**Bài tập 1**

Nhóm 13:

* 16521169 - Vi Chí Thiện
* 16520147 - Nguyễn Duy Cương
* Phạm Trung Trường - 17520186
* Vũ Minh Thắng - 17521041

**Các bài đề thi OOP các năm trước vào làm thử 2018-2019, 2019-2020, code thử**   
+ Năm 2018-2019

Câu 1 :

1. Phân biệt khái niệm Overload và Override :

* Overload : trong class con tồn tại một method được định nghĩa với cùng tên nhưng khác số lượng tham số với method của class cha.
* Override : trong subclass tồn tại 1 method được định nghĩa với cùng tên và cùng số lượng tham số với method của class cha

1. Phân biệt các kiểu kế thừa :

* Private : chỉ có thể truy cập trong cùng 1 lớp.
* Protected : có thể truy cập trong cùng lớp và trong lớp kế thừa.
* Public : có thể truy cập trong cùng 1 lớp, trong lớp kế thừa và bên ngoài lớp.

Câu 2 :

#include <iostream.h>

Class time {

Private:

Int h,m,s;

Public:

time(int h,int m, int s){ h=m=s= 0};

istream &operator >> (istream &in, time &t)

{

in >> time.h.>>time.m>>time.s;

return in;

}

ostream &operator <<(ostream &out, time &t)

{

out << time.h + ”gio “ + time.m +”phut “ + time.s +”giay.”;

return out;

}

time operator ++()

{

time.s = time.s+1;

if(time.s>=60)

{

time.m+=1;

time.s-=60;

}

if(time.m>=60)

{

time.h+=1;

time.m-=60;

}

if(time.h>=24) time.h-=24;

}

}

Câu 3 :

Main.cpp

#include <iostream>

#include "person.h"

using namespace std;

int main()

{

int n;

int a;

cout<<"Nhap so luong nguoi : ";

cin>>n;

Person\* p = new Person[n];

//Nhap chi so tung nguoi

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

cout<<"Nhap chi so nguoi thu ";

cout<<i<<endl;

p[i].input();

}

cout<<endl;

cout<<"Chon 1 nguoi trong danh sach de xem thong tin!";

cin>>a;

p[a-1].out();

cout<<"Danh sach cac nguoi co nguy co cao can luu y";

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

p[i].check();

if(p[i].isCheck==true){

cout<<"Nguoi thu ";

cout<<i+1<<endl;

}

}

}

#include <iostream>

using namespace std;

class Ability{

public:

int num;

bool good;

Ability(){

num =0;

good = false;

};

void input(){

cin>>num;

}

bool isGood(){

if (num>=70)

{

good = true;

return true;

}

else if(num <=30)

{

good = false;

return false;

}

}

};

ability.h

class O : public Ability{

public:

void out(){

if(good==true)

cout<<"high O"<<endl;

else

cout<<"low O"<<endl;

}

};

class C : public Ability{

public:

void out(){

if(good==true)

cout<<"high C"<<endl;

else

cout<<"low C"<<endl;

}

};

class E : public Ability{

public:

void out(){

if(good==true)

cout<<"high E"<<endl;

else

cout<<"low E"<<endl;

}

};

class A : public Ability{

public:

void out(){

if(good==true)

cout<<"high A"<<endl;

else

cout<<"low A"<<endl;

}

};

class N : public Ability{

public:

void out(){

if(good==true)

cout<<"high N"<<endl;

else

cout<<"low N"<<endl;

}

};

person.h

#include <iostream>

#include "ability.h"

class Person{

public:

O o;

C c;

E e;

A a;

N n;

bool isCheck;

void input(){

cout<<"Nhap chi so O ";

o.input();

cout<<"Nhap chi so C ";

c.input();

cout<<"Nhap chi so E ";

e.input();

cout<<"Nhap chi so A ";

a.input();

cout<<"Nhap chi so N ";

n.input();

}

void out(){

o.out();

c.out();

e.out();

a.out();

n.out();

}

void check(){

if(c.good == false)

isCheck = true;

if(n.good == true)

isCheck = true;

if(e.good == false && n.good == true)

isCheck = true;

}

};