\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Report: HW8

Author: C14031162 洪正皇 <a0987856762@gmail.com>

Class: 乙班

Description:

將IP用link list的方式掛起來，做insert、delete、search等動作

I have learn:

了解link list用法，在gdb模式下找bug。

Error message:

Segmentation fault

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Code:

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

//一個node的結構

typedef struct node{

int value[5];

struct node \*next;

}Node;

int main()

{

char IP[20];//IP用來抓IP，一次做一行

int i,j,length,tmp,pos,insert[5];//insert的五格分別儲存小數點及斜線隔開的五組數字，length,tmp用來將IP中的數字一組一組放進insert，例如125=>length=3,tmp=100+20+5,pos儲存insert存到哪一格。

Node \*one[256];//這邊先宣告三組長度範圍的起點

Node \*two[4096];

Node \*tre[4096];

for(i=0;i<256;i++) {one[i] = (Node\*)malloc(sizeof(Node)); one[i]->next=NULL; }//各起點初始化

for(i=0;i<4096;i++) {two[i] = (Node\*)malloc(sizeof(Node)); two[i]->next=NULL; }

for(i=0;i<4096;i++) {tre[i] = (Node\*)malloc(sizeof(Node)); tre[i]->next=NULL; }

FILE \*fp;//開啟資料

fp=fopen("IPv4\_400k.txt","r");

if (!fp) {

printf("檔案開啟失敗...\n");

exit(1);

}

else

{

Node \*ptr=NULL;//用來移動去找適合位置

Node \*new\_node;//儲存新輸入的IP位置後掛在找到的ptr下方

for(i=0;i<20;i++) IP[i]=-10;//初始化-10

fscanf(fp,"%s",&IP);//讀一串IP

while(IP[0]!=-10)//檔案結束前持續掛上IP，一次一串

{

i=0;

while(IP[i]!=-10)//將字元換成正確數字

{

if(IP[i]<='9'&&IP[i]>='0')

IP[i]=IP[i]-'0';

i++;

}

IP[i-1]=-10;

j=pos=0;

while(IP[j]!=-10)//記錄這串IP的各組數字存進insert

{

length=tmp=0;

i=j;

while(IP[i]>=0&&IP[i]<=9)

{

length++;

i++;

}

for(i=j;i<j+length;i++)

{

tmp=tmp+(IP[i]\*(int)pow(10,j+length-i-1));

}

insert[pos]=tmp;

pos++;

j=i+1;

}

new\_node = (Node\*)malloc(sizeof(Node));//給新節點一個空間

new\_node->next=NULL;

//檢查此串IP要掛在哪一組陣列，將ptr持續往下指到尾端，掛上new\_node

if(insert[4]>=8&&insert[4]<16)

{

ptr=one[insert[0]];

while(ptr->next!=NULL)

{ptr=ptr->next;}

ptr->next=new\_node;

}

else if(insert[4]>=16&&insert[4]<25)

{

ptr=two[insert[0]\*16+(int)insert[1]/16];

while(ptr->next!=NULL)

{ptr=ptr->next;}

ptr->next=new\_node;

}

else if(insert[4]>=25&&insert[4]<=32)

{

ptr=tre[insert[0]\*16+(int)insert[1]/16];

while(ptr->next!=NULL)

{ptr=ptr->next;}

ptr->next=new\_node;

}

for(i=0;i<5;i++){//將IP分五組數字存進new\_node

new\_node->value[i]=insert[i];}

for(i=0;i<20;i++) IP[i]=-10;//初始化，再抓一行IP

fscanf(fp,"%s",&IP);

}

}

fclose(fp);//讀完後關檔

FILE \*fop;//接下來要做搜尋，開一個result檔案寫出

fop=fopen("result.txt","w");

if (!fp) {

printf("檔案開啟失敗...\n");

exit(1);

}

//開檔搜尋

fp=fopen("IPv4\_search.txt","r");

if (!fp) {

printf("檔案開啟失敗...\n");

exit(1);

}

else

{

int match,print[5];//match紀錄是否有找到，print儲存要印出的IP

Node \*ptr=NULL;

Node \*pre\_ptr=NULL;

for(i=0;i<20;i++) IP[i]=-10;

fscanf(fp,"%s",&IP);

while(IP[0]!=-10)

{

match=0;

i=0;

while(IP[i]!=-10)

{

if(IP[i]<='9'&&IP[i]>='0')

IP[i]=IP[i]-'0';

i++;

}

IP[i-1]=-10;

j=pos=0;

while(IP[j]!=-10)//將一串IP字元輸出成五組數字

{

length=tmp=0;

i=j;

while(IP[i]>=0&&IP[i]<=9)

{

length++;

i++;

}

for(i=j;i<j+length;i++)

{

tmp=tmp+(IP[i]\*(int)pow(10,j+length-i-1));

}

insert[pos]=tmp;

pos++;

j=i+1;

}

for(i=0;i<5;i++) print[i]=0;

ptr=tre[insert[0]\*16+(int)insert[1]/16];//從最長的那組分組開始找

if(match==0&&ptr->next!=NULL)//match代表沒找到過，不過在此還沒開始找當然是0，後面兩組長度分組會有用

{

while(ptr->next!=NULL)//一直search到尾端

{

ptr=ptr->next;

if(ptr->value[0]==insert[0]&&ptr->value[1]==insert[1]&&ptr->value[2]==insert[2]&&(int)(ptr->value[3]/((int)pow(2,32-ptr->value[4])))==(int)(insert[3]/((int)pow(2,32-ptr->value[4]))))

{//長度是幾，就去配對幾bit，若配對到先暫存，繼續尋找是否有更長

match=1;

if(ptr->value[4]>print[4])

{

for(i=0;i<5;i++) print[i]=ptr->value[i];

}

}

}

}

ptr=two[insert[0]\*16+(int)insert[1]/16];//開始找第二組長的分組

if(match==0&&ptr->next!=NULL)

{

while(ptr->next!=NULL)

{

ptr=ptr->next;

if(ptr->value[0]==insert[0]&&ptr->value[1]==insert[1]&&(int)(ptr->value[2]/((int)pow(2,24-ptr->value[4])))==(int)(insert[2]/((int)pow(2,24-ptr->value[4]))))

{

match=1;

if(ptr->value[4]>print[4])

{

for(i=0;i<5;i++) print[i]=ptr->value[i];

}

}

}

}

ptr=one[insert[0]];//找最後一組長度分組

if(match==0&&ptr->next!=NULL)

{

while(ptr->next!=NULL)

{

ptr=ptr->next;

if(ptr->value[0]==insert[0]&&(int)(ptr->value[1]/((int)pow(2,16-ptr->value[4])))==(int)(insert[1]/((int)pow(2,16-ptr->value[4]))))

{

match=1;

if(ptr->value[4]>print[4])

{

for(i=0;i<5;i++) print[i]=ptr->value[i];

}

}

}

}

//找完後就印出來

for(i=0;i<3;i++) fprintf(fop,"%d.",print[i]);

fprintf(fop,"%d/",print[3]); fprintf(fop,"%d\n",print[4]);

for(i=0;i<20;i++) IP[i]=-10;

fscanf(fp,"%s",&IP);

}

fclose(fop);

}

fclose(fp);

//開檔刪除

fp=fopen("IPv4\_delete.txt","r");

if (!fp) {

printf("檔案開啟失敗...\n");

exit(1);

}

else

{

Node \*ptr=NULL;

Node \*pre\_ptr=NULL;

for(i=0;i<20;i++) IP[i]=-10;

fscanf(fp,"%s",&IP);

while(IP[0]!=-10)

{

i=0;

while(IP[i]!=-10)

{

if(IP[i]<='9'&&IP[i]>='0')

IP[i]=IP[i]-'0';

i++;

}

IP[i-1]=-10;

j=pos=0;

while(IP[j]!=-10)

{

length=tmp=0;

i=j;

while(IP[i]>=0&&IP[i]<=9)

{

length++;

i++;

}

for(i=j;i<j+length;i++)

{

tmp=tmp+(IP[i]\*(int)pow(10,j+length-i-1));

}

insert[pos]=tmp;

pos++;

j=i+1;

}

//檢查IP的長度，要分在哪一組

if(insert[4]>=8&&insert[4]<16)

{

ptr=one[insert[0]];

while(ptr->next!=NULL)

{

pre\_ptr=ptr;

ptr=ptr->next;

if(ptr->value[0]==insert[0]&&ptr->value[1]==insert[1]&&ptr->value[2]==insert[2]&&ptr->value[3]==insert[3]&&ptr->value[4]==insert[4])

{//若找到此IP，把上一個link list指向下一個，free掉要刪除的這個

if(ptr->next==NULL)

{

pre\_ptr->next=NULL;

free(ptr);

break;

}

else

{

pre\_ptr->next=ptr->next;

free(ptr);

break;

}

}

}

}//找第二組長度

else if(insert[4]>=16&&insert[4]<25)

{

ptr=two[insert[0]\*16+(int)insert[1]/16];

while(ptr->next!=NULL)

{

pre\_ptr=ptr;

ptr=ptr->next;

if(ptr->value[0]==insert[0]&&ptr->value[1]==insert[1]&&ptr->value[2]==insert[2]&&ptr->value[3]==insert[3]&&ptr->value[4]==insert[4])

{

if(ptr->next==NULL)

{

pre\_ptr->next=NULL;

free(ptr);

break;

}

else

{

pre\_ptr->next=ptr->next;

free(ptr);

break;

}

}

}

}//找第三組長度

else if(insert[4]>=25&&insert[4]<=32)

{

ptr=tre[insert[0]\*16+(int)insert[1]/16];

while(ptr->next!=NULL)

{

pre\_ptr=ptr;

ptr=ptr->next;

if(ptr->value[0]==insert[0]&&ptr->value[1]==insert[1]&&ptr->value[2]==insert[2]&&ptr->value[3]==insert[3]&&ptr->value[4]==insert[4])

{

if(ptr->next==NULL)

{

pre\_ptr->next=NULL;

free(ptr);

break;

}

else

{

pre\_ptr->next=ptr->next;

free(ptr);

break;

}

}

}

}

for(i=0;i<20;i++) IP[i]=-10;

fscanf(fp,"%s",&IP);

}

}

fclose(fp);

//開檔插入

fp=fopen("IPv4\_insert.txt","r");

if (!fp) {

printf("檔案開啟失敗...\n");

exit(1);

}

else

{

Node \*ptr=NULL;

Node \*new\_node;

for(i=0;i<20;i++) IP[i]=-10;

fscanf(fp,"%s",&IP);

while(IP[0]!=-10)

{

i=0;

while(IP[i]!=-10)

{

if(IP[i]<='9'&&IP[i]>='0')

IP[i]=IP[i]-'0';

i++;

}

IP[i-1]=-10;

j=pos=0;

while(IP[j]!=-10)

{

length=tmp=0;

i=j;

while(IP[i]>=0&&IP[i]<=9)

{

length++;

i++;

}

for(i=j;i<j+length;i++)

{

tmp=tmp+(IP[i]\*(int)pow(10,j+length-i-1));

}

insert[pos]=tmp;

pos++;

j=i+1;

}

//先宣告新節點要插入

new\_node = (Node\*)malloc(sizeof(Node));

new\_node->next=NULL;

if(insert[4]>=8&&insert[4]<16)//分類此新IP長度

{

ptr=one[insert[0]];

while(ptr->next!=NULL)

{ptr=ptr->next;}

ptr->next=new\_node;

}

else if(insert[4]>=16&&insert[4]<25)

{

ptr=two[insert[0]\*16+(int)insert[1]/16];

while(ptr->next!=NULL)

{ptr=ptr->next;}

ptr->next=new\_node;

}

else if(insert[4]>=25&&insert[4]<=32)

{

ptr=tre[insert[0]\*16+(int)insert[1]/16];

while(ptr->next!=NULL)

{ptr=ptr->next;}

ptr->next=new\_node;

}

//把IP放入新節點中

for(i=0;i<5;i++){

new\_node->value[i]=insert[i];}

for(i=0;i<20;i++) IP[i]=-10;

fscanf(fp,"%s",&IP);

}

}

fclose(fp);

return 0;

}

Compilation:

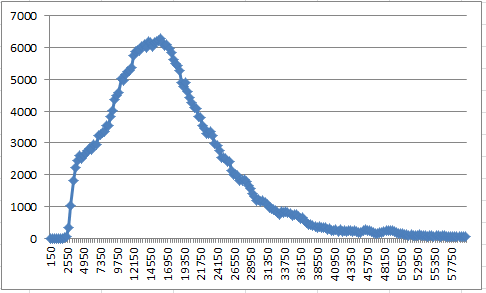
gcc –lm -o hw8 hw8.c

Execution:

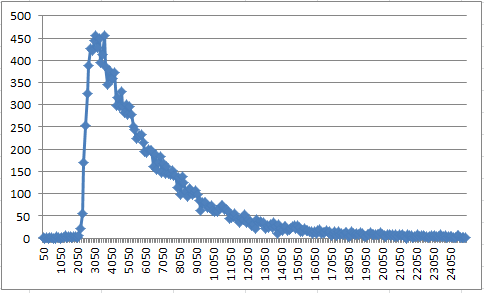
./hw8

Output:

一個result.txt檔案

Search time average : 17876.58414

Delete time average : 6679.49505



Insert time average : 14758.34765

