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

Report: HW6

Author: E94046157 蔡宇軒 <[zfk.662624@gmail.com](mailto:zfk.662624@gmail.com)>

Class: 二乙

Description:

In this program, I learned how to create a linked list, and combine it with array. I refer the code that professor provided. And I modified it so it can use with array.After finishing this program , I learned pointer and linked list is important in C language, and it’s very fun,too.

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

#include<stdio.h>

#include <stdlib.h>

#include <time.h>

//------------time

inline unsigned long long int rdtsc()

{

unsigned long long int x;

asm volatile ("rdtsc" : "=A" (x));

return x;

}

typedef struct node {

unsigned int data;

struct node \*next;

}node;

//---------------------

void insert(node \*\*h,node \*p);

node\* search\_a\_list(node \*head[],unsigned int d,int m);

node\* delete(node \*\*head,unsigned int d,int m);

//--------------------

int main(int argc,char\*argv[])

{

int k=atoi(argv[1]);int n=atoi(argv[2]);int m=atoi(argv[3]);

srand(time(NULL));

unsigned int a,r,tmp[m]; int i,total=1;unsigned int index;

for(i=0;i<k;i++)total\*=2; node\*head[total]; for(i=0;i<total;i++)head[i]=NULL;

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

{

a=rand(); a=(a & 1) << 31; r=rand(); r = r | a;

node \*new=malloc(sizeof(node));

new->data=r;new->next=NULL;

index=r>>(32-k);

insert(&(head[index]),new);

//------------------------------search

while(i<m){tmp[i]=r;break;}

}

unsigned int i1;unsigned long long int begin,end,total\_time,average;

begin=rdtsc();

for(i=0;i<m;i++){i1=tmp[i];node\*f1=search\_a\_list(head,i1,total);}

end=rdtsc();

total\_time=end-begin; average=(total\_time/m);

printf("average search clock cycles:%lu\n",average);

FILE \*fptr;

fptr=fopen("result.txt","wb");

for(i=0;i<m;i++)fprintf(fptr, "%u\r\n",tmp[i]);

//---------------------------another m

begin=rdtsc();

for(i=0;i<m;i++)

{

a=rand(); a=(a & 1) << 31; r=rand(); r = r | a;

node \*new=malloc(sizeof(node));

new->data=r;new->next=NULL;

index=r>>(32-k);

insert(&(head[index]),new);

while(i<m){tmp[i]=r;break;}

}

end=rdtsc();

total\_time=end-begin; average=(total\_time/m);

printf("average insert clock cycles:%lu\n",average);

begin=rdtsc();

for(i=0;i<=1;i++){i1=tmp[i];node\*f1=delete(head,i1,total);}

end=rdtsc();

total\_time=end-begin; average=(total\_time/m);

printf("average delete clock cycles:%lu\n",average);

}

//------------------------------------------end of main function

void insert(node \*\*head, node \*p)

{

node \*t = \*head;

if(p==NULL) return;

if(\*head == NULL) {\*head = p; return;}

if (p->data <= (\*head)->data){

p->next = \*head;

\*head = p;

return;

}

while (t->next !=NULL && t->next->data < p->data)

t = t->next;

p->next = t->next;

t->next = p;

return;

}

node \*search\_a\_list(node \*head[],unsigned int d,int m)

{

int i;

for(i=0;i<m;i++)

{

node \* t=head[i];

while (t != NULL && t->data != d){ if(t->data >d) break; t = t -> next;}

if(t==NULL) ;else return t;

}

}

node \*delete(node \*head[],unsigned int d,int m)

{

int i;

for(i=0;i<m;i++)

{

node \*t=head[i], \*prev=NULL;

while (t != NULL && t->data != d){

if(t->data >d) break;

prev = t;

t = t -> next;

}

if(prev==NULL&&t!=NULL) {head[i] = t->next; t->next=NULL;return t;}

else if(t==NULL); else{prev->next = t->next;t->next=NULL; return t;}

}

}Compilation:

gcc -o hw8 hw8.c

Execution:

./hw8 6 10000 1000

Output:

average search clock cycles:132119

average insert clock cycles:2639

average delete clock cycles:92