TRƯỜNG ĐẠI HỌC GIAO THÔNG VÂN TẢI

KHOA CÔNG NGHỆ THÔNG TIN

---------------o0o---------------



**BÀI TẬP LỚN MÔN HỌC**

**CẤU TRÚC DỮ LIỆU VÀ GIẢI THUẬT**

Giảng viên hướng dẫn: Hoàng Văn Thông

Sinh viên thực hiện: - Lớp CNTT 5-K61

**Hà Nội, tháng 12 năm 2021**

Mục Lục

1. **Đề bài**
2. **Phân tích bài toán**
3. **Cài đặt các lớp và hàm main bằng C++**
4. **Danh sách tài liệu tham khảo**
5. **Đề bài**

* **Đề số 9**

Viết chương trình quản lý danh sách sinh viên bằng **danh sách liên kết kép**, mỗi sinh viên là một cấu trúc gồm:

- Mã số sinh viên

- Họ đệm, tên sinh viên

- Điểm trung bình

- Tên lớp

Chương trình gồm các chức năng sau:

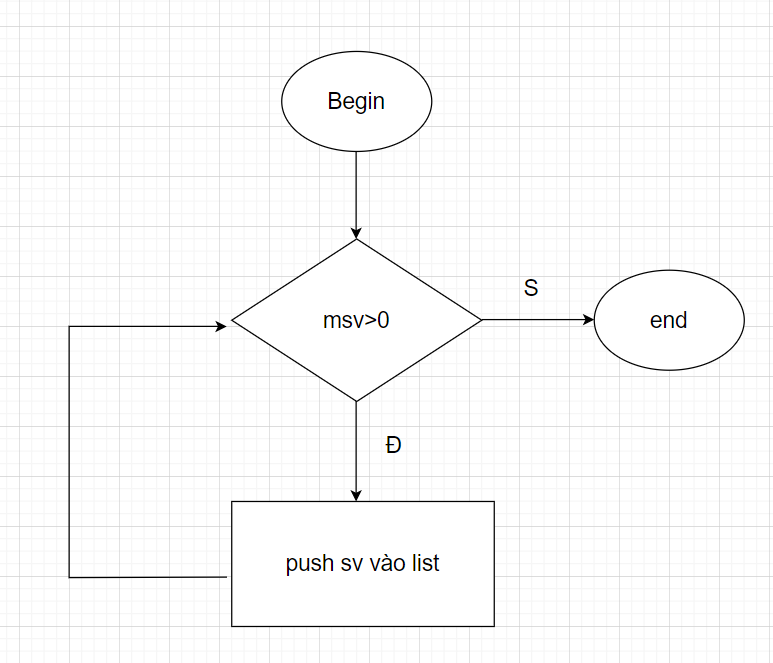
* Tạo danh sách sinh viên: Quá trình nhập danh sách sẽ dừng lại khi nhập mã sv <= 0.
* Thêm 1 sinh viên vào danh sách, vị trí sinh viên thêm vào do người dùng chọn.
* Liệt kê danh sách sinh viên trên màn hình theo thứ tự giảm dần của tổng điểm.
* Căn cứ vào ĐTB của sinh viên, liệt kê tất cả sinh viên đoạt loại giỏi (có ĐTB >= 8.0)
* Nhập vào tên lớp, in ra màn hình danh sách sinh viên của lớp này.
* Nhập vào mã số sinh viên, sau đó in ra vị trí của sinh viên trong danh sách.
* Sắp xếp danh sách sinh viên theo thứ tự tăng dần của tên
* Xoá toàn bộ danh sách những sinh viên có ĐTB < 5.0

1. **Phân tích bài toán**
2. **Yêu cầu bài toán**

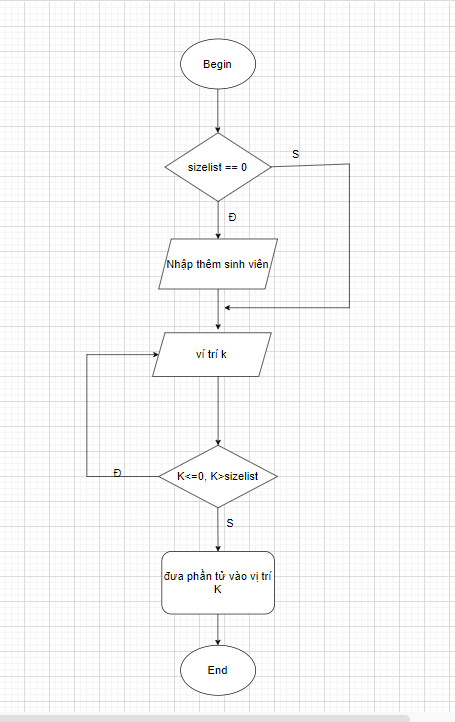
* Xây dựng lớp danh sách liên kết kép
* Viết chương trình quản lí một danh sách liên kết kép, mỗi nút lưu một thông tin sinh viên, chương trình có chức năng như sau:
  + Tạo danh sách sinh viên
  + Thêm 1 sinh viên vào danh sách, vị trí thêm vào tự chọn
  + Liệt kê danh sách sinh viên
  + In ra vị trí của sinh viên
  + Sắp xếp danh sách sinh viên
  + Xóa sinh viên

1. **Các lớp, thuộc tính, phương thức của lớp**

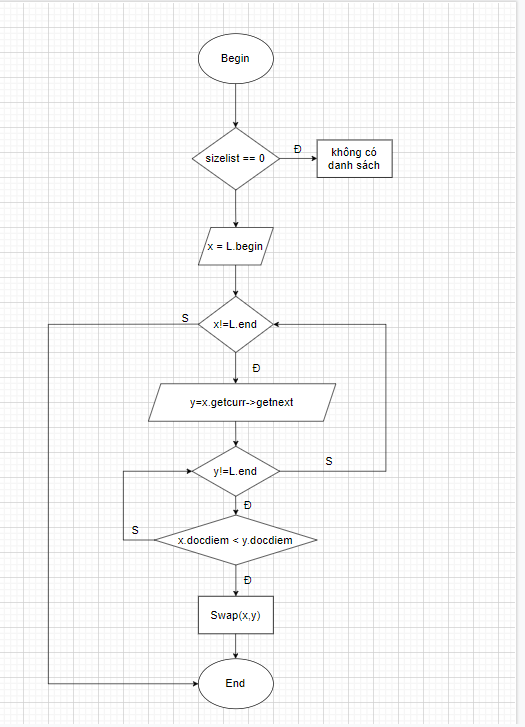
* Xây dựng lớp node
  + Thuộc tính:
    - phần tử elem
    - con trỏ prev
    - con trỏ next
  + Phương thức:
    - getelem: đọc giá trị của phần tử elem
    - getprev: chỉ đến con trỏ prev
    - getnext: chỉ đến con trỏ next
    - setelem: gán giá trị cho phần tử elem
    - setprev: gán địa chỉ cho con trỏ prev
    - setnext: gán địa chỉ cho con trỏ next
* Xây dựng lớp dlist\_iterator và dlist\_reverse\_iterator
  + Thuộc tính: đối tượng curr khai báo từ lớp node
  + Phương thức:
    - getcurr: trả về một đối tượng Iterator
    - Toán tử ! để so sánh hai đối tượng Iterator
    - Toán tử \* để lấy ra phần tử elem của Iterator
    - Toán tử ++ để trả về đối tượng Iterator tiếp theo
* Xây dựng lớp dlist
  + Thuộc tính:
    - Phần tử num: số lượng trong danh sách
    - Phần tử index: vị trí của đối tượng trong danh sách
    - Con trỏ head: đối tượng ở đầu danh sách
    - Con trỏ trail: đối tượng ở cuối danh sách
  + Phương thức:
    - gethead: chỉ đến đối tượng ở đầu danh sách
    - gettrail: chỉ đến đối tượng ở cuối danh sách
    - empty: kiểm tra danh sách rỗng
    - size: trả về số lượng trong danh sách
    - push\_back: thêm một phần tử vào đầu danh sách
    - push\_front: thêm một phần tử vào cuối danh sách
    - insert: thêm một phần tử vào vị trí bất kì
    - front: trả về giá trị của đối tượng ở đầu danh sách
    - back: trả về giá trị của đối tượng ở cuối danh sách
    - pop\_back: xóa phần tử ở đầu danh sách
    - pop\_front: xóa phần tử ở cuối danh sách
    - erase: xóa phần tử ở vị trí bất kì
    - getindex: trả về vị trí của đối tượng trong danh sách
* Xây dựng lớp SINHVIEN
  + Thuộc tính:
    - mssv: mã số sinh viên
    - hoten: họ tên
    - lop: lớp
    - dtb: điểm trung bình
  + Phương thức:
    - Toán tử nhập thông tin sinh viên
    - Toán tử xuất thông tin sinh viên
    - docmssv: trả về mã số sinh viên
    - docdiem: trả về điểm trung bình
    - doclop: trả về lớp
    - docht: trả về họ tên
* Xây dựng lớp CHUCNANG
  + Thuộc tính:
    - Danh sách liên kết kép L từ lớp SINHVIEN
    - Đối tượng sv của lớp SINHVIEN
  + Phương thức:
    - taodanhsach: nhập thông tin sinh viên và đẩy vào danh sách liên kết



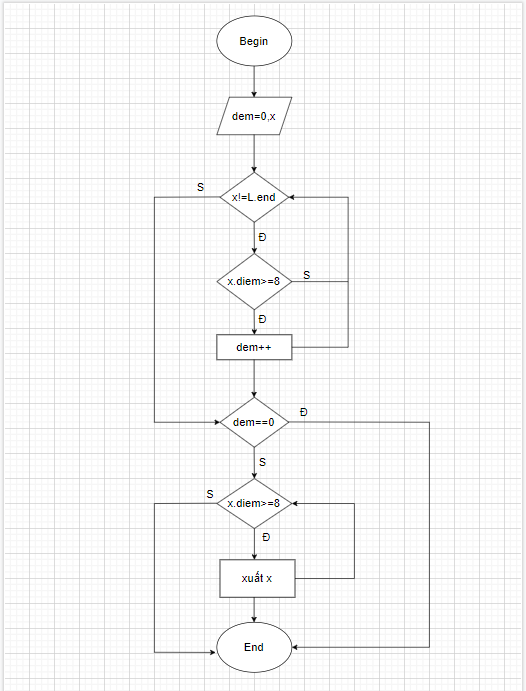
* + - themsinhvien: thêm sinh viên vào vị trí bất kì: nhập vị trí, sau đó nhập thông tin sinh viên và đẩy vào vị trí đó



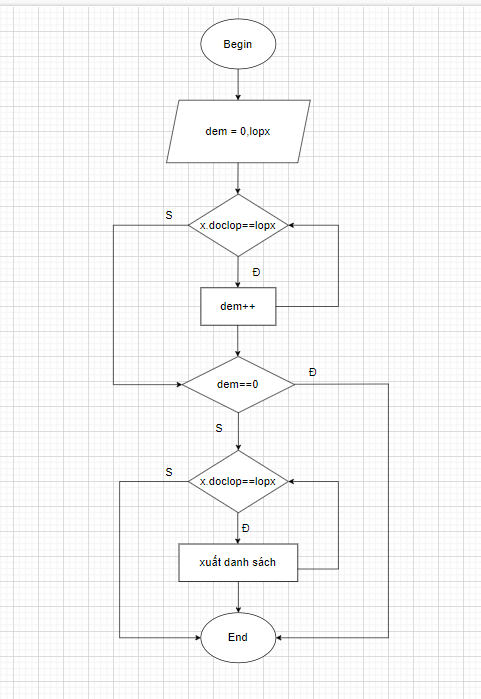
* + - xuatdanhsach: sắp xếp danh sách sinh viên theo thứ tự giảm dần của điểm, sau đó lấy thông tin sinh viên từ danh sách và in ra màn hình



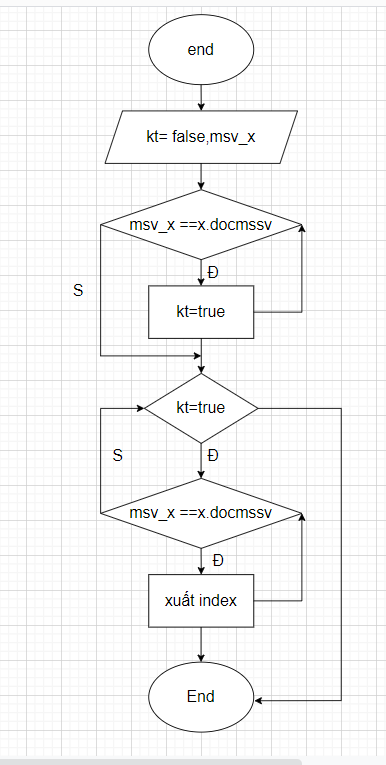
* + - xuatdsgioi: xét tất cả điểm của các sinh viên trong danh sách, sinh viên nào có điểm >= 8.0 thì in ra màn hình



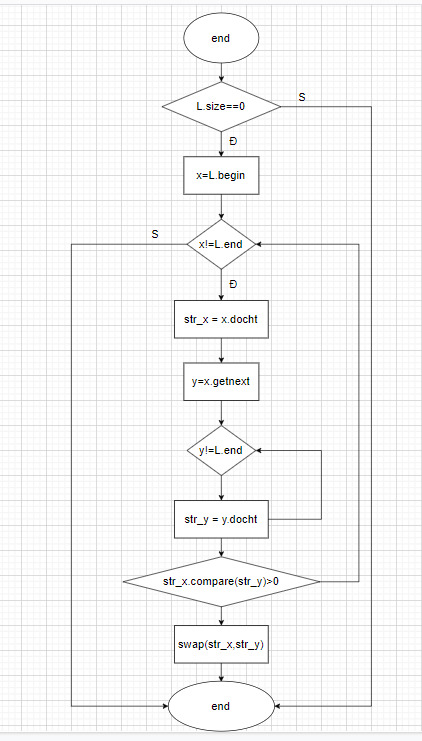
* + - xuatdslop: nhập vào một tên lớp bất kì, xét tất cả tên lớp của các sinh viên trong danh sách, sinh viên nào có tên lớp trùng với tên lớp nhập vào thì in ra màn hình



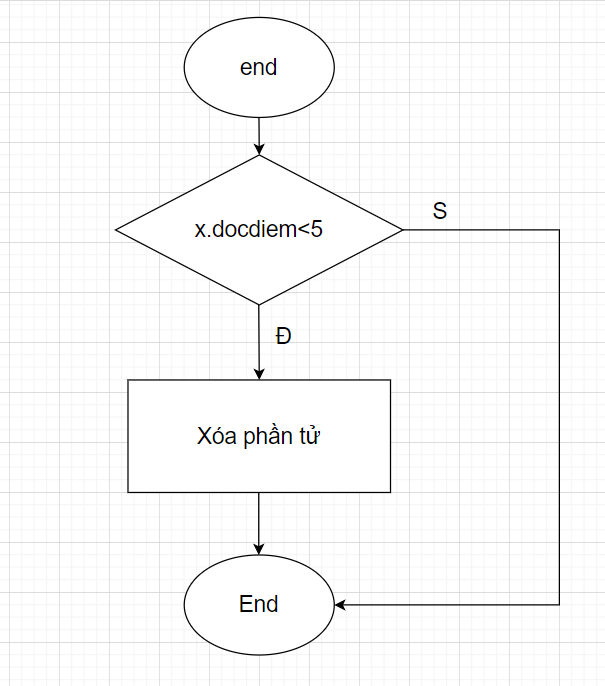
* + - xuatvt: nhập vào một mã số sinh viên bất kì, xét tất cả mã số sinh viên của các sinh viên trong danh sách, sinh viên nào có mã số sinh viên trùng với mã số nhập vào thì in ra màn hình



* + - sapxepten: sắp xếp danh sách sinh viên theo thứ tự tăng dần của họ tên, sau đó lấy thông tin sinh viên từ danh sách và in ra màn hình



* + - xoadanhsach: xét tất cả điểm của các sinh viên trong danh sách, sinh viên nào có điểm < 5.0 thì xóa sinh viên đó khỏi danh sách



1. **Cài đặt các lớp và hàm main bằng C++**
2. **Lớp node**

#include<bits/stdc++.h>

#ifndef \_\_node\_\_cpp\_\_

#define \_\_node\_\_cpp\_\_

using namespace std;

template<class T>

class node

{

private:

T elem;

node<T> \*prev,\*next;

public:

node(T x,node<T> \*L=0,node<T> \*R=0)

{

elem=x;

prev=L;

next=R;

}

T &getelem()

{

return elem;

}

node<T> \*getprev()

{

return prev;

}

node<T> \*getnext()

{

return next;

}

void setelem(T x)

{

elem=x;

}

void setprev(node<T> \*p)

{

prev=p;

}

void setnext(node<T> \*p)

{

next=p;

}

};

#endif

1. **Lớp dlist\_iterator và dlist\_reverse\_iterator**

#include<bits/stdc++.h>

#include"node.cpp"

#ifndef \_\_dlist\_iterator\_\_cpp\_\_

#define \_\_dlist\_iterator\_\_cpp\_\_

using namespace std;

template<class T>

class dlist\_iterator

{

private:

node<T> \*curr;

public:

dlist\_iterator<T>(node<T> \*c=0)

{

curr=c;

}

node<T> \*getcurr()

{

return curr;

}

bool operator!=(dlist\_iterator<T> it)

{

return curr!=it.getcurr();

}

T &operator\*()

{

return curr->getelem();

}

dlist\_iterator<T> operator++(int)

{

dlist\_iterator it=curr;

curr=curr->getnext();

return it;

}

dlist\_iterator<T> operator++()

{

curr=curr->getnext();

return curr;

}

};

template<class T>

class dlist\_reverse\_iterator

{

private:

node<T> \*curr;

public:

dlist\_reverse(node<T> \*c=0)

{

curr=c;

}

node<T> \*getcurr()

{

return curr;

}

bool operator!=(dlist\_reverse\_iterator<T> it)

{

return curr!=it.getcurr();

}

dlist\_reverse\_iterator operator++(int)

{

dlist\_reverse\_iterator it=curr;

curr=curr->getnext();

return it;

}

dlist\_reverse\_iterator operator++()

{

curr=curr->getnext();

return curr;

}

};

#endif

1. **Lớp dlist**

#include<bits/stdc++.h>

#include"node.cpp"

#include"dlist\_iterator.cpp"

#ifndef \_\_dlist\_\_cpp\_\_

#define \_\_dlist\_\_cpp\_\_

using namespace std;

template<class T>

class dlist

{

private:

unsigned int num,index;

node<T> \*head,\*trail;

public:

dlist()

{

num=0;

head=trail=0;

}

dlist(int k,T x)

{

num=0;

head=trail=0;

while(--k)

{

push\_back(x);

}

}

node<T> \*gethead()

{

return head;

}

node<T> \*gettrail()

{

return trail;

}

T &front()

{

return head->getelem();

}

T &back()

{

return trail->getelem();

}

bool empty()

{

return num==0;

}

unsigned size()

{

return num;

}

typedef dlist\_iterator<T> iterator;

iterator begin()

{

return head;

}

iterator end()

{

return NULL;

}

typedef dlist\_reverse\_iterator<T> reverse\_iterator;

reverse\_iterator rbegin()

{

return trail;

}

reverse\_iterator rend()

{

return NULL;

}

void push\_back(T x)

{

if(empty())

head=trail=new node<T>(x);

else

{

trail->setnext(new node<T>(x,trail,0));

trail=trail->getnext();

}

++num;

}

void push\_front(T x)

{

if(empty())

head=trail=new node<T>(x);

else

{

head->setprev(new node<T>(x,0,head));

head=head->getprev();

}

++num;

}

void pop\_back()

{

if(num==0)

return;

if(num==1)

head=trail=0;

else

{

trail=trail->getprev();

trail->setnext(NULL);

}

--num;

}

void pop\_front()

{

if(num==0)

return;

if(num==1)

head=trail=0;

else

{

head=head->getnext();

head->setprev(0);

}

--num;

}

void insert(iterator it,T x)

{

node<T> \*p=it.getcurr();

if(p==head)

return push\_front(x);

else

{

node<T> \*q=p->getprev();

node<T> \*r = new node<T>(x,q,p);

p->setprev(r);

q->setnext(r);

++num;

}

}

void erase(iterator it)

{

if(it.getcurr()==head)

return pop\_front();

else if(it.getcurr()==trail)

return pop\_back();

else

{

node<T> \*L=it.getcurr()->getprev();

node<T> \*R=it.getcurr()->getnext();

L->setnext(R);

R->setprev(L);

}

--num;

}

unsigned getindex(iterator it)

{

index=0;

node<T> \*p=it.getcurr();

node<T> \*q=head;

while(q!=p)

{

++index;

q=q->getnext();

}

return index;

}

};

#endif

1. **Lớp sinhvien**

#include<bits/stdc++.h>

#ifndef \_\_sinhvien\_\_cpp\_\_

#define \_\_sinhvien\_\_cpp\_\_

using namespace std;

class sinhvien

{

private:

int mssv;

string hoten,lop;

float dtb;

public:

sinhvien()

{

mssv=0;

hoten="";

lop="";

dtb=0;

}

friend istream &operator>>(istream &is,sinhvien &sv)

{

cout<<"Nhap ma so sinh vien: ";

is>>sv.mssv;

fflush(stdin);

cout<<"Nhap ho va ten: ";

fflush(stdin);

getline(is,sv.hoten);

cout<<"Nhap ten lop: ";

getline(is,sv.lop);

cout<<"Nhap diem trung binh: ";

is>>sv.dtb;

cout<<"\n";

return is;

}

friend ostream &operator<<(ostream &os,sinhvien sv)

{

os<<"||"<<setw(15)<<left<<sv.mssv;

os<<"||"<<setw(30)<<left<<sv.hoten;

os<<"||"<<setw(15)<<left<<sv.lop;

os<<"||"<<setw(15)<<left<<sv.dtb<<"||";

return os;

}

int docmssv()

{

return mssv;

}

float docdiem()

{

return dtb;

}

string doclop()

{

return lop;

}

string docht()

{

return hoten;

}

};

#endif

1. **Lớp chucnang**

#include<bits/stdc++.h>

#include"dlist.cpp"

#include"node.cpp"

#include"sinhvien.cpp"

#ifndef \_\_chucnang\_\_cpp\_\_

#define \_\_chucnang\_\_cpp\_\_

using namespace std;

class chucnang

{

private:

dlist<sinhvien> L;

sinhvien sv;

public:

void taodanhsach()

{

while(cin>>sv&&sv.docmssv()>0){

L.push\_back(sv);

}

}

void themsinhvien()

{

if(L.size()==0)

{

cout<<"Nhap thong tin sinh vien can them"<<endl;

cin>>sv;

L.push\_back(sv);

}

else

{

int k;

cout<<"Nhap vi tri can them: ";

do

{

cin>>k;

if(k<=0||k>L.size())

cout<<"Vi tri can them khong hop le! Vui long nhap lai: ";

}

while(k<=0||k>L.size());

dlist<sinhvien>::iterator x=L.begin();

for(int i=1;i<k;++i)

{

++x;

}

cout<<"Nhap thong tin sinh vien can them"<<endl;

cin>>sv;

L.insert(x,sv);

}

}

void xuatdanhsach()

{

if(L.size()==0){

cout<<"Khong ton tai sinh vien trong danh sach"<<endl;

}

else

{

for(dlist<sinhvien>::iterator x=L.begin();x!=L.end();++x)

{

for(dlist<sinhvien>::iterator y=x.getcurr()->getnext();y!=L.end();++y){

if((\*x).docdiem()<(\*y).docdiem()){

sinhvien temp = (\*x);

(\*x) = (\*y);

(\*y) = temp;

}

}

}

cout<<"||"<<setw(15)<<left<<"Ma so sinh vien";

cout<<"||"<<setw(30)<<left<<"Ho va ten";

cout<<"||"<<setw(15)<<left<<"Lop";

cout<<"||"<<setw(15)<<left<<"Diem trung binh"<<"||"<<endl;

for(dlist<sinhvien>::iterator x=L.begin();x!=L.end();++x){

cout<<\*x<<endl;

}

}

}

void xuatdsgioi()

{

int dem=0;

for(dlist<sinhvien>::iterator x=L.begin();x!=L.end();++x)

{

if((\*x).docdiem()>=8)

++dem;

}

if(dem==0)

cout<<"Khong co sinh vien doat loai gioi"<<endl;

else

{

cout<<"Co "<<dem<<" sinh vien dat loai gioi";

cout<<endl<<"||"<<setw(15)<<left<<"Ma so sinh vien";

cout<<"||"<<setw(30)<<left<<"Ho va ten";

cout<<"||"<<setw(15)<<left<<"Lop";

cout<<"||"<<setw(15)<<left<<"Diem trung binh"<<"||"<<endl;

for(dlist<sinhvien>::iterator x=L.begin();x!=L.end();++x)

{

if((\*x).docdiem()>=8)

cout<<\*x<<endl;

}

}

}

void xuatdslop()

{

int dem=0;

string lopx;

cout<<"Nhap ten lop: ";

fflush(stdin);

getline(cin,lopx);

for(dlist<sinhvien>::iterator x=L.begin();x!=L.end();++x)

{

if((\*x).doclop()==lopx){

++dem;

}

}

if(dem==0)

cout<<"Khong co sinh vien thuoc lop "<<lopx<<endl;

else

{

cout<<"Co "<<dem<<" sinh vien thuoc lop "<<lopx;

cout<<endl<<"||"<<setw(15)<<left<<"Ma so sinh vien";

cout<<"||"<<setw(30)<<left<<"Ho va ten";

cout<<"||"<<setw(15)<<left<<"Lop";

cout<<"||"<<setw(15)<<left<<"Diem trung binh"<<"||"<<endl;

for(dlist<sinhvien>::iterator x=L.begin();x!=L.end();++x)

{

if((\*x).doclop()==lopx){

cout<<\*x<<endl;

}

}

}

}

void xuatvt()

{

int mssv\_x;

bool kt=false;

cout<<"Nhap ma so sinh vien: ";

cin>>mssv\_x;

for(dlist<sinhvien>::iterator x=L.begin();x!=L.end();++x)

{

if(mssv\_x==(\*x).docmssv())

kt=true;

}

if(kt)

{

cout<<"Vi tri cua sinh vien co ma so sinh vien "<<mssv\_x<<" trong danh sach: ";

for(dlist<sinhvien>::iterator x=L.begin();x!=L.end();++x)

{

if(mssv\_x==(\*x).docmssv())

cout<<L.getindex(x)+1<<" ";

}

}

else

cout<<"Khong ton tai sinh vien co ma so "<<mssv\_x<<" trong danh sach";

}

void sapxepten()

{

if(L.size()==0)

cout<<"Khong ton tai sinh vien trong danh sach"<<endl;

else

{

for(dlist<sinhvien>::iterator x=L.begin();x!=L.end();++x)

{

string str\_x=(\*x).docht();

for(dlist<sinhvien>::iterator y=x.getcurr()->getnext();y!=L.end();++y){

string str\_y=(\*y).docht();

if(str\_x.compare(str\_y)>0){

sinhvien temp = (\*x);

(\*x) = (\*y);

(\*y) = temp;

}

}

}

cout<<"||"<<setw(15)<<left<<"Ma so sinh vien";

cout<<"||"<<setw(30)<<left<<"Ho va ten";

cout<<"||"<<setw(15)<<left<<"Lop";

cout<<"||"<<setw(15)<<left<<"Diem trung binh"<<"||"<<endl;

for(dlist<sinhvien>::iterator x=L.begin();x!=L.end();++x)

{

cout<<\*x<<endl;

}

}

}

void xoadanhsach()

{

for(dlist<sinhvien>::iterator x=L.begin();x!=L.end();++x)

{

if((\*x).docdiem()<5)

L.erase(x);

}

}

void nhanphimbatky()

{

cout<<endl;

system("pause");

system("cls");

}

};

#endif

1. **Hàm main**

#include<bits/stdc++.h>

#include"sinhvien.cpp"

#include"chucnang.cpp"

using namespace std;

int main()

{

chucnang x;

int chon;

luachon:cout<<"Chuong trinh quan ly danh sach sinh vien bang danh sach lien ket kep"<<endl;

cout<<endl<<"1. Tao danh sach sinh vien";

cout<<endl<<"2. Them mot sinh vien vao danh sach";

cout<<endl<<"3. Liet ke danh sach sinh vien theo thu tu giam dan cua diem";

cout<<endl<<"4. Liet ke tat ca cac sinh vien doat loat gioi (co DTB >= 8.0)";

cout<<endl<<"5. In danh sach sinh vien theo ten lop tuy chon";

cout<<endl<<"6. In vi tri cua sinh vien theo ma so sinh vien tuy chon";

cout<<endl<<"7. Sap xep danh sach sinh vien theo thu tu tang dan cua ten";

cout<<endl<<"8. Xoa toan bo danh sach sinh vien co DTB < 5.0";

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

cout<<endl<<"Moi ban chon chuc nang: ";

chon:cin>>chon;

switch(chon)

{

case 1:

system("cls");

x.taodanhsach();

cout<<endl<<"Ban da tao thanh cong danh sach sinh vien!";

x.nhanphimbatky();

goto luachon;

break;

case 2:

system("cls");

x.themsinhvien();

cout<<endl<<"Ban da them thanh cong sinh vien vao danh sach!";

x.nhanphimbatky();

goto luachon;

break;

case 3:

system("cls");

x.xuatdanhsach();

x.nhanphimbatky();

goto luachon;

break;

case 4:

system("cls");

x.xuatdsgioi();

x.nhanphimbatky();

goto luachon;

break;

case 5:

system("cls");

x.xuatdslop();

x.nhanphimbatky();

goto luachon;

break;

case 6:

system("cls");

x.xuatvt();

cout<<endl;

x.nhanphimbatky();

goto luachon;

break;

case 7:

system("cls");

x.sapxepten();

x.nhanphimbatky();

goto luachon;

break;

case 8:

system("cls");

x.xoadanhsach();

cout<<"Ban da xoa sinh vien thanh cong!"<<endl;

x.nhanphimbatky();

goto luachon;

break;

case 9:

system("cls");

cout<<"Da thoat chuong trinh";

return 0;

break;

default:

cout<<"Vui long chon lai chuc nang: ";

goto chon;

break;

}

}

**V. Danh sách tài liệu tham khảo**

- Những trang web hướng dẫn về hướng đối tượng,kế thừa

- <https://www.cplusplus.com/>

- <https://codelearn.io/>

- <https://nguyenvanhieu.vn/>

- Những video record lại của môn cấu trúc dữ liệu trên teams