Algoritma Analizi – 2. Ödev

Elif Yağmur Duran – 18011071

Başlamadan önce çalıştıramadığımı belirtmek istiyorum. Çalışmayan kısımların çoğunluğu file işlemlerinden kaynaklı. Yazdığım fonksiyonları elle doldurdum bir hashtable dizisinde sorunsuz çalıştırdım ama filedan okuma yazma işlemlerini beceremediğim için çalışmıyor.

# Main öncesi tanımlamalar

//gerekli kütüphaneler

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <math.h>

//makro tanımları

#define R 31 //horner metodunda kullanılmak üzeren küçük bir asal sayı

#define M 997 //tablo boyutu

#define MM 996 //dokümanda verilen M-1

//global tanımlamalar

struct ItemStruct {

int key; //stringler horners fonksiyonundan geçirildikten sonra elde edilen key

char\* docs; //doküman isimlerinin stringi, dinamik olması için char pointerı olarak tanımlandı

}; //hashtableın bir satırını saklayan struct

typedef struct ItemStruct item; //ismini kısaca item yaptık

item\* table[M]; //itemler dizisi

int horners(char str[]); //hash fonksiyonu sayı almak zorunda, stringleri sayıya çeviren fonksiyon

int h1(int key); //hashdouble fonksiyonu içinde geçen hashler

int h2(int key);

int hashDouble(int key, int i); //clustering problemini çözmek için double hashing

item\* find(char str[]); //verilen string i hashtable da arayan fonksiyon

void insert(char str[], char docname[]); //table a yeni bir kelime ekleyen ya da varolan kelimenin dokümanlarının ismini güncelleyen fonksiyon – ama çalıştıramadığım için burada değil.

# Main fonksiyonu

## Değişkenler

int main(){

int wordcount = 0;

int load\_factor = 0; //kelime sayısı/M den doluluk oranını tespit eder

int selection; //yeni doküman eklemek için 1, kelime aramak için 2, çıkmak için 0.

FILE \*fhash; //hash file a pointer

FILE \*f1;//diğer file lar için ekstra pointer

char filename[50];//file adını tutan string

char tmptxt[20];//diğer şeyler için temporary string

item\* tmp;//temporary structure

fhash = fopen("hashfile.txt", "w");//hashfile ı yazma modunda açılır

printf("to add a new document, select 1. to search for a word, select 2. if you want to exit the program, select 0.\t");

scanf("%d", &selection);

while( selection != 0 ){ //selection 0 olmadıkça program çalışmaya devam edebilsin diye döngü burdan sonra whie ın içinde if(selection == 0) veya if(selection == 1) inceliyoruz, ikisi de yoksa yeni bir selection yapılması isteniyor

## Selection 1

if(selection == 1){/\*scan a new doc\*/

printf("\nenter name of document:");//dokumanın adını alıp açma

scanf("%s", filename);

f1 = fopen(filename, "w");

if( f1 == NULL ){

printf("file failed to open.") ; //açıldı mı kontrolü

}else{

while( f1 != NULL){//açıldıysa pointer null değilken fsanf le her kelime tek tek alınıp insert edilmeli ama insert çalışmıyor

fscanf(f1,"%s",tmptxt);

insert(tmptxt, filename);

wordcount++; //burda da loadfactor kontrolleri

load\_factor = wordcount/M;

if( load\_factor >= 0.8){

printf("\nwarning. hash table is filled too much.");

}

if( load\_factor >= 1){

printf("\nhash table full. exiting program.");

selection = 0;

}

}

}

## Selection 2

}else if(selection == 2){/\*search for a word\*/

printf("\nenter word:\t");

scanf("%s", tmptxt);

if(find(tmptxt) == NULL){//find fonksiyonu null döndürdüyse yok deriz

printf("\ncouldn't find word.");

}else{varsa fonksiyonun döndürdüğü itemi tmp struct a koyup dokuman isimlerini yazdırırız.

tmp = find(tmptxt);

printf("\ndocs that include given word:%s\t", tmp->docs);

}

}else{//buradan sonrası yanlış seçim durumu ve main in sonu

printf("\nwrong selection. please enter 1 to add a new document, or 2, to search for a word.\t");

}

printf("\nto add a new document, select 1. to search for a word, select 2. if you want to exit the program, select 0.\t");

scanf("%d", &selection);

}

printf("\nexiting program. have a nice day.");

return 0;

}

# Diğer Fonksiyonlar

int horners(char str[]){//string i sayıya çevirme formülünü uygulayan fonksiyon

int key = 0;

int i;

int n = strlen(str);

for( i=0 ; i<n ; i++){

key =+ ( pow(R, n-i-1)\*str[i] );

}

return key;

}

int h1(int key){

return key % M;

}

int h2(int key){

return 1 + (key % MM);

}

int hashDouble(int key, int i){

return ( h1(key) + i\*h2(key) ) % M; //h1 ve h2 fonksiyonlarıyla birlikte hashdouble fonksiyonu, aldığı key hornersten string in geçirilmesiyle elde edilir

}

item\* find(char str[]){

int key = horners(str); //string hornersten geçirilip sayıya çevrilir

int i = 1;

int j = hashDouble(key, i); //elde edilen key double hashing için uygun hale getirilir

while (i <= M && table[j] != NULL){i tablodan çıkmadığı sürece ve boş bi hücreye rastlamadığımız sürece sıradaki hücre adığımız hücreye mi bağlı bakarız

if (table[j]->key == key){

return table[j];//bulunduysa o hücre döndürülür

}

i++; //bulunamadıysa i artırılıp double hashing formülü yeniden hesaplanır

j = hashDouble(key, i);

}

return NULL;//bulunamadıysa null döndürülür

}

void insert(char str[], char docname[]) {

}