DS LK ĐẶC

Hàm nhập mãng

void input (int a[], int n)

{

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

{

cout<<"nhap a["<<i<<"] = ";

cin>>a[i];

}

}

Hàm tìm kiếm

int search (int a[], int n, int x)

{

int i = 0;

while ( (i<n) && (a[i] != x) )

i++;

if (i==n)

return -1;

return i;

}

Thêm phần tử

int Insert (int a[], int &n, int x, int i)

{

if (i>=0 && i < n)

{

for (int j=n-1; j>=i; j--)

a[j+1] = a[j];

a[i] = x;

n++;

return 1;

}

return 0;

}

Xoá Phần tử

int Delete(int a[], int &n, int i)

{

if (i>=0 && i < n)

{

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

a[j] = a[j+1];

n--;

return 1;

}

return 0;

}

Hàm chọn

void Choice(int a[], int n, int x, int vitri)

{

int it;

do{//Menu

cin>>it;

switch(it)

{

case 1:

cout<<"Nhap mang:"<<endl;

input(a,n);

break;

case 2:

cout<<"Mang vua nhap:"<<endl;

output(a,n);

break;

case 3:

cout<<"Xin nhap gia tri tim:";

cin>>x;

vitri=search(a,n,x);

if(vitri != -1)

cout<<x<<" co xuat hien trong mang tai vi

tri:"<<vitri<<endl;

else

cout<<x<<" khong co xuat hien trong

mang"<<endl;

break;

case 4:

cout<<"Xin nhap gia tri can chen:";

cin>>x;

5

cout<<"Xin nhap vi tri can chen trong mang:";

cin>>vitri;

if(Insert(a,n,x,vitri)==1)

{

cout<<x<<" da them vao mang thanh

cong!"<<endl;

}

else

cout<<x<<" da them vao mang khong thanh

cong!"<<endl;

break;

case 5:

cout<<"Xin nhap vi tri can xoa trong mang:";

cin>>vitri;

if(Delete(a,n,vitri)==1)

{

cout<<"Phan tu tai vi tri:"<<vitri<<" da duoc

xoa thanh cong!"<<endl;

}

else

cout<<"Xoa khong thanh cong!"<<endl;

break;

default:

cout<<"Ban da chon chuc nang thoat chuong trinh!";

break;

}

}while(it==1||it==2||it==3||it==4||it==5);

}

int main()

{

int n;//nhap so luong phan tu

int a[100];

int x, vitri;

vitri = 0;

x = 0;

//nhap so luong phan tu n

do{

cout<<"Xin nhap n=";

cin>>n;

if(n <= 0 || n>100)

cout<<"Xin nhap n>0 va n <=100";

}while(n <= 0 || n>100);

Choice(a,n,x,vitri);

system("pause");

return 0;

}

DANH SÁCH LIÊN KẾT ĐƠN

Khai báo cấu trúc node ds lk đơn

struct Node

{

int info;

Node \* link;

};

Khai báo Node đầu (first)

Node \* first;

Khai báo hàm khởi tạo danh sách

void init()

{

first = NULL;

}

Hàm nhập cần thêm

void nhapx(int &x)

{

cout<<"Xin nhap gia tri can them/tim:";

cin>>x;

}

Thêm vào đầu danh sách

void Insert\_first(int x)

{

Node \*p;

p = new Node;

p->info = x;

p->link = first;

first = p;

}

Thêm vào cuối danh sách

void Insert\_last(int x)

{

Node \* p;

p = new Node;

p->info = x;

p->link = NULL;

if(first == NULL)//danh sach rong

first = p;

else

{

Node \*q = first;

while(q->link!=NULL)

q = q->link;

q->link = p;

}

}

Duyệt danh sách

void Process\_list()

{

Node \*p;

p = first;

while(p!=NULL)

{

cout<<p->info<<endl;

p = p->link;

}

}

Duyệt ds in ra phần tử chẳn

void Process\_list1()

{

Node \*p;

p = first;

while(p!=NULL)

{

if(p->info % 2 == 0)

{

cout<<p->info<<endl;

}

p = p->link;

}

}

Hàm tìm kiếm

Node \*Search(int x)

{

Node \*p = first;

while((p != NULL)&&(p->info!=x))

p = p->link;

return p;

}

Xoá phần tử đầu

int Delete\_first()

{

if(first != NULL)

{

Node \* p = first;

first = first->link;

delete p;

return 1;

}

return 0;

}

Xoá cuối danh sách

int Delete\_last()

{

if(first != NULL)

{

Node \*p,\*q;

p = first;

q = NULL;

if(p!=NULL)

while(p->link != NULL)

{

q = p;

p p->link;

}

if(p!=first)

q->link = NULL;

else

first = NULL;

delete p;

return 1;

}

return 0;

}

// ham quan ly chuc nang

void Choice(int x)

{

int it;

do{

//menu

cout<<"=================Menu============="<<endl;

cout<<"1. Tao danh sach rong"<<endl;

cout<<"2. Them phan tu vao dau sach"<<endl;

cout<<"3. Them phan tu vao cuoi danh sach"<<endl;

cout<<"4. Duyet va xuat cac phan tu chan"<<endl;

cout<<"5. Tim kiem"<<endl;

cout<<"6. Xoa phan tu dau"<<endl;

cout<<"7. Xoa phan tu cuoi"<<endl;

cout<<"8. Xuat danh sach"<<endl;

cout<<"9. Thoat chuong trinh"<<endl;

cout<<"=================================="<<endl;

cout<<"Xin chon chuc nang:";

cin>>it;

switch(it)

{

case 1:

init();

break;

case 2:

nhapx(x);

Insert\_first(x);

break;

case 3:

nhapx(x);

10

Insert\_last(x);

break;

case 4:

cout<<"Cac phan tu co gia tri la so chan trong

danh dach:"<<endl;

Process\_list1();

break;

case 5:

nhapx(x);

if(Search(x) != NULL)

cout<<x<<" co xuat hien trong danh sach

tai Node:"<<Search(x)<<endl;

else

cout<<x<<" khong co xuat hien trong danh

sach"<<endl;

break;

case 6:

if(Delete\_first() != 0)

cout<<"Gia tri dau danh sach da duoc xoa

thanh cong"<<endl;

else

cout<<"Gia tri dau danh sach khong duoc

xoa thanh cong"<<endl;

break;

case 7:

if(Delete\_last() != 0)

cout<<"Gia tri cuoi danh sach da duoc xoa

thanh cong"<<endl;

else

cout<<"Gia tri cuoi danh sach khong duoc

xoa thanh cong"<<endl;

break;

case 8:

cout<<"Danh sach cac phan tu trong danh

sach:"<<endl;

Process\_list();

break;

default:

1

cout<<"Ban da chon chuc nang thoat chuong

trinh."<<endl;

break;

}

}while(it>=1 && it<=8);

}

int main()

{

int x=0;

Choice(x);

system("pause");

return 0;

}

DANH SÁCH LIÊN KẾT ĐÔI

#include "iostream"

#include "conio.h"

using namespace std;

//khai bao cau truc node ds lk don

struct Node

{

int info;

Node \*next, \*previous;

};

//Khai bao Node dau(first)

Node \* first, \*last;

//khai bao ham khoi tao danh sach

void init()

{

first = NULL;

last = NULL;

}

//ham nhap gia tri can them

void nhapx(int &x)

{

cout<<"Xin nhap gia tri can them/tim:";

cin>>x;

}

//them phan tu vao dau danh sach

void Insert\_first(int x)

{

Node \*p;

p = new Node;

p->info = x;

p->next= first;

p->previous = NULL;

if(first != NULL)

first->previous = p;

else // danh sách rỗng, trước khi thêm p

last = p; // sau khi them phần tử p, danh sách có 1

phần tử

first = p;

}

//Them phan tu vao cuoi danh sach

void Insert\_last(int x)

{

Node \*p;

p = new Node;

p->info = x;

p->next = NULL;

p->previous = last;

if(last != NULL)

last->next = p;

else // danh sách rỗng, trước khi thêm p

first = p; // sau khi them phần tử p, danh sách có

1 phần tử

last = p;

}

//Duyet danh sach

void Process\_list()

{

Node \*p;

p = first;

while (p!=NULL)

{

cout<<p -> info<<endl;

p=p->next;

}

}

//tim kiem, duyet danh sach tu dau den cuoi

Node \*Search(int x)

{

Node \*p;

p = first;

while ( (p!=NULL) && (p->info != x) )

p=p->next;

return p;

}

//Xoa phan tu dau

int Delete\_first()

{

if (first != NULL)

{

Node \*p = first;

first = first->next;

delete p;

if (first !=NULL) // trường hợp ds khác rỗng phần

tử nào

first->previous = NULL;

else

last = NULL; // trường hợp ds không còn phần tử nào

return 1;

}

return 0;

}

//xoa cuoi danh sach

int Delete\_last()

{

if (last != NULL)

{

Node \*p = last;

last = last->previous;

if (last !=NULL) // trường hợp ds khác rỗng nào

last ->next= NULL;

else

first = NULL; // trường hợp ds không còn phần tử nào

delete p;

return 1;

}

return 0;

}

// ham quan ly chuc nang

void Choice(int x)

{

int it;

do{

//menu

cout<<"=================Menu============="<<endl;

cout<<"1. Tao danh sach rong"<<endl;

cout<<"2. Them phan tu vao dau sach"<<endl;

cout<<"3. Them phan tu vao cuoi danh sach"<<endl;

cout<<"4. Duyet danh sach"<<endl;

cout<<"5. Tim kiem"<<endl;

cout<<"6. Xoa phan tu dau"<<endl;

cout<<"7. Xoa phan tu cuoi"<<endl;

cout<<"8. Thoat chuong trinh"<<endl;

cout<<"=================================="<<endl;

cout<<"Xin chon chuc nang:";

cin>>it;

switch(it)

{

case 1:

init();

break;

case 2:

nhapx(x);

Insert\_first(x);

break;

case 3:

nhapx(x);

Insert\_last(x);

break;

case 4:

cout<<"Cac phan tu trong danh dach:"<<endl;

Process\_list();

break;

case 5:

nhapx(x);

if(Search(x) != NULL)

cout<<x<<" co xuat hien trong danh sach

tai Node:"<<Search(x)<<endl;

else

cout<<x<<" khong co xuat hien trong danh

sach"<<endl;

break;

case 6:

if(Delete\_first() != 0)

cout<<"Gia tri dau danh sach da duoc xoa thanh cong"<<endl;

else

cout<<"Gia tri dau danh sach khong duoc xoa thanh cong"<<endl;

break;

case 7:

if(Delete\_last() != 0)

cout<<"Gia tri cuoi danh sach da duoc xoa thanh cong"<<endl;

else

cout<<"Gia tri cuoi danh sach khong duoc xoa thanh cong"<<endl;

break;

default:

cout<<"Ban da chon chuc nang thoat chuong trinh."<<endl;

break;

}

}while(it>=1 && it<=7);

}

int main()

{

int x=0;

Choice(x);

system("pause");

return 0;

}

thứ tự tăng dần bằng thuật toán

Selection Sort

void hoanvi(int &x, int &y)

{

int tam = x;

x = y;

y = tam;

}

void SelectionSort(int a[], int n )

{

int min;

for (int i=0; i<n-1 ; i++)

{

min = i;

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

if (a[j] < a[min])

min = j;

if (min != i)

hoanvi(a[min], a[i]);

}

}

thứ tự tăng dần bằng thuật toán

Interchange Sort

void InterchangeSort(int a[], int n)

{

int i, j;

for (i = 0 ; i<n-1 ; i++)

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

if(a[j]< a[i])

hoanvi(a[i], a[j]);

}

thứ tự tăng dần bằng thuật toán Bubble

Sort

void BubbleSort(int a[], int n)

{

int i, j;

for (i = 0 ; i<n-1 ; i++)

for (j =n-1; j>i ; j --)

if(a[j]< a[j-1])

hoanvi(a[j], a[j-1]);

}

thứ tự tăng dần bằng thuật toán Heap Sort

void shift(int a[], int i, int n)

{

int j = 2\*i+1;

if (j>=n) // nếu vị trí j không tồn tại trong danh sách

đang xét thì thoát khỏi chương trình

return;

if (j+1 <n) // nếu không tồn tại vị trí j+1 trong danh

sách đang xét thì thoát khỏi chương trình

if ( a[j]<a[j+1] ) // nếu vị trí j tồn tại phần tử a[j]<a[j+1]

j++;

if (a[i] >= a[j] )

return;

else

{

int x = a[i];

a[i] = a[j];

a[j] = x;

shift(a, j, n);

}

}

void HeapSort(int a[], int n)

{

int i = n/2;

while (i >=0) // tạo heap ban đầu

{

shift(a, i, n-1); i - -;

}

int right=n-1; // right là vị trí cuối Heap đang xét

while (right>0)

{

if(a[0] > a[right])

swap(a[0], a[right]); // hoán vị phần tử a[0] cho phần tử

cuối Heap đang xét

right - -; // giới hạn lại phần tử cuối đang xét

if (right > 0) // Kiểm tra dãy đang xét còn nhiều hơn 1 phần tử

shift(a, 0, right); // tạo Heap lại tại vị trí 0

}

}

Quick Sort

void QuickSort(int a[], int left, int right)

{

int i, j, x;

x = a[(left+right)/2]; //Chọn phần tử giữa làm giá trị mốc

i = left;

j = right;

while(i <= j)

{

while(a[i] < x) i++;

while(a[j] > x) j--;

if(i <= j)

{

hoavi(a[i], a[j]); //đổi chỗ a[i] và a[j]

i++ ; j--;

}

}

if(left < j)

QuickSort(a, left, j);

if(i < right)

QuickSort(a, i, right);

}

TÌM KIẾM

int timkiem(int a[], int n, int x)

{

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

{

if(a[i] == x)

return i;

}

return -1;

}

void timkiem(int a[], int n, int x)

{

int b[100];

int dem = 0;// dem so lan x xuat hien

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

{

if(a[i] == x)

{

b[dem ++] = i;

}

}

if(dem != 0)

{

cout<<x<<” co xuat hien ”<<dem<<” lan trong mang tai

cac vi tri:”;

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

cout<<b[i]<<”\t”;

}

else

cout<<X<<” khong co xuat hien trong mang”;

}

BinarySearch

int BinarySearch(int a[], int n, int x)

{

int left=0, right=n-1, mid;

while(left<=right)

{

mid=(left+right)/2;

if(a[mid]==x) return mid;

if(x>a[mid]) left=mid+1;

else right=mid-1;

}

return -1; // không tìm thấy x trong danh sách a;

}

Cây nhị phân tìm kiếm

#include "iostream"

#include "conio.h"

using namespace std;

// cấu trúc 1 node

struct Node

{

int info; // lưu trữ giá trị của phần tử, giá trị khóa

của node

Node \*left; // left lưu địa chỉ phần tử bên trái (cây

con trái)

Node \*right; //right lưu trữ địa chỉ phần tử bên phải

(cây con phải)

};

Node \*root;

// cấu trúc 1 node

void tree\_empty()

{

root = NULL;

}

//ham nhap gia tri can them

void nhapx(int &x)

{

cout<<"Xin nhap gia tri can them/tim:";

cin>>x;

}

//tim kiem

Node \*search(Node \*p, int x)

{

while (p != NULL)

{

if (p->info == x)

return p;

else if (p -> info > x)

p=p -> left;

else // p->info<x

p=p -> right;

}

return NULL;

}

//them Node

void InsertNode(Node \*&p, int x)

{

if (p == NULL)

{

p=new Node;

p->info = x;

p->left = NULL;

p->right = NULL;

}

else

{

if (p->info == x)

return; // đã có node có giá trị x

else if (p -> info > x)

return InsertNode(p -> left, x);

else

return InsertNode(p -> right, x);

}

}

//duyet cay

void duyetLNR(Node \*p)

{

if (p != NULL)

{

duyetLNR(p->left);

cout<<p->info<<" ";

duyetLNR(p->right);

}

}

void duyetNLR(Node \*p)

{

if (p != NULL)

{

cout<<p->info<<" ";

duyetLNR(p->left);

duyetLNR(p->right);

}

}

void duyetLRN(Node \*p)

{

if (p != NULL)

{

duyetLNR(p->left);

duyetLNR(p->right);

cout<<p->info<<" ";

}

}

// xoa node

void searchStandFor(node \*&p, Node \*&q)

{

if (q->left = = NULL)

{

p->info == q->info

p = q;

q = q -> right;

}

else

searchStandFor(p, q->left)

}

int Delete(Node \*&T, int x)

{

if (T== NULL) return 0;

if (T->info = = x)

{

Node \*p = T;

if (T->left == NULL)

T=T->right;

else if (T -> right == NULL)

T=T->left;

else // có 2 con

searchStandFor(p, T->right);

delete p;

return 1;

}

if(T->info<x) return Delete(T->right, x);

if(T->info>x) return Delete(T->left, x);

}

// ham quan ly chuc nang

void Choice(int x)

{

int it;

do{

//menu

switch(it)

{

case 1:

tree\_empty();

break;

case 2:

nhapx(x);

InsertNode(root, x);

break;

case 3:

nhapx(x);

if(search(root, x) != NULL)

cout<<x<<" co xuat hien trong danh sach

tai Node:"<<search(root, x)<<endl;

else

cout<<x<<" khong co xuat hien trong danh

sach"<<endl;

break;

case 4:

if(Delete(root,x) != 0)

cout<<"Gia tri da duoc xoa thanh

cong"<<endl;

else

cout<<"Gia tri khong duoc xoa thanh

cong"<<endl;

break;

case 5:

cout<<"Duyet cay NLR"<<endl;

duyetNLR(root);

break;

case 6:

cout<<"Duyet cay LNR"<<endl;

duyetLNR(root);

break;

case 7:

cout<<"Duyet cay NLR"<<endl;

duyetNLR(root);

break;

default:

cout<<"Ban da chon chuc nang thoat chuong

trinh."<<endl;

break;

}

}while(it>=1 && it<=7);

}

int main()

{

int x=0;

Choice(x);

system("pause");

return 0;

}