//Registration Header

#pragma once

#include"subject.h"

#include "student.h"

#include <string>

using namespace std;

class Registration {

private:

string studentID;

string subjectName;

Student \*stu[2];

Subject \*sub[2];

public:

Registration();

Registration(string pstuID, string psubName, int no1, int no2, int no3, int no4);

void displayDetails();

~Registration();

};

//Registration.cpp

#include "registration.h"

#include "student.h"

#include "subject.h"

#include <string>

#include <iostream>

using namespace std;

Registration::Registration()

{

studentID="";

subjectName= "";

stu[0] = new Student();

stu[1] = new Student();

sub[0] = new Subject();

sub[1] = new Subject();

}

Registration::Registration(string pstuID, string psubName, int no1, int no2, int no3, int no4)

{

studentID = pstuID;

subjectName = psubName;

stu[0] = new Student();

stu[1] = new Student();

sub[0] = new Subject();

sub[1] = new Subject();

}

void Registration::displayDetails()

{

}

Registration::~Registration()

{

//Destructor

cout << "Registration Destroyed" << endl;

delete[] sub;

delete[] stu;

}

//Subject Header

#pragma once

#include"exam.h"

#include"workMaterials.h"

#include <string>

using namespace std;

class Subject {

private:

string subjectID;

string subjectName;

Exam\* exm1[2];

Subject\* sub[2];

Supervisor \*sup1[2];

WorkMaterials\* work[2];

public:

Subject();

Subject(string psubjectID, string psubjectName, int no1, int no2, int no3, int no4);

void displaySubject();

~Subject();

};

//Subject.cpp

#include "subject.h"

#include "workMaterials.h"

#include "subject.h"

#include "exam.h"

#include "supervisor.h"

#include <string>

#include <iostream>

using namespace std;

Subject::Subject()

{

subjectID= "";

subjectName= "";

work[0] = new WorkMaterials;

work[1] = new WorkMaterials;

}

Subject::Subject(string psubjectID, string psubjectName, int no1, int no2, int no3, int no4)

{

subjectID= psubjectID;

subjectName= psubjectName;

sup1[0] = new Supervisor();

sup1[1] = new Supervisor();

work[0] = new WorkMaterials();

work[1] = new WorkMaterials();

}

void Subject::displaySubject()

{

}

Subject::~Subject()

{

//Destructor

cout << "Subject destroyed" << endl;

delete[] sup1;

delete[] work;

}