//Multiple Inheritance

#include <iostream>

using namespace std;

class Polygon {

protected:

int width, height;

public:

Polygon (int a, int b) : width(a), height(b) {}

};

class Output {

public:

static void print (int i);

};

void Output::print (int i) {

cout << i << '\n';

}

class Rectangle: public Polygon, public Output {

public:

Rectangle (int a, int b) : Polygon(a,b) {}

int area ()

{ return width\*height; }

};

class Triangle: public Polygon, public Output {

public:

Triangle (int a, int b) : Polygon(a,b) {}

int area ()

{ return width\*height/2; }

};

int main () {

Rectangle rect (4,5);

Triangle trgl (4,5);

rect.print (rect.area());

Triangle::print (trgl.area());

return 0;

}

//Hybrid Inheritance

#include<iostream>

using namespace std;

class stu{ //First base Class//

int id;

char name[20];

public: //If not declared, data members are by default defined as private//

void getstu(){

cout << "Enter stuid, name";

cin >> id >> name;

}

};

class marks: public stu{//derived class//

protected://without this command, data members will not be available next//

int m, p, c;// without ‘protected:’ command, m1, m2, & m3 are private members//

public:

void getmarks(){

cout << "Enter 3 subject marks:";

cin >> m >> p >> c;

}

};

class sports{//Second base class//

protected:

int spmarks;

public:

void getsports(){

cout << "Enter sports marks:";

cin >> spmarks;

}

};

class result : public marks, public sports{//Derived class by hybrid inheritance//

int tot;

float avg;

public :

void show(){

tot=m+p+c;

avg=tot/3.0;

cout << "Total=" << tot << endl;

cout << "Average=" << avg << endl;

cout << "Average + Sports marks =" << avg+spmarks;

}

};

int main(){

result r;//object//

r.getstu();

r.getmarks();

r.getsports();

r.show();

return 0;

}