// Listdon\_dathuc.cpp : Defines the entry point for the console application.

//

#include "stdio.h"

#include "conio.h"

#include "stdlib.h"

typedef struct cautrucdonthuc

{

int heso;

int somu;

}donthuc;

typedef struct cautrucdon

{

donthuc info;

struct cautrucdon \*next;

}NODE;

typedef struct dathuc

{

NODE \*head,\*tail;

}LIST;

void menu()

{

printf("\nO: thoat");

printf("\n1: nhap cac don thuc");

printf("\n2: Xuat da thuc");

printf("\n3: Them don thuc vao dau danh sach");

printf("\n4: Them don thuc vao cuoi danh sach");

printf("\n5: Rut gon");

printf("\n6: Sap xep da thuc");

printf("\n7: Cong 2 da thuc");

printf("\n9: Tru 2 da thuc");

printf("\n10: Nhan 2 da thuc");

}

void khoitaods(LIST &L)

{

L.head=L.tail=NULL;

}

NODE \*taophantu(donthuc x)

{

NODE \*p;

p=new NODE;

if(p==NULL)

{

printf("\nKhong du bo nho!");

exit(1);

}

else

{

p->info=x;

p->next=NULL;

}

return p;

}

void themdau(LIST &L, donthuc x)

{

NODE \*moi;

moi=taophantu(x);

if(L.head==NULL)

L.head=L.tail=moi;

else

{

moi->next=L.head;

L.head=moi;

}

}

void themcuoi(LIST &L, donthuc x)

{

NODE \*moi;

moi=taophantu(x);

if(L.head==NULL)

L.head=L.tail=moi;

else

{

L.tail->next=moi;

L.tail=moi;

}

}

void nhapDonthuc(donthuc &x)

{

int tam;

printf("\nNhap he so: ");

scanf("%d",&tam);

x.heso=tam;

printf("Nhap so mu: ");

scanf("%d",&tam);

x.somu=tam;

}

void nhapds(LIST &L)

{

NODE \*p;

donthuc x;

p=L.head;

printf("\nNhap -1 de ket thuc");

int i=0;

do

{

printf("\nNhap phan tu thu %d: ",i+1);

nhapDonthuc(x);

themcuoi(L,x);

i++;

}

while(x.somu>0);

}

void xuatDonthuc(donthuc x)

{

if(x.somu==0)

printf(" %d ",x.heso);

else

{

if(x.somu==1)

printf(" %dx +",x.heso);

else

printf(" %dx^%d +",x.heso,x.somu);

}

}

void xuatds(LIST L)

{

NODE \*p;

p=L.head;

while(p!=NULL)

{

xuatDonthuc(p->info);

p=p->next;

}

printf("\n");

}

void xoa(LIST &L, NODE \*p)

{

NODE \*q;

q=L.head;

if (q==p)

{

L.head=p->next;

delete p;

}

while(q&&(q->next!=p))

q=q->next;

if(q!=NULL)

{

if(p==L.tail)

L.tail=q;

q->next=p->next;

delete p;

}

}

LIST rutgon(LIST &L)

{

NODE \*p,\*q;

p=L.head;

q=p->next;

while(q!=NULL)

{

if(q->info.somu==p->info.somu)

{

p->info.heso=p->info.heso+q->info.heso;

p->next=q->next;

xoa(L,q);

q=q->next;

}

else

{

q=q->next;

p=p->next;

}

}

return L;

}

void doicho(donthuc &x, donthuc &y)

{

donthuc tam;

tam=x;

x=y;

y=tam;

}

void sapxep(LIST &L)

{

NODE \*p,\*q,\*vt;

p=L.head;

while(p!=L.tail)

{

vt=p;

q=p->next;

while(q)

{

if(q->info.somu>vt->info.somu)

vt=q;

q=q->next;

}

doicho(p->info,vt->info);

p=p->next;

}

}

LIST cong2dt(LIST L1, LIST L2)

{

LIST L;

if(L1.head==NULL)

{

L.head=L2.head;

L.tail=L2.tail;

}

else

{

if(L2.head==NULL)

{

L.head=L1.head;

L.tail=L1.tail;

}

else

{

L.head=L1.head;

L1.tail->next=L2.head;

L.tail=L2.tail;

}

}

sapxep(L);

rutgon(L);

return L;

}

LIST tru2dt(LIST L1, LIST L2)

{

LIST L;

NODE \*p;

p=L2.head;

while(p)

{

p->info.heso=p->info.heso\*-1;

p=p->next;

}

L=cong2dt(L1,L2);

return L;

}

donthuc nhandonthuc(NODE \*p, NODE \*q)

{

donthuc x;

x.heso=p->info.heso\*q->info.heso;

x.somu=p->info.somu+q->info.somu;

return x;

}

LIST nhan2dt(LIST L1, LIST L2)

{

donthuc x;

LIST L;

khoitaods(L);

NODE \*p,\*q;

p=L1.head;

while(p)

{

q=L2.head;

while(q)

{

x=nhandonthuc(p,q);

themdau(L,x);

q=q->next;

}

p=p->next;

}

sapxep(L);

rutgon(L);

return L;

}

int main()

{

int c;

donthuc x,y;

LIST L,L1,L2;

khoitaods(L);

do

{

menu();

printf("\nChon chuc nang can thuc hien: ");

scanf("%d",&c);

switch(c)

{

case 0:

exit (1);

case 1:

khoitaods(L);

nhapds(L);

break;

case 2:

printf("\n");

xuatds(L);

break;

case 3:

printf("\nNhap don thuc can them: ");

nhapDonthuc(x);

themdau(L,x);

break;

case 4:

printf("\nNhap don thuc can them: ");

nhapDonthuc(x);

themcuoi(L,x);

break;

case 5:

rutgon(L);

break;

case 6:

sapxep(L);

break;

case 7:

khoitaods(L);

khoitaods(L1);

khoitaods(L2);

printf("\nNhap da thuc L1: ");

nhapds(L1);

printf("\nNhap da thuc L2: ");

nhapds(L2);

L=cong2dt(L1,L2);

break;

case 8:

khoitaods(L);

khoitaods(L1);

khoitaods(L2);

printf("\nNhap da thuc L1: ");

nhapds(L1);

printf("\nNhap da thuc L2: ");

nhapds(L2);

L=tru2dt(L1,L2);

break;

case 9:

khoitaods(L);

khoitaods(L1);

khoitaods(L2);

printf("\nNhap da thuc L1: ");

nhapds(L1);

printf("\nNhap da thuc L2: ");

nhapds(L2);

L=nhan2dt(L1,L2);

break;

default:

printf("\nSai chuc nang!!!");

break;

}

}

while (1);

return 1;

}

#include<stdio.h>

#include<conio.h>

#include<alloc.h>

#include<string.h>

typedef struct SV{

char MSSV[10];

char HoTen[100];

char NgaySinh[20];

char DiaChi[100];

float Diem;

};

typedef struct Node{

SV \* info;

Node \*next;

};

typedef struct List{

Node \*pHead;

Node \*pTail;

};

void KhoiTao\_List(List \*L){

L->pHead=NULL;

L->pTail=NULL;

}

SV \* KhoiTao\_SinhVien(){

SV \* sv = new SV;

// sv=(SV)malloc(sizeof(SV));

float temp;

printf("\n Nhap ma so sinh vien:");gets(sv->MSSV);

printf("\n Nhap ho ten: ");gets(sv->HoTen);

printf("\n Nhap ngay sinh: ");gets(sv->NgaySinh);

printf("\n Nhap dia chi: ");gets(sv->DiaChi);

printf("\n Nhap diem: ");scanf("%f",&temp);

sv->Diem=temp;

//flushall();

return sv;

}

Node \* KhoiTao\_Node(SV \* pt){

// Node \*p=(Node)malloc(sizeof(Node));

Node\*p=new Node;

if(p==NULL)

printf("Khong lam");

else{

p->info=pt;

p->next=NULL;

}

return p;

}

void InDanhSach(List \* L){

Node \*p= L->pHead;

while (p!=NULL){

printf("\n Ma so sv: %s",p->info->MSSV);

printf("\t Ho ten sv: %s",p->info->HoTen);

printf("\t Ngay sinh: %s",p->info->NgaySinh);

printf("\t Dia chi: %s",p->info->DiaChi);

printf("\t Diem: %.2f",p->info->Diem);

p=p->next;

printf("-----> \n");

}

}

void chencuoi(List \*L)

{

SV \*t=KhoiTao\_SinhVien();

Node \* p=KhoiTao\_Node(t);

if(L->pHead==NULL){

L->pHead=L->pTail=p;

}

else{

L->pTail->next=p;

L->pTail = p;

}

}

void sapxep\_theodiem(List\*L){

Node \*p1,\*p2;

SV \*temp;

for(p1=L->pHead;p1!=NULL;p1=p1->next)

for(p2=p1->next;p2!=NULL;p2=p2->next)

if(p1->info->Diem>p2->info->Diem){

temp=p1->info;

p1->info=p2->info;

p2->info=temp;

}

}

void xoacuoi(List \* L){

Node \*p= L->pHead;

if(L->pHead==NULL||L->pTail==NULL)

printf("\nKhong lam");

else{

while(p->next!=L->pTail && p!=NULL)

p=p->next;

if(L->pHead==L->pTail)

L->pHead=L->pTail=NULL;

else{

L->pTail=p;

delete(p->next);

L->pTail->next=NULL;

}

}

}

void main(){

clrscr();

int n;

List L;

printf("\n Nhap vao so sv: ");

scanf("%d",&n);

KhoiTao\_List(&L);

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

printf("\n\nNhap thong tin sinh vien thu %d :\n\n",i);

chencuoi(&L);

}

printf("\nDanh sach sinh vien vua nhap");

InDanhSach(&L);

sapxep\_theodiem(&L);

printf("\nDanh sach sau khi sap xep");

InDanhSach(&L);

printf("\n\nDanh sach sau khi xoa cuoi");

xoacuoi(&L);

InDanhSach(&L);

getch();

}

#include<stdio.h>

#include<conio.h>

#include<alloc.h>

#include<string.h>

typedef struct SV{

char MSSV[10];

char HoTen[100];

char NgaySinh[20];

char DiaChi[100];

float Diem;

};

typedef struct Node{

SV \* info;

Node \*next;

};

typedef struct List{

Node \*pHead;

Node \*pTail;

};

void KhoiTao\_List(List \*L){

L->pHead=NULL;

L->pTail=NULL;

}

SV \* KhoiTao\_SinhVien(){

SV \* sv = new SV;

// sv=(SV)malloc(sizeof(SV));

float temp;

printf("\n Nhap ma so sinh vien:");gets(sv->MSSV);

printf("\n Nhap ho ten: ");gets(sv->HoTen);

printf("\n Nhap ngay sinh: ");gets(sv->NgaySinh);

printf("\n Nhap dia chi: ");gets(sv->DiaChi);

printf("\n Nhap diem: ");scanf("%f",&temp);

sv->Diem=temp;

//flushall();

return sv;

}

Node \* KhoiTao\_Node(SV \* pt){

// Node \*p=(Node)malloc(sizeof(Node));

Node\*p=new Node;

if(p==NULL)

printf("Khong lam");

else{

p->info=pt;

p->next=NULL;

}

return p;

}

void InDanhSach(List \* L){

Node \*p= L->pHead;

while (p!=NULL){

printf("\n Ma so sv: %s",p->info->MSSV);

printf("\t Ho ten sv: %s",p->info->HoTen);

printf("\t Ngay sinh: %s",p->info->NgaySinh);

printf("\t Dia chi: %s",p->info->DiaChi);

printf("\t Diem: %.2f",p->info->Diem);

p=p->next;

printf("-----> \n");

}

}

void chencuoi(List \*L)

{

SV \*t=KhoiTao\_SinhVien();

Node \* p=KhoiTao\_Node(t);

if(L->pHead==NULL){

L->pHead=L->pTail=p;

}

else{

L->pTail->next=p;

L->pTail = p;

}

}

void sapxep\_theodiem(List\*L){

Node \*p1,\*p2;

SV \*temp;

for(p1=L->pHead;p1!=NULL;p1=p1->next)

for(p2=p1->next;p2!=NULL;p2=p2->next)

if(p1->info->Diem>p2->info->Diem){

temp=p1->info;

p1->info=p2->info;

p2->info=temp;

}

}

void xoacuoi(List \* L){

Node \*p= L->pHead;

if(L->pHead==NULL||L->pTail==NULL)

printf("\nKhong lam");

else{

while(p->next!=L->pTail && p!=NULL)

p=p->next;

if(L->pHead==L->pTail)

L->pHead=L->pTail=NULL;

else{

L->pTail=p;

delete(p->next);

L->pTail->next=NULL;

}

}

}

void main(){

clrscr();

int n;

List L;

printf("\n Nhap vao so sv: ");

scanf("%d",&n);

KhoiTao\_List(&L);

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

printf("\n\nNhap thong tin sinh vien thu %d :\n\n",i);

chencuoi(&L);

}

printf("\nDanh sach sinh vien vua nhap");

InDanhSach(&L);

sapxep\_theodiem(&L);

printf("\nDanh sach sau khi sap xep");

InDanhSach(&L);

printf("\n\nDanh sach sau khi xoa cuoi");

xoacuoi(&L);

InDanhSach(&L);

getch();

}

#include "conio.h"

#include "stdio.h"

#include "stdlib.h"

#include "iostream"

#define ESC 27

using namespace std;

typedef struct Data

{

int ts;

int ms;

};

typedef struct tagLNode

{

Data info;

struct tagLNode \*pNext;

}LNode;

typedef struct tagLList

{

LNode \*pHead;

LNode \*pTail;

}LList;

//tao danh sach rong

void create\_list(LList &l)

{

l.pHead = l.pTail = NULL;

}

//tao phan tu co truong la data

LNode \*create\_node\_list(Data x)

{

LNode \*p = new LNode;

if(!p)

{

cout<<"Ko du bo nho!";

exit(1);

}

p->info = x;

p->pNext = NULL;

return p;

}

//them phan tu vao dau danh sach

void add\_head\_list(LList &l,Data x)

{

LNode \*newnode = create\_node\_list(x);

if(!l.pHead)

{

l.pHead = newnode;

l.pTail = l.pHead;

}

else

{

newnode->pNext = l.pHead;

l.pHead = newnode;

}

}

void add\_tail\_list(LList &l, Data x)

{

LNode \*newnode = create\_node\_list(x);

if(!l.pHead)

{

l.pHead = newnode;

l.pTail = l.pHead;

}

else

{

l.pTail->pNext = newnode;

l.pTail = newnode;

}

}

//nhap xau

void input\_list(LList &l )

{

Data x;

LNode \*p;

do

{

cout<<"\nNhap tu so: ";

cin>>x.ts;

cout<<"\nNhap mau so: ";

cin>>x.ms;

if(x.ms == 0 )

break;

p = create\_node\_list(x);

add\_tail\_list(l,p->info);

} while (x.ms != 0);

}

//tim phan tu trong xau

LNode \*search\_node\_list(LList &l , Data x)

{

LNode \*p = l.pHead;

if( !l.pHead )

return NULL;

else

{

while (p != NULL && (p->info.ms != x.ms || p->info.ts != x.ts))

{

p = p->pNext;

}

}

return p;

}

//Them phan tu vao sau phan tu Q

void add\_afterQ(LList &l)

{

Data x, y;

LNode \*p , \*q=NULL;

cout<<"\nNhap phan tu can them phia sau: ";

cin>>x.ts>>x.ms;

cout<<"\nNhap phan tu muon them: ";

cin>>y.ts>>y.ms;

q = search\_node\_list(l,x);

p = create\_node\_list(y);

if(q)

{

p->pNext =q->pNext;

q->pNext = p;

if(l.pTail == q)

l.pTail = p;

}

else

{

cout<<"\nVi tri ko ton tai!";

exit(1);

}

}

//them phan tu x vao truoc phan tu thu k

void add\_at\_list(LList &l , int index ,Data x)

{

int count = 1;//đếm

int found = 0;//kết quả

LNode \*p ,\*q=NULL;

p = l.pHead;

while (p && found == 0)

{

if(count == index-1) // dem = chi so -1

{

q = new LNode; //gan cho q vung nho

q->info = x;

q->pNext = p->pNext;

p->pNext = q;

found = 1;

}

if(index == 1)

{

add\_head\_list(l,x);

found = 1;

}

count ++;

p = p->pNext;

}

if(found == 0)

{

cout<<"\nKo tim thay vi tri can tim!";

}

}

//xoa phan tu dau xau

void remove\_head\_list(LList &l)

{

LNode \*p;

if( !l.pHead )

return ;

p = l.pHead;

l.pHead = l.pHead->pNext;

p->pNext = NULL;

free(p);

}

//xoa phan tu cuoi xau

void remove\_tail\_list(LList &l)

{

LNode \*p = l.pHead;

while (p)

{

if(p->pNext == l.pTail)

break;

p = p->pNext;

}

if( !p && l.pHead )

{

p = l.pHead;

l.pHead = l.pTail = NULL;

free(p);

}

else

{

l.pTail = p;

free(p->pNext);

p->pNext =NULL;

}

}

//xoa phan tu có giá tri = x

void remove\_value\_list(LList &l , Data x)

{

LNode \* p, \*q = NULL;

p = l.pHead;

while (p && (p->info.ts != x.ts && p->info.ms != x.ms))

{

q=p;

p = p->pNext;

}

if( !p ) return;

if(q)

{

q->pNext = p->pNext;

p->pNext = NULL;

delete p;

}

else

{

l.pHead = l.pHead->pNext;

}

}

//xoa phan tu sau phan tu Q

void remove\_afterQ\_list(LList &l, int index)

{

int count = 1;// dem

int found = 0;//ket qua

LNode \*p, \*q=NULL;

p = l.pHead;

while (p && found == 0)

{

if(count == index)//dem = chi so

{

q = p->pNext;

p->pNext = q->pNext;

q->pNext = NULL;

free(q);

found = 1;

}

count ++;

p = p->pNext;

}

if(found == 0)

cout<<"\nKO tim thay phan tu!";

}

//xoa toan bo xau

void remove\_list(LList &l)

{

LNode \*p;

while (l.pHead != NULL)

{

p = l.pHead;

l.pHead = l.pHead->pNext;

delete p;

}

}

//xuat xau

void print\_list(LList l)

{

LNode \*p = l.pHead;

while(p!=NULL)

{

cout<<p->info.ts<<"/"<<p->info.ms<<" ";

p = p->pNext;

}

}

//sao chep xau

void copy\_list(LList l ,LList &cpy)

{

LNode \*p =l.pHead;

while (p)

{

add\_tail\_list(cpy,p->info);

p = p->pNext;

}

}

//dao nguoc xau

void invert\_list(LList l, LList &l1)

{

LNode \*p = l.pHead;

while (p != NULL)

{

add\_head\_list(l1,p->info);

p = p->pNext;

}

}

//bubble sort

void bubblesort\_list(LList &l)

{

if(l.pHead == l.pTail )return;

int swap;

do

{

swap = 0;

for(LNode \*j = l.pHead ; j->pNext != NULL ; j=j->pNext)

{

if(j->info.ts/j->info.ms > j->pNext->info.ts/j->pNext->info.ms)

{

swap = 1;

Data x = j->info;

j->info = j->pNext->info;

j->pNext->info = x;

}

}

} while (swap);

}

//so trung vi

int median\_list(LList l)

{

LList temp;

create\_list(temp);

copy\_list(l,temp);

bubblesort\_list(temp);

int count = 0;

LNode \*p = l.pHead;

while (p)

{

count ++;

p = p->pNext;

}

p = temp.pHead;

for(int i=0 ; i < count/2 ; i++)

{

p = p->pNext;

}

return p->info.ts/p->info.ms;

}

void menu()

{

cout<<"\n0: exit(go ESC)";

cout<<"\n1: Nhap xau";

cout<<"\n2: Xuat xau";

cout<<"\n3: Them pt sau pt Q";

cout<<"\n4: Them pt truoc pt k";

cout<<"\n5: Tim phan tu";

cout<<"\n6: Xoa pt dau xau";

cout<<"\n7: Xoa pt cuoi xau";

cout<<"\n8: Xoa phan tu co gia tri x";

cout<<"\n9: Xoa phan tu sau phan tu x";

cout<<"\n10: Xoa toan bo xau";

cout<<"\n11: Sao chep xau";

cout<<"\n12: Dao nguoc xau";

cout<<"\n13: Bubble Sort";

cout<<"\n14: tim so trung vi";

cout<<"\n";

}

int main()

{

Data x;

int n;

LNode \*p;

LList l;

LList l1;

create\_list(l);

create\_list(l1);

menu();

int c;

do

{

cout<<"\nNhap lua chon: ";

cin>>c;

switch(c)

{

case 0:

exit(1);

break;

case 1:

{

input\_list(l);

break;

}

case 2:

{

print\_list(l);

break;

}

case 3:

{

add\_afterQ(l);

print\_list(l);

break;

}

case 4:

{

int a;

Data b;

cout<<"\nNhap vi tri: ";

cin>>a;

cout<<"\nNhap phan tu can them: ";

cin>>b.ts>>b.ms;

add\_at\_list(l,a,b);

cout<<"\n";

print\_list(l);

break;

}

case 5:

{

cout<<"\nNhap phan tu can tim: ";

cin>>x.ts>>x.ms;

p = search\_node\_list(l,x);

if(p!=NULL)

cout<<"\nTim thay!";

else

cout<<"\nKo Tim thay!";

break;

}

case 6:

{

remove\_head\_list(l);

print\_list(l);

break;

}

case 7:

{

remove\_tail\_list(l);

print\_list(l);

break;

}

case 8:

{

cout<<"\Nhap phan tu muon xoa: ";

cin>>x.ts>>x.ms;

remove\_value\_list(l,x);

print\_list(l);

break;

}

case 9:

{

cout<<"\nNhap vi tri phan tu muon xoa sau: ";

cin>>n;

remove\_afterQ\_list(l,n);

print\_list(l);

break;

}

case 10:

{

remove\_list(l);

print\_list(l);

}

case 11:

{

copy\_list(l,l1);

print\_list(l1);

remove\_list(l);

cout<<"\nList l da xoa :\n ";

print\_list(l);

cout<<"\nList l1 van con: \n";

print\_list(l1);

break;

}

case 12:

{

invert\_list(l,l1);

cout<<"\nList sau khi dao nguoc: \n";

print\_list(l1);

break;

}

case 13:

{

bubblesort\_list(l);

print\_list(l);

break;

}

case 14:

{

cout<<"\nSo trung vi la: "<<median\_list(l);

break;

}

}

} while (1);

return 0;

}

#include <stdio.h>

#include <conio.h>

#include <string.h>

#include <stdlib.h>

typedef struct ngaysinh

{

int ng,th,nam;

}ngaysinh;

typedef struct sinhvien

{

char hoten[50],MSSV[20],gioitinh[4];

ngaysinh ngs;

float DTB;

} data;

typedef struct node

{

data info;

node \*next;

}node;

typedef struct list

{

node \*head,\*tail;

}list;

void creatds(list &l)

{

l.head=l.tail=NULL;

}

node \*getnode(data x)

{

node \*p=new node;

if(p)

{

p->info=x;

p->next=NULL;

}

return p;

}

void addfirst(list &l,node \*p)

{

if(p!=NULL)

{

if(l.head==NULL)

l.head=l.tail=p;

else

{

p->next=l.head;

l.head=p;

}

}

}

node \*addlast(list &l,data x)

{

node \*p=getnode(x);

if(p!=NULL)

{

if(p==NULL)

l.head=l.tail=NULL;

else

{

l.tail->next=p;

l.tail=p;

}

}

return p;

}

node \*addtail(list &l,node \*p)

{

if(p)

{

if(p==NULL)

l.head=l.tail=p;

else

{

l.tail=p->next;

l.tail=p;

}

}

return p;

}

void addafter(list &l,node \*q,data x)

{

node \*p=getnode(x);

if(p==NULL) return;

if(q!=NULL)

{

p->next=q->next;

q->next=p;

if(q!=l.tail) l.tail=p;

}

else addfirst(l,p);

}

void addbefore(list &l,node \*q,data x)

{

node \*p=getnode(x),\*t=l.head,\*k;

if(p==NULL) return;

if(q=l.head)

{

addfirst(l,p);

return;

}

if(q!=NULL)

{

while(t!=q)

{

k=t;

t=t->next;

}

p->next=k->next;

k->next=p;

}

else addfirst(l,p);

}

node \*timms(list l,char \*x)

{

node \*p=l.head;

while(p!=NULL && strcmp(p->info.MSSV,x)!=0)

{

p=p->next;

}

return p;

}

node \*timhoten(list l,char \*x)

{

node \*p=l.head;

while(p!=NULL && strcmp(p->info.hoten,x)!=0)

{

p=p->next;

}

return p;

}

data nhap1sv(data x)

{

printf("MSSV: ");

fflush(stdin);

gets\_s(x.MSSV);

printf("Ho ten: ");

fflush(stdin);

gets\_s(x.hoten);

printf("Gioi tinh: ");

fflush(stdin);

gets\_s(x.gioitinh);

printf("Ngay sinh: ");

scanf\_s("%d %d %d",x.ngs.ng,x.ngs.th,x.ngs.nam);

printf("Diem trung binh: ");

float temp;

scanf\_s("%f",&temp);

x.DTB=temp;

return x;

}

void nhapds(list &l)

{

data x;

while(1)

{

printf("MSSV: ");

fflush(stdin);

gets\_s(x.MSSV);

printf("Ho ten: ");

fflush(stdin);

gets\_s(x.hoten);

printf("Gioi tinh: ");

fflush(stdin);

gets\_s(x.gioitinh);

printf("Ngay sinh: ");

scanf\_s("%d %d %d",x.ngs.ng,x.ngs.th,x.ngs.nam);

printf("Diem trung binh: ");

float temp;

scanf\_s("%f",&temp);

x.DTB=temp;

if(addlast(l,x)==NULL) break;

}

}

void xuatsv(data x)

{

printf("MSSV: %8s",x.MSSV);

printf("Ho ten: %10s",x.hoten);

printf("Gioi tinh: %4s",x.gioitinh);

printf("Ngay sinh: %d/%d/%d ",x.ngs.ng,x.ngs.th,x.ngs.nam);

}

void inds(list l)

{

if(l.head==NULL)

{

printf("Danh sach rong");

return;

}

node \*p=l.head;

while(p)

{

xuatsv(p->info);

p=p->next;

}

}

int removehead(list &l)

{

node \*p=l.head;

if(p!=NULL)

{

l.head=l.head->next;

delete p;

return 1;

}

return 0;

}

int removetail(list &l)

{

node \*p=l.head,\*q;

if(l.head==NULL)

return 0;

if(l.head==l.tail)

return removehead(l);

else

{

while(p!=l.tail)

{

q=p;

p=q->next;

}

l.tail=q;

q->next=NULL;

delete p;

return 1;

}

}

int removeafter(list &l,node \*q)

{

node \*p;

if(q!=NULL)

{

if(q=l.tail)

return 0;

p=q->next;

q->next=p->next;

if(p=l.tail)

l.tail=q;

delete p;

return 1;

}

else

return 0;

}

int removebefore(list &l,node \*q)

{

if(q!=NULL)

{

node \*t=l.head,\*p=t->next,\*w;

if(l.head==NULL || q==l.head) return 0;

if(p==q) return removehead(l);

while(p!=q)

{

w=t;

t=p;

p=p->next;

}

w->next=t->next;

delete t;

return 1;

}

else return 0;

}

void menu()

{

printf("Chuong trinh quan ly sinh vien");

printf("1. Input ");

printf("2. Output ");

printf("3. Add after ");

printf("4. Add before ");

printf("5. Search student ID ");

printf("6. Search student name ");

printf("7. Remove head ");

printf("8. Remove tail ");

printf("9. Remove node ");

printf("10. Remove after ");

printf("11. Remove before ");

printf("0. Exit ");

}

void main()

{

list l;

int chon;

creatds(l);

char a[50];

node \*p;

data x;

do

{

menu();

printf("Chon chuc nang : ");

scanf\_s("%d",&chon);

switch(chon)

{

case 1:nhapds(l); break;

case 2:inds(l); break;

case 3:

{

fflush(stdin);

printf("Them mot sinh vien vao dau DS ");

gets\_s(a);

p=timms(l,a);

x=nhap1sv(x);

if(p!=NULL) addafter(l,p,x);

else addlast(l,x);

break;

}

case 4:

{

fflush(stdin);

printf("Them mot sinh vien vao cuoi DS ");

gets\_s(a);

p=timms(l,a);

x=nhap1sv(x);

if(p!=NULL) addbefore(l,p,x);

else addlast(l,x);

break;

}

case 5:

{

fflush(stdin);

printf("Nhap MSSV can tim ");

gets\_s(a);

p=timms(l,a);

if(p==NULL) printf(" Khong tim thay ");

else xuatsv(p->info);

break;

}

case 6:

{

fflush(stdin);

printf("Nhap ten sinh vien can tim: ");

gets\_s(a);

p=timhoten(l,a);

if(p==NULL) printf(" Khong tim thay ");

else xuatsv(p->info);

break;

}

case 7:

{

if(removehead(l)==1) printf("Da xoa xong");

else printf("Khong xoa duoc ");

break;

}

case 8:

{

if(removetail(l)==1) printf("Da xoa xong");

else printf("Khong xoa duoc");

break;

}

case 9:

{

fflush(stdin);

printf("Nhap MSSV: ");

gets\_s(a);

p=timms(l,a);

if(p==NULL)

{

printf("Khong xoa duoc");

break;

}

p=p->next;

if(p!=NULL) removebefore(l,p);

else removetail(l);

break;

}

case 10:

{

fflush(stdin);

printf("Nhap MSSV can xoa : ");

gets\_s(a);

p=timms(l,a);

if(removeafter(l,p)==1) printf("Da xoa xong");

else printf("Khong xoa duoc");

break;

}

case 11:

{

fflush(stdin);

printf("Nhap MSSV can xoa : ");

gets\_s(a);

p=timms(l,a);

if(removebefore(l,p)==1) printf("Da xoa xong");

else printf("Khong xoa duoc");

break;

}

case 12:

{

return;

}

}

}while(1);

\_getch();

}