**Exercise 11: Searching and its Applications**

**adt.h:**

#include<stdio.h>

struct num

{

int A[100];

int n;

};

void init(struct num\*,int );

void ins(struct num\*,int[]);

void ls(struct num\*,int);

int bs(struct num\*,int);

void disp(struct num\*);

**impl.h:**

#include "adt.h"

void init(struct num\* N,int n)

{

N->n=n;

}

void ins(struct num\* N, int x[])

{

for(int i=0;i<N->n;i++)

{

N->A[i]=x[i];

}

}

void ls(struct num\* N,int x)

{

for(int i=0;i<N->n;i++)

{

if(x==N->A[i])

{

printf("\n%d is found at pos: %d\n",x,i+1);

break;

}

}

}

int bs(struct num\* N,int x)

{

int low=0,high=N->n-1;

while (low <= high)

{

int mid = low + (high - low) / 2;

if (N->A[mid] == x)

return mid;

if (N->A[mid] < x)

low = mid + 1;

else

high = mid - 1;

}

return -1;

}

void disp(struct num\* N)

{

for(int i=0;i<N->n;i++)

{

printf("%d ",N->A[i]);

}

}

**appl.c:**

#include "impl.h"

#include<stdlib.h>

int freq(struct num \*N,int x)

{

/\*int count=0;

for(int i=0;i<N->n;i++)

{

if(x==N->A[i])

{

//printf("\n%d is found at %d\n",x,i);

//break;

count++;

}

//if(count==2)

//return i+1;

}

return count;\*/

return ls2(N,x);

}

void pairs(struct num \*N,int k)

{

if(k==0)

return;

//printf("\nhello\n");

for(int gap=N->n/2;gap>0;gap/=2)

{

for(int i=gap;i<N->n;i++)

{

for(int j=i-gap;j>=0;j-=gap)

{

if(N->A[j+gap]>N->A[j])

break;

else

{

int temp=N->A[j+gap];

N->A[j+gap]=N->A[j];

N->A[j]=temp;

}

}

}

}

//disp(N);

/\*

for(in ti=0;i<N->n;i++)

{

if(bs(N,N->A[i]+k)

if(bs(N,N->A[i]-k)

}

\*/

int l,r,count=0;

for(l=r=0; r<N->n; )

{

count=0;

int lc=freq(N,N->A[l]);

int rc=freq(N,N->A[r]);

if(abs(N->A[r]-N->A[l])==k)

{

/\*for(int i=0;i<lc\*rc;i++)

{

printf("[%d, %d]\n", N->A[l], N->A[r]);

}\*/

count=lc\*rc;

printf("[%d, %d] : %d\n", N->A[l], N->A[r],count);

l+=lc;

r+=rc;

}

else if(N->A[r]-N->A[l]>k)

l+=lc;

else

r+=rc;

}

}

int same(struct num \*N1,struct num \*N2)

{

/\*if(N1->n!=N2->n)

return 0;

int a,b,n=N1->n;

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

{

for(int j=i+1;j<n;++j)

{

if(N1->A[i]>N1->A[j])

{

a=N1->A[i];

N1->A[i]=N1->A[j];

N1->A[j]=a;

}

if(N2->A[i]>N2->A[j])

{

b=N2->A[i];

N2->A[i]=N2->A[j];

N2->A[j]=b;

}

}

}

int i;

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

{

if(N1->A[i]!=N2->A[i])

return 0;

}

return 1;\*/

if(N1->n!=N2->n)

return 0;

int n=N1->n;

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

{

if(!ls3(N2,N1->A[i]))

return 0;

}

return 1;

}

void nThOcc(struct num \*N,int x,int k)

{

/\*int count=0;

for(int i=0;i<N->n;i++)

{

if(x==N->A[i])

{

//printf("\n%d is found at %d\n",x,i);

//break;

count++;

}

if(count==2)

return i+1;

}

return -1;\*/

int low=0,high=N->n-1,left,right;

while (low <= high)

{

int mid = low + (high - low) / 2;

//int left;

if (N->A[mid] == x)

{

high=mid-1;

left=mid;

}

if (N->A[mid] < x)

low=mid+1;

else

high=mid-1;

}

low=0,high=N->n-1;

while (low <= high)

{

int mid = low + (high - low) / 2;

if (N->A[mid] == x)

{

low=mid+1;

right=mid;

}

if (N->A[mid] < x)

low=mid+1;

else

high=mid-1;

}

//int k=2;

int n=right-left+1;

if(n<k)

{

printf("\n%d only occurs %d times!",x,n);

return;

}

int i=left+k-1;

if(N->A[i]==x)

{

printf("\n%d's second occurance is found at %d!",x,i);

//return i;

}

else

printf("\n%d's second occurance is not found!",x);

//return -1;

}

int main()

{

struct num \*N=(struct num\*)malloc(sizeof(struct num));

init(N,10);

int x[20],x1[20],f,ch,n,ele,min;

//int x[]={4,7,3,2,1,9,8,7,3,4};

//int x1[]={4,7,3,2,1,9,8,7,3,4};

printf("\nMenu:\n1.Linear Search\n2.Binary Search\n3.Find posiiton of second oocurrence\n4.Find Frequency\n5.Check whether 2 lists are same\n6.Retrieve pairs with k differnce\n7.Exit\n");

do{

printf("\nChoice: ");

scanf("%d",&ch);

switch(ch)

{

case 1:

printf("\nEnter the number of elements: ");

scanf("%d",&n);

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

{

printf("Enter the element: ");

scanf("%d",&x[i]);

x1[i]=x[i];

}

printf("\nArray: ");

ins(N,x1);

disp(N);

printf("\nLS:\n");

//disp(N);

printf("\nEnter element to be found: ");

scanf("%d",&f);

//int f=7;

ls(N,f);

break;

case 2:

/\*printf("\nEnter the number of elements: ");

scanf("%d",&n);

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

{

printf("Enter the element: ");

scanf("%d",&x[i]);

x1[i]=x[i];

}

printf("\nArray: ");

ins(N,x1);

disp(N); \*/

printf("\nEnter element to be found: ");

scanf("%d",&f);

//int f=7;

min=0;

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

{

for (int j=i+1;j<n;++j)

{

if(x[i]>x[j])

{

min=x[i];

x[i]=x[j];

x[j]=min;

}

}

}

ins(N,x);

printf("\n\nBS:\n");

disp(N);

//f=8;

int k=bs(N,f);

if(k==-1)

printf("\nElement %d is not there!\n",f);

else

printf("\nElement %d is found at %d index\n",f,k);

break;

case 3:

/\*printf("\nEnter the number of elements: ");

scanf("%d",&n);

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

{

printf("Enter the element: ");

scanf("%d",&x[i]);

x1[i]=x[i];

}

printf("\nArray: ");

ins(N,x1);

printf("\nLS:\n");

disp(N);\*/

printf("\nEnter element whose second occ has to be found: ");

scanf("%d",&f);

//int f=7;

/\*int min=0;

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

{

for (int j=i+1;j<n;++j)

{

if(x[i]>x[j])

{

min=x[i];

x[i]=x[j];

x[j]=min;

}

}

}\*/

ins(N,x);

printf("\n\n\nSecond Occ:");

//f=7;

//ins(N,x);4

disp(N);

nThOcc(N,f,2);

break;

case 4:

printf("\nEnter element whose freq is needed: ");

scanf("%d",&f);

printf("\n\nFreq:");

int h=freq(N,f);

printf("\n%d's freq: %d\n",f,h);

break;

case 5:

printf("\nEnter the number of elements: ");

struct num \*N1=(struct num\*)malloc(sizeof(struct num));

struct num \*N2=(struct num\*)malloc(sizeof(struct num));

int n1,n2,A[20],B[20];

scanf("%d",&n1);

for(int i=0;i<n1;i++)

{

printf("Enter the element: ");

scanf("%d",&A[i]);

}

printf("\nArray: ");

init(N1,n1);

ins(N1,A);

disp(N1);

printf("\nEnter the number of elements: ");

scanf("%d",&n2);

for(int i=0;i<n2;i++)

{

printf("Enter the element: ");

scanf("%d",&B[i]);

}

printf("\nArray: ");

init(N2,n2);

ins(N2,B);

disp(N2);

printf("\nSame?: \n");

int i=same(N1,N2);

if(i)

printf("Same\n");

else

printf("Not same\n");

break;

case 6:

printf("\nEnter the number of elements: ");

struct num \*N3=(struct num\*)malloc(sizeof(struct num));

int n3,C[20];

scanf("%d",&n3);

init(N3,n3);

for(int i=0;i<n3;i++)

{

printf("Enter the element: ");

scanf("%d",&C[i]);

}

printf("\nArray: ");

ins(N3,C);

disp(N3);

printf("\nEnter difference: ");

scanf("%d",&f);

printf("\n\nPairs for diff %d:\n",f);

pairs(N3,f);

break;

case 7:printf("\nExiting...\n");

break;

default:printf("\nInavlid Input!\n");

}

/\*int A[] = {4,7,3,2,1,9,8};

int B[] = {1,9,8,2,4,3,7};

int a,b;

int n1=sizeof(A)/sizeof(A[0]);

int n2=sizeof(A)/sizeof(A[0]);

init(N1,n1);

init(N2,n2);

ins(N1,A);

ins(N2,B);

printf("\n\n");

disp(N2);

printf("\n");

disp(N1);

int C[] = {5,20,3,2,50,8,20,20,2};

//int C[] = {2, 6, 4, 8, 0, -4, 10, 7, -7, 9};

struct num \*N3=(struct num\*)malloc(sizeof(struct num));

init(N3,sizeof(C)/sizeof(C[0]));

ins(N3,C);

f=18;

//pairs(N3,6);\*/

}while(ch!=7);

return 0;

}

**Sample I/O:**





