍNEM\ALGOR¦TMA ANAL¦Z¦\ÍDEV 2\tests\main2.exe
Mevcut bir Hash Table dosyasi var bu dokuman ile sadece kelime aramak icin=1,yeniden dokuman ekleyerek arama yapmak icin
=2 secimlerini yapin=2
Dosya algilandi ve Hash Table Olusturuldu
Lutfen arama yapmak istediginiz dokumanin adini girin(.txt formatinda):yanlisdokuman.txt
Dosya acilirken hata olustu!

Process exited after 17.34 seconds with return value 3221225477
Press any key to continue . . .