DATA STRUCTURES AND ALGORITHMS

SUBMITTED BY – ABHINAV GEMAWAT (18335214)

1. For the first part of this assignment you should write a hash table to store the frequency of names in an (unordered) list of Irish surnames. It will therefore use char arrays as keys and store ints as values. Use the hash function hash1 that adds the integer value of the chars in the string:

You should use a linear probing strategy to handle collisions. For simplicity, you do not need to worry about a dynamically growing hash table. Allocate a fixed amount of memory for your keys and values at the start of the program and focus on implementing hashing and linear probing. Test your hash table by loading the test data provided in names.csv. Write a program that allows you to test the frequency of a given string. NB: Load the data using a cvs parser (yours from assignment 0 or the one in the solution on blackboard) (if that doesn’t work for you, keep making progress by hardcoding the data in your code!). Update your code to count the number of collisions obtained and display it.

ANSWER :

#include <stdio.h>

#include <stdlib.h>

struct record{

    int freq;

    char name[];

}arr[100];

int hash(char\* s)

{

    int hash = 0;

    while(\*s){

        hash = hash + \*s;

        s++;

    }

    return hash;

}

int index\_name(int key,int probe){

    int index;

    index=key%53+probe;

}

int main(){

    FILE \* fp;

    char buff[100];

    fp = fopen("names.txt","r");

    int asc;

    int flag=0;

    int k=0;

    int index=0;

    int count=0;

    if (fp == NULL)

    {

        perror("Error opening file\n");

    }

    else

        while(fgets(buff,sizeof(buff),fp))

        {

          asc=hash(buff);

          k=0;

          flag=0;

          while(flag==0)

          {

              index=index\_name(asc,k);

              if(strcmp(arr[index].name,buff)==0){

                  arr[index].freq++;

                  flag=1;

              }

              else if(strlen(arr[index].name)>1){

                  k++;

                  count++;

              }

              else {

                  strcpy(arr[index].name,buff);

                  arr[index].freq=1;

                  flag=1;

              }

        }

char s[100];

k = 0;

char z[100];

strcpy(z,"\n");

printf("enter string\n");

gets(s);

strcat(s,z);

int count1=0;

a=hash(s);

flag=0;

while(flag==0){

index=find\_index(a,k);

if(strcmp(s1[index].name,s)==0){

printf("frequency=%d\nnumber of collisions=%d\nindex=%d\nvalue at index=%s\n",s1[index].freq,count,index,s1[index].name);

flag=1;

}

else if(strlen(s1[index].name)>1){

k++;

count1++;

}

else{

printf("not found");

break;

}

}

}

1. Now find a better hash function hash2 for the data considered. Feel free to consult online resources. Note, the sample data is only a sample! Do not overfit your function to the sample provided, it should work well with any lists of Irish surnames. Justify your choice of hash function in your report (half a page max). Test your function on the sample data, indicate in your report how many collisions occur, is it better than the result from task 1?

ANSWER :

#include <stdio.h>

#include <stdlib.h>

struct record{

    int freq;

    char name[];

}arr[100];

int hash(char\* s){

double hash = 0.00;

int m=26;

char sub='A';

double power=100;

while(\*s){

hash = hash + (\*s)\*power;

s++;

power=power/26;

sub='a';

}

int index\_name(int key,int probe){

    int index;

    index=key%53+probe;

}

int main(){

    FILE \* fp;

    char buff[100];

    fp = fopen("names.txt","r");

    int asc;

    int flag=0;

    int k=0;

    int index=0;

    int count=0;

    if (fp == NULL)

    {

        perror("Error opening file\n");

    }

    else

        while(fgets(buff,sizeof(buff),fp))

        {

          asc=hash(buff);

          k=0;

          flag=0;

          while(flag==0)

          {

              index=index\_name(asc,k);

              if(strcmp(arr[index].name,buff)==0){

                  arr[index].freq++;

                  flag=1;

              }

              else if(strlen(arr[index].name)>1){

                  k++;

                  count++;

              }

              else {

                  strcpy(arr[index].name,buff);

                  arr[index].freq=1;

                  flag=1;

              }

        }

char s[100];

k = 0;

char z[100];

strcpy(z,"\n");

printf("enter string\n");

gets(s);

strcat(s,z);

int count1=0;

a=hash(s);

flag=0;

while(flag==0){

index=find\_index(a,k);

if(strcmp(s1[index].name,s)==0){

printf("frequency=%d\nnumber of collisions=%d\nindex=%d\nvalue at index=%s\n",s1[index].freq,count,index,s1[index].name);

flag=1;

}

else if(strlen(s1[index].name)>1){

k++;

count1++;

}

else{

printf("not found");

break;

}

}

}

The hash function used here is better than the previous one because it reduces the number of collisions. From number of collisions=38 in the first program the number of collisions gets reduced to 23 in the second one.