



BLM3021
ALGORİTMA ANALİZİ

ÖDEV-3

DİNAMİK PROGRAMLAMA
PROBLEM-2

Problem2:

Sorgulanan bir cümlede yanlış yazılmış kelimeler varsa bu kelimelerin yerine doğru kelimeler öneren bir sistem tasarlanacaktır.

Örnek :

Kullanıcı: It is coold

Bilgisayar: “coold”is not in the dictionary. Did you mean:“cool “ or “cold”

Kullanıcı: cold

Bilgisayar: It is cold

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.

int insertWrong (hkv data) Fonksiyonu :

Hatalı kelimeler tablosuna ekleme işlemi yaptığımız fonksiyon.hkv structi parametre olarak gelir ve bu structimizde hatali kelime verilerini tutarız.Öncelikle key değeri oluşturup gerekli kontrolleri yaptıktan sonra hatali kelimemizi,hatali kelimeler tablosuna ekleriz.

int LevenshteinED(char kelime[30],char sKelime[30]) Fonksiyonu:

Parametre olarak verilmiş iki kelimenin mesafelerini Levenshtein Edit Distance metoduyla bulan fonksiyonumuzdur.Uzunlukları tutacak olan 2 değişkenimiz,indis değerlerimiz,matrisimiz ve bonus puan için iki string uzunluğu farkını tutan diff değişkenimiz vardır.Dönüş değeri,iki kelimenin mesafesidir ve bonus kısmı için oluşturulan kontrolde ise -1 döner.Böylelikle döndüğü zaman dönüş değerleri çerçevesinde kontroller yapılarak işlemlere devam eder(-1 döndüğü taktirde mesafe 2den büyüktür ve kelime önerilmez anlamına gelir).

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 yazdırıyoruz. Eğer eşleşen bir key değeri yok ise kelimenin hash tablosunda olmadığını kullanıcıya mesaj çıktısı vererek gösterir ayrıca bu işleminde kaç adımda tamamlandığı yazdırılır.

void searchWrong (char kelime[30],int key) Fonksiyonu :

Parametre olarak kelimenin kendisi ve key değeri gelir.Bu fonksiyona yalnızca search edilmiş ve sözlükte bulunamamış olan kelimeler gelir ve key değeri oluşturma işlemleri tekrar gerçekleşmemesi için key değeri hazır gelir.İlk kontrol olarak parametre olarak gelen horner key ile hash value(key) oluşturulur ve hatalı kelime tablosunda aramak için kontrolü yapılır.Tabloda var ise ekrana çıktı verilerek uyarılır yok ise Levenshtein Edit Distance metodu uygulanması için girilen hatalı kelime ve sözlükteki tüm kelimeler döngü içerisinde sırayla gönderilir.Önerilecek kelime olduğu taktirde kullanıcıya bilgilendirme yapılır ve kullanıcı bu önerilen kelimeler içinden yeni kelime girmesi gerekir.Kullanıcının girdiği yeni önerilen kelime hatalı kelimeler(hash) tablosuna kelimenin asıl hali ve keyiyle birlikte eklenir.

void searchSentence(char cumle[256]) Fonksiyonu:

Cümlelerin tam hali parametre olarak gelir ve cümleyi kelime kelime ayırma işlemi yapılarak cümle sonuna gelene kadar oluşan tüm kelimeler tek tek search fonksiyonuna gönderilerek aratılır.

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.

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

Maindeki akış ilk önce verilmiş olan sözlüğün okunma işlemidir.sözlük okunduğu vakit kelime kelime ayrılır ve horner metod ile keyleri oluşturulup insert fonksiyonuna yani hash tablosuna ekleme işlemi gerçekleştirilir.Bu işlemlerden sonra dosya kapatılır ve sözlükteki değerler ekrana printArray fonksiyonu ile bastırılır.Son blokta ise kullanıcıdan arattırmak istediği cümle girilmesi istenir ve cümledeki kelimeleri ayırıp tek tek aratan searchSentence fonksiyonuna kullanıcının girdiği cümleyi parametre olarak gönderip işleme sokar.

Levenshtein Edit Distance Ve Bonus Kısmı:

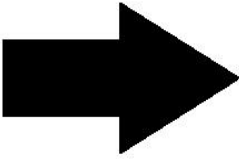
Levenshtein edit distance metodunu uygularken dikkat etmemiz gereken iki kontrol var.Öncelikle kelimeleri char olarak tek tek düşünerek mesafeler tablomuzu dolduruyoruz.Daha sonra bizi ilgilendiren kısımlar hep diagonaldeki mesafe değerleri olmalı.

Char char kontrol ettiğimizde karakterler birbiri ile aynı ise bir önceki diagonaldeki değer bir sonraki diagonaldeki değere yerleşir;

		o n		x	r
	o	n		x	r
o	0	1	2	3	4
n	1	0	1	2	3
u	2	1	0	1	2
r	3	2	1	1	2
	4	3	2	2	1

Eğer farklılık görüldüğü ilk adımda(yukardaki şekilden bir sonraki karakter kontrolünde)Aşağıdaki şekilde gösterildiği gibi değerler arası hangisi en küçük kontrolü yapıldıktan sonra en küçük olan değere +1 (cost) eklenir;

		o	n	x	r
	0	1	2	3	4
o	1	0	1	2	3
n	2	1	0	1	2
u	3	2	1		2
r	4	3	2	2	1



		o	n	x	r
	0	1	2	3	4
o	1	0	1	2	3
n	2	1	0	1	2
u	3	2	1	1	2
r	4	3	2	2	1

Bonus kısmı için sorulan sorunun cevabı ise kesin bir adım sayısı olmamakla beraber,diagonal kontrolünde ilk 3ün görüldüğü yerdir.İlk distance=3 görüldükten sonra bir sonraki diagonallerde kesinlikle 3den aşağı bir değer gelemez(mevcut char kontrolünden sonra bütün karakterler birbirine eşit olsa dahi distance değeri 3 olarak kalır ve daha altında düşmez);

1)strlen(onur)=n olsun,farklı kısımlar ile birlikte n+3 denilebilir.

		o	n	u	r	y	y	y
	0	1	2	3	4	5	6	7
o	1	0	1	2	3	4	5	6
n	2	1	0	1	2	3	4	5
u	3	2	1	0	1	2	3	4
r	4	3	2	1	0	1	2	3
x	5	4	3	2	1	1	2	3
x	6	5	4	3	2	2	2	3
x	7	6	5	4	3	3	3	3

2)strlen(onur)=n 'in daha uzun hali:Spesifik olarak net bi sayı veremeyiz fakat n+3 diyebiliriz.

		o	o	o	o	n	u	r	y	y	y	-	-	-
	0	1	2	3	4	5	6	7	8	9	10			
o	1	0	1	2	3	4	5	6	7	8	9			
o	2	1	0	1	2	3	4	5	6	7	8			
o	3	2	1	0	1	2	3	4	5	6	7			
o	4	3	2	1	0	1	2	3	4	5	6			
n	5	4	3	2	1	0	1	2	3	4	5			
u	6	5	4	3	2	1	0	1	2	3	4			
r	7	6	5	4	3	2	1	0	1	2	3			
x	8	7	6	5	4	3	2	1	1	2	3			
x	9	8	7	6	5	4	3	2	2	2	3			
x	10	9	8	7	6	5	4	3	3	3	3	3		
.														
.														
.														

Sonuç olarak bonus problemine verilecek cevap=**diagonelde ilk 3 geldiği adımdır.**

Ekran çıktıları:

```
Key:90941525 , Value:center
Key:19969060 , Value:using
Key:12971737 , Value:named
Key:12971752 , Value:names
Key:562969617 , Value:string
Key:1698950472 , Value:quadratic
Key:19980194 , Value:usual
Key:562975383 , Value:strong
Key:830977290 , Value:example
Key:1416970887 , Value:overview
Key:998189 , Value:about
Key:998205 , Value:above
Key:519990852 , Value:reason
Kelime adedi=338
=====BASLANGIC=====
(1)Lutfen aramak istediginiz cumleyi girin:It is coold
(*)Kelime sozlukte mevcut=Key:299,Kelime:it
=====
(*)Kelime sozlukte mevcut=Key:298,Kelime:is
=====
(*)coold kelimesi sozlukte(Hash tablosunda) yok
(*)coold kelimesi hatali kelime tablosunda(Hash tablosunda) yok
(*)coold kelimesi icin onerilen kelimeler:
good Mesafesi->2
hold Mesafesi->2
(*)Lutfen coold kelimesi icin onerilen yeni kelimelerden birini yazin:good
(*)Onerdiginiz kelime hatali kelime tablosuna eklendi!Key=3232219,Onerilen kelime=good,asil kelime=coold
=====
=====BASLANGIC=====
(1)Lutfen aramak istediginiz cumleyi girin:
```

```
Key:90941525 , Value:center
Key:19969060 , Value:using
Key:12971737 , Value:named
Key:12971752 , Value:names
Key:562969617 , Value:string
Key:1698950472 , Value:quadratic
Key:19980194 , Value:usual
Key:562975383 , Value:strong
Key:830977290 , Value:example
Key:1416970887 , Value:overview
Key:998189 , Value:about
Key:998205 , Value:above
Key:519990852 , Value:reason
Kelime adedi=338
=====BASLANGIC=====
(1)Lutfen aramak istediginiz cumleyi girin:YILDIZ TEKNİK UNIVERSİTESİ
(*)yildiz kelimesi sozlukte(Hash tablosunda) yok
(*)yildiz kelimesi hatali kelime tablosunda(Hash tablosunda) yok
(*)Kelimenin sozlukteki tum kelimeler ile mesafesi 2den buyuk.Onerilecek kelime bulunamadi.
=====
(*)teknik kelimesi sozlukte(Hash tablosunda) yok
(*)teknik kelimesi hatali kelime tablosunda(Hash tablosunda) yok
(*)Kelimenin sozlukteki tum kelimeler ile mesafesi 2den buyuk.Onerilecek kelime bulunamadi.
=====
(*)universitesi kelimesi sozlukte(Hash tablosunda) yok
(*)universitesi kelimesi hatali kelime tablosunda(Hash tablosunda) yok
(*)Kelimenin sozlukteki tum kelimeler ile mesafesi 2den buyuk.Onerilecek kelime bulunamadi.
=====
=====BASLANGIC=====
(1)Lutfen aramak istediginiz cumleyi girin:
```

```
Key:830977290 , Value:example
Key:1416970887 , Value:overview
Key:998189 , Value:about
Key:998205 , Value:above
Key:519990852 , Value:reason
Kelime adedi=338
=====BASLANGIC=====
(1)Lutfen aramak istediginiz cumleyi girin:its is very long day
(*)its kelimesi sozlukte(Hash tablosunda) yok
(*)its kelimesi hatali kelime tablosunda(Hash tablosunda) yok
(*)its kelimesi icin onerilen kelimeler:
is Mesafesi->1
it Mesafesi->1
(*)Lutfen its kelimesi icin onerilen yeni kelimelerden birini yazin:it
(*)Onerdiginiz kelime hatali kelime tablosuna eklendi!Key=9288,Onerilen kelime=it,asil kelime=its
=====
(*)Kelime sozlukte mevcut=Key:298,Kelime:is
=====
(*)very kelimesi sozlukte(Hash tablosunda) yok
(*)very kelimesi hatali kelime tablosunda(Hash tablosunda) yok
(*)very kelimesi icin onerilen kelimeler:
try Mesafesi->2
wiry Mesafesi->2
(*)Lutfen very kelimesi icin onerilen yeni kelimelerden birini yazin:wiry
(*)Onerdiginiz kelime hatali kelime tablosuna eklendi!Key=660790,Onerilen kelime=wiry,asil kelime=very
=====
(*)Kelime sozlukte mevcut=Key:372348,Kelime:long
=====
(*)day kelimesi sozlukte(Hash tablosunda) yok
(*)day kelimesi hatali kelime tablosunda(Hash tablosunda) yok
(*)day kelimesi icin onerilen kelimeler:
may Mesafesi->1
way Mesafesi->1
(*)Lutfen day kelimesi icin onerilen yeni kelimelerden birini yazin:way
(*)Onerdiginiz kelime hatali kelime tablosuna eklendi!Key=3900,Onerilen kelime=way,asil kelime=day
=====
=====BASLANGIC=====
(1)Lutfen aramak istediginiz cumleyi girin:
```

```
Key:19969060 , Value:using
Key:12971737 , Value:named
Key:12971752 , Value:names
Key:562969617 , Value:string
Key:1698950472 , Value:quadratic
Key:19980194 , Value:usual
Key:562975383 , Value:strong
Key:830977290 , Value:example
Key:1416970887 , Value:overview
Key:998189 , Value:about
Key:998205 , Value:above
Key:519990852 , Value:reason
Kelime adedi=338
=====BASLANGIC=====
(1)Lutfen aramak istediginiz cumleyi girin:Onur KOCKAN
(*)onur kelimesi sozlukte(Hash tablosunda) yok
(*)onur kelimesi hatali kelime tablosunda(Hash tablosunda) yok
(*)onur kelimesi icin onerilen kelimeler:
on Mesafesi->2
or Mesafesi->2
one Mesafesi->2
four Mesafesi->2
hour Mesafesi->2
once Mesafesi->2
only Mesafesi->2
your Mesafesi->2
(*)Lutfen onur kelimesi icin onerilen yeni kelimelerden birini yazin:önerilmeyen_kelime
(*)Lutfen yalnızca size onerilen kelimelerden birini girin.
=====
(*)Lutfen onur kelimesi icin onerilen yeni kelimelerden birini yazin:
```

```

Key:1416970887 , Value:overview
Key:998189 , Value:about
Key:998205 , Value:above
Key:519990852 , Value:reason
Kelime adedi=338
=====BASLANGIC=====
(1)Lutfen aramak istediginiz cumleyi girin:Onur KOCKAN
(*)onur kelimesi sozlukte(Hash tablosunda) yok
(*)onur kelimesi hatali kelime tablosunda(Hash tablosunda) yok
(*)onur kelimesi icin onerilen kelimeler:
on Mesafesi->2
or Mesafesi->2
one Mesafesi->2
four Mesafesi->2
hour Mesafesi->2
once Mesafesi->2
only Mesafesi->2
your Mesafesi->2
(*)Lutfen onur kelimesi icin onerilen yeni kelimelerden birini yazin:önerilmeyen_kelime
(*)Lutfen yalnızca size onerilen kelimelerden birini girin.
=====
(*)Lutfen onur kelimesi icin onerilen yeni kelimelerden birini yazin:only
(*)Onerdiginiz kelime hatali kelime tablosuna eklendi!Key=460988,Onerilen kelime=only,asil kelime=onur
=====
(*)kockan kelimesi sozlukte(Hash tablosunda) yok
(*)kockan kelimesi hatali kelime tablosunda(Hash tablosunda) yok
(*)Kelimenin sozlukteki tum kelimeler ile mesafesi 2den büyük.Onerilecek kelime bulunamadi.
=====
=====BASLANGIC=====
(1)Lutfen aramak istediginiz cumleyi girin:

```

Kodlar:

```

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>
#define M 999983

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

typedef struct hataliKelimeVerileri{// tum hatali kelimelerimiz icin olusturdugumuz
struct yapimiz
    int key;
    char value[30]; //asil (sozlukte bulunmayan)kelime
    char rValue[30]; //onerilen kelime
}hkv;

typedef struct onerilenKelimeler{//kelime onerimi esnasinda onerilen kelimeleri tutmak
icin olusturdugumuz structimiz
    char kelime[30];    //kelimenin kendisi
    int key;    //unique degeri
    int distance;    //karsilastirildigi kelime ile olan uzakligi.
}recKelime;

recKelime aRec[M]; //onerilecek kelimelerin tutuldugu dizi.
kv a[M];    //kelimeleri tutacak M(1000 sayisina en yakn asal say=997) boyutlu struct
dizimiz.
hkv aNot[M]; //arama sonucu sozlukte olmayan kelimelerin tutuldugu dizi.

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.

```



```

    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== -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);
    }
    else{//eklenmemis ise
        while(a[keyGetir].key!= -1 && a[keyGetir].key!=data.key && i<M){//bos yer bulana
            kadar ve gelen deger mevcut indisdeki degere esit degilse 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);
        }
    }

}

int insertWrong (hkv data){ //hatali kelime tablosuna ekleme islemi.
    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(aNot[keyGetir].key== -1){//eger bu data hic eklenmemis ise
        //tum degiskenleri ile olusan hash valuedeki indise ekle
        aNot[keyGetir].key=data.key;
        strcpy(aNot[keyGetir].value,data.value);
        strcpy(aNot[keyGetir].rValue,data.rValue);
        printf("(*)Onerdiginiz kelime hatali kelime tablosuna eklendi!Key=%d,Onerilen
kelime=%s,asil
kelime=%s\n=====\\n",aNot[keyGetir].key,aNot[keyGetir].rValue,aNot[keyGetir
].value);
    }
    else{//eklenmemis ise
        while(aNot[keyGetir].key!= -1 && aNot[keyGetir].key!=data.key && i<M){//bos yer
            bulana kadar ve gelen deger mevcut indisdeki degere esit degilse iteratif bir sekilde
            ilerle ve
                i++;
                keyGetir = h(data.key,i);//en son olusturdugun keyi ve indisi tut
            }
            //bu indise eleman ekle.
            aNot[keyGetir].key=data.key;
            strcpy(aNot[keyGetir].value,data.value);
            strcpy(aNot[keyGetir].rValue,data.rValue);
            printf("(*)Onerdiginiz kelime hatali kelime tablosuna eklendi!\\nKey=%d,Onerilen
kelime=%s,asil
kelime=%s\n=====\\n",aNot[keyGetir].key,aNot[keyGetir].rValue,aNot[keyGetir
].value);
        }
    }

}

int LevenshteinED(char kelime[30],char sKelime[30]){//Levenshtein Edit Distance metodu
ile iki kelimenin birbirine olan uzakligini bulan fonksiyon.
    int len1=strlen(kelime);//1.kelimenin uzunlugu
    int len2=strlen(sKelime);//2.(sozlukten sirayla gelen)kelimenin uzunlugu
    int matrix[len1 + 1][len2 + 1];//[0,0]bo olan indis dahil uzunluklarkin 1 fazlasi
ile olusturdugumuz tablo icin tanimladigimiz matrisimiz.
    int diff = len1-len2;//bonus kismi icin farki tutmas beklenen degiskenimiz.
    int i,j;//indisleri tutacak olan degiskenlerimiz.
    for (i = 0; i <= len1; i++) { //matrisin 0.sutunu dolduruluyor.
        matrix[i][0] = i;
    }
}

```

```

for (i = 0; i <= len2; i++) { //matrisin 0.satiri dolduruluyor.
    matrix[0][i] = i;
}
for (i = 1; i <= len1; i++) { //matrisimizde hucreler geziliyor.
    char c1,c2;

    c1 = kelime[i-1]; //kontrol icin ilk kelimedeki karakterler sirayla atilacak.
    for (j = 1; j <= len2; j++) {

        c2 = sKelime[j-1]; //kontrol icin ikinci kelimedeki karakterler sirayla
        atilacak.

        if (c1 == c2) { //karakterler birbirine esit ise diagonelin bi ustunu bozma
            matrix[i][j] = matrix[i-1][j-1];
        }
        else { //esit degil ise
            int delete;
            int insert;
            int substitute;
            int minimum;

            delete = matrix[i-1][j] + 1;
            insert = matrix[i][j-1] + 1;
            substitute = matrix[i-1][j-1] + 1;
            minimum = delete;
            //minimumlarn bulup +1 ekleyip ilgili matrix[i][j] indisine yazyoruz
            if (insert < minimum) {
                minimum = insert;
            }
            if (substitute < minimum) {
                minimum = substitute;
            }
            matrix[i][j] = minimum;
            //BONUS kisim icin olan kontrol.
            if(i == (j + diff) && matrix[i][j] > 2) { //burada onemli olan her zaman
                matrix[j][j+diff] hucrelerine bakiliyor olamasidir.eger bu hucrelerdeki degerlerden biri
                2 den buyuk olursa
                    //kelimelerin uzakligi diagonelde 2 veya 2den buyuk gelir,daha
                    dusuk mesafe gelemez.Boylelikle mesafeler kesin 2 den buyuk demektir
                    //eger bu sart saglanirsa uzaklik 2den buyuk demektir -1 doner
                    return -1;
            }
        }
    }
}

return matrix[len1][len2]; //son satir ve son sutundaki bulunan deger mesafe
degeridir.Bu deger dondurulur.

}

void searchWrong(char kelime[30],int key){ //hatali kelime tablosunda arama yapan
fonksiyonumuz.
    if (key != 0) { //kelimenin sayi karsiligi olusturulmustur.Arama islemine
gecilebilir.
        int i=0;
        int keyGetir = h(key,i); //search yapmamiz icin keyGetir isimli degiskeni tekrar
olusturuyoruz.
        if(aNot[keyGetir].key==key) { //olusan keydeki hash valuemiz parametre olarak
gelen key degerine esit ise tek adimda bulunmustur.Direkt ekrana bastir.
            printf("(*)Kelime hatali kelime tablosunda mevcut.Key:%d,Kelime:%s\n(*)Bu
kelime icin onerilmis
kelime=%s\n=====\\n",aNot[keyGetir].key,aNot[keyGetir].value,aNot[keyGetir]
.rValue);
        }
        else { //kelime hatali kelime tablosunda yok ise;
            printf("(*)%s kelimesi hatali kelime tablosunda(Hash tablosunda)
yok\\n",kelime);

```

```

//Levensthein Edit Distance uygulanir.
int z,checkOne=0;//z degiskeni dongu icin,checkOne degiskeni ise mesafesi 1
olan kelimelerin var olup olmadigini tutmaya yariyan degiskenimiz.
int dCounter=0;//onerilen kelime sayisini kontrol icin tutan degiskenimiz.
for(z=0;z<M;z++){//tum diziyi gez
    if(a[z].key!=-1){//eleman olan indislere bak
        int dd=LevenshteinED(kelime,a[z].value);//mesafesini al.
        if(dd<=2 && dd!=-1){//mesafesi 2 veya 2den kucuk ise
            //onerilen kelimeler structina at.
            strcpy(aRec[dCounter].kelime,a[z].value);
            aRec[dCounter].key=1;
            aRec[dCounter].distance=dd;
            dCounter++;//onerilen kelime sayilarini tutan degiskenimiz
arttirilir.
                if(dd==1)//mesafesi 1 olanlar mevcut ise.Switch degiskenimizi
1liyoruz ve Boylelikle mesafesi 2 olanlar kullaniciya onerilmeyecek.
                    checkOne=1;
            }
        }
    }
    if(dCounter==0)//onerilecek kelime hic bulunamadi.Cunku onerilen kelime
sayisi=0
        printf("(*)Kelimenin sozlukteki tum kelimeler ile mesafesi 2den
buyuk.Onerilecek kelime bulunamadi.\n=====\\n");
    else{//onerilecek kelimeler onerildi ve kullanici isleme devam edicek.
        printf("(*)%s kelimesi icin onerilen kelimeler:\\n",kelime,a[z].value);
        int k;//indis gorevi gormesi beklenen dongu degiskenimiz.
        for(k=0;k<dCounter;k++){//onerilen kelime sayisi kadar git.
            if(aRec[k].key==1 && aRec[k].distance==1){//once distance i 1 olan
onerilen kelimeleri bastir.
                printf("%s Mesafesi->%d\\n",aRec[k].kelime,aRec[k].distance);
            }
        }
        for(k=0;k<dCounter;k++){//distance i 1 olan kelime yok ise,distance i 2
olan kelimeleri bastir.
            if(aRec[k].key==1 && aRec[k].distance==2 && checkOne==0){//distance
i 1 olanlar yok sadece 2 olan onerilen kelimeler.
                printf("%s Mesafesi->%d\\n",aRec[k].kelime,aRec[k].distance);
            }
        }
    }
    char newKelime[30];//kullanicinin onericegi yeni kelimeyi tutacak olan
degiskenimiz.
    int chck=0;//kullanicinin onerilen kelimeler disinda farkli kelime
girmesini engelliycek olan degiskenimiz while icinde kullanılacaktır.
    while(chck==0){
        printf("(*)Lutfen %s kelimesi icin onerilen yeni kelimelerden
birini yazin:",kelime);
        gets(newKelime);

        for(k=0;k<dCounter;k++){//sadece onerilen kelime sayisi kadar
ilerle.
            if(strcmp(aRec[k].kelime,newKelime)==0)//kullanicinin girdigi
kelime,onerilen kelimeler arasindami?Mevcut indisdeki kelime ile kiyasla.
                chck=1;//evet ise switchi 1le
            }
        if(chck==1){//kullanici onerilen kelimeler icinden bir kelime
onerdiyse
            //hatali kelime verileri structi(hatali kelime tablosu icin
olusturdugumuz struct)olusturulup;
            hkv data;
            strcpy(data.value,kelime);
            strcpy(data.rValue,newKelime);
            data.key=key;
            insertWrong(data);//hatali kelime tablosuna ekleme islemi yapan
fonksiyonumuz gonderilir.
        }
        else

```

```

        printf("(*)Lutfen yalnızca size önerilen kelimelerden birini
        girin.\n=====\\n");
    }

}

}

}

void search(char kelime[30]){//kelimeyi sozlukte ariyan fonksiyonumuz
//HORNER METOD ILE KELIMENIN SAYIYA CEVRILME ISLEMI:
int length = strlen(kelime);
int power = length-1;
unsigned int key = 0;
int i=0,j,k=0;
char gecici;
//kelimeyi kucuk harflere cevirme.
int fark = 'a' - 'A';//ASCII table a gore once kucuk harfler sonra buyuk harfler
oldugu icin fark 32 olur
while (kelime[k] != '\\0') //{NULL deger olmadikca devam et
    if (kelime[k] >= 'A' && kelime[k] <= 'Z') //{harf buyuk ise kucult.
        kelime[k] += fark;//fark kadar eklersek bu harfin ASCII degeri buyur ve ayn
        harfin kucugune sahip olmus olur
    }
    k++;
}
//kelimeyi kucultme islemi sonu.
//HORNET METODUN ISLEM KISMI
for(j=0;j<length;j++){
    gecici = kelime[j];
    if (gecici >= 'a')
        gecici = 'A' - 'a' + gecici;

    key += (pow(31,power)*(gecici-'A'+1));//us olarak 31(Asal sayi)verildi ve
    formül uygulandı.
    power--;
}
key=abs(key);//tasmadan dolayi olusabilecek eksili key degerlerini onlemek adina
mutlak degerini alip oyle key olusturuyoruz.
if (key != 0){//kelimenin sayi karsiligi olusturulmustur.Arama islemine
gecilebilir.
    int keyGetir = h(key,i);//search yapmamiz icin keyGetir isimli degiskeni
    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 sozlukte
mevcut=Key:%d,Kelime:%s\\n=====\\n",a[keyGetir].key,a[keyGetir].value);
    }
    else{//kelime sozlukte yok ise
        printf("(*)%s kelimesi sozlukte(Hash tablosunda) yok\\n",kelime);
        searchWrong(kelime,key);//bir diger asama olan hatali kelime tablosunda
        arama asamasina gecilir.
    }

}

}

void searchSentence(char cumle[256]){//cumlenin tamamini alip kelime kelime ayirarak
aratma fonksiyonuna gonderen fonksiyonumuz.
    int i=0,k=0,j;//k degiskeni kelime sonuna kadar dongunun gidilmesi icin kullanılan
    counter
    int kelimeSonu=strlen(cumle);//kelimenin sonunu tutmamiz icin olusturdugumuz
    degisken.Parametre olarak verilen cumlenin boyutunu tutar.
    char temp;//karakter kontrolu icin temp degiskenimiz
    char temp2[256];//cumle kontrolu icin temp2 degiskenimizi tanımladık.
    if(kelimeSonu==1)//tek harfli ise direkt aramaya sok.

```

```

        search(cumle);
    else{
        while(k<=kelimeSonu){//cumle sonuna kadar git
            temp = cumle[k];//harfi kontrol etmek icin al
            if(temp == ' ' || k == kelimeSonu){//bosluk veya satir sonuna gelindi ise
eger kelime olusturur.
                search(temp2);//olusan kelimeyi aramaya gonder
                for(j=0;j<30;j++){//kelime tutan temp2 degiskenini temizle
                    temp2[j]='\0';
                    i=0;
                }
                else{
                    temp2[i] = temp;
                    i++;
                }
                k++;
            }
        }
    }

}

void printArray(){//sozlukteki tum elemanlari ekrana bastiran fonksiyonumuz
    int i=0,j=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 \n",a[i].key,a[i].value);
            j++;
        }
    }
    printf("Kelime adedi=%d\n",j);
}

void resetHashTable(){//hash tablolarin sifirlanmasi icin olusturulan fonksiyon.Hem
sozluk hemde hatali kelime tablosu temizlenir.
    int i,j;
    for(i=0;i<M;i++){//dizi boyutu kadar ilerle
        a[i].key=-1;
    }
    for(j=0;j<M;j++){
        aNot[j].key=-1;
    }
}

int main(int argc, char *argv[]) {
    FILE* fp;
    fp = fopen("smallDictionary.txt","r");//kontrol icin bu dosya aciliyor
    if(fp == NULL){//dosya acilirken bir hata olusur ise
        printf("Dosya acilirken hata olustu!");//ekrana uyarı bastir.
        fclose(fp);
    }
    else{//sorunsuz bir sekilde program acildi ise ana akisa gec;
        resetHashTable();//Hash tableimiz temizleniyor.

        //dosyadaki kelimeler tek tek okunur ve hash tablosuna aktarilir.
        int j=0,i=0;//indis degiskenlerimiz
        kv data;//kelime verileri olarak olusturdugumuz struct adina tanımladigimiz
degiskenimiz.
        unsigned int key=0;//horner metodu icin hesapladigimiz keyleri tutacak olan
degiskenimiz.
        char temp[50];//string kontrolu icin olusturdugumuz temp degiskenimiz.
        char temp2[50];//stringimizde olusan 1 kelimeyi kontrol etmek icin
olusturdugumuz temp degiskenimiz.
        while(!feof(fp)){//dosya sonuna kadar git
            temp[i] = fgetc(fp);//okudugun karakteri al.
            if(temp[i] == ' ' || temp[i] == '\n'){//bosluk veya satir sonuna gelindi
ise eger kelime olusturur
                int length = strlen(temp2);
                //HORNET METODUN uygulanmasi.

```

```

        int power = length-1;
        key = 0;
        data.key=0;
        char temp3;
        for(j=0;j<length;j++){//horner metod ile kelimenin sayiya
donusturulmesi
            temp3 = temp2[j];
            if (temp3 >= 'a')
                temp3 = 'A' - 'a' + temp3;

            key += (pow(31,power)*(temp3-'A'+1));
            power--;
        }
        key=abs(key);//olusacak tasmalardan oturu negatif bir key degeri
        gelmemesi adina keyimizin mutlak degerini aliyoruz.
        if (key != 0){//kelimemiz insert edilmeye hazır ve keyiyle birlikte
        elde edilmistir.Sozluk hash tablosuna insert islemi icin gonderilir.
            data.key=key;
            strcpy(data.value,temp2);
            insert(data);

        }
        for(j=0;j<50;j++){//kelimeleri olusturduktan sonra temp gorevine devam
        etmesi icin temp2 degiskenimizi sifirliyoruz.
            temp2[j] = '\0';
            data.value[j]='\0';
        }
        i=0;

    }
    else{//yukardaki sart saglanmiyorsa kelime okunmaya devam ediyordur.
        temp2[i] = temp[i];
        i++;
    }
}
fclose(fp);//dosyayi kapat.
printArray();//ve sozlugu yazdir

while(1){
    char cumle[256];
    printf("=====BASLANGIC=====\\n(1) Lutfen aramak
istediginiz cumleyi girin:");
    gets(cumle);
    searchSentence(cumle);//kullaniciyan girilen cumleyi,cumle arama fonksiyonuna
gonder.
}

}

return 0;
}

```