**Лабораторна робота №8**

**Виконав студент групи ТІ-92 1-го курсу**

**П.І.Б Черноусов Денис Ігорович**

Завдання :

Задача 8.1. Дана следующая, почти законченная программа, добавьте недостающие оператор-функции:

#include using namespace std; class array { int nums[10]; public: array(); void set(int n[10]); void show(); array operator +(array obj2); array operator -(array obj2); array operator ==(array obj2); }; array::array() { for (int i=0; i << nums[i] << ' '; cout << "\n"; } // Fill in operator functions. int main() { array obj1, obj2, obj3; int i[10] = {1, 2, 3, 4, 5, 6, 7, 8 ,9 ,10}; obj1.set(i); obj2.set(i); obj3 = obj1 + obj2; obj3.show(); obj3 = obj1 - obj3; obj3. show(); if (obj1 == obj2) cout << "obj1 equals obj2\n"; else cout << "obj1 does not equal obj2\n"; if (obj1 == obj3) cout << "obj1 equals obj3\n"; else cout << "obj1 does not equal obj3\n"; return 0; }

Перегруженный оператор + должен поэлементно складывать оба операнда. Перегруженный оператор — должен вычитать все элементы правого операнда из элементов левого. Перегруженный оператор == должен возвращать значение true, если все элементы обоих операндов равны, в противном случае он должен возвращать значение false.

Задача 8.2. Переработайте программу Задача 8.1 так, чтобы перегрузить операторы с использованием дружественных функций.

Задача 8.3. Используя класс и функции Задача 8.1, перегрузите оператор ++ с помощью функциичлена класса, а оператор — с помощью дружест венной функции. (Перегрузите только префиксные формы операторов ++ и —).

Задача 8.4. Исследуйте следующую конструкцию:

#include using namespace std; class mybase { int a, b; public: int c; void setab(int i, int j) { a = i; b = j; } void getab(int &i, int &j) { i = a; j = b; } }; class derived1 : public mybase { // ... }; class derived2 : private mybase { // ... }; int main() { derived1 obj1; derived2 obj2; int i, j; // ... }

Какая из следующих инструкций правильна внутри функции main() и почему?

A) obj1.getab(i, j);

B) obj2.getab(i, j);

C) obj1.c = 10;

D) obj2.c = 10;

Задача 8.5. Если бы переменные а и b внутри класса myclass Задача 8.4 стали не закрытыми (по умолчанию), а защищенными членами, изменился бы какой-нибудь из ваших ответов на вопросы этого упражнения? Если да, то почему?

Задача 8.6. В приведенном ниже фрагменте добавьте конструктор для класса myderived. Он должен передать указатель на инициализируемую строку конструктору класса mybase. Кроме того, конструктор myderived() должен инициализировать переменную len длиной строки. #include #include using namespace std; class mybase { char str[80]; public: mybase(char \*s); { strcpy (str, s); } char \*get() { return str; } }; class myderived : public mybase { int len; public: // add myderived() here... int getlen() { return len; } void show() { cout << get () << '\n'; } }; int main() { myderived obо("hello"); obj.show(); cout << obj.getlen() << '\n'; return 0; }

Задача 8.7. Используя следующий фрагмент, создайте соответствующие конструкторы саг() и truck(). Они должны передавать необходимые аргументы объектам класса vehicle. Кроме этого конструктор саг() должен при создании объекта инициализировать переменную passengers, а конструктор truck() - переменную loadlimit.

#include using namespace std; // A base class for various types of vehicle: class vehicle { int num\_wheels; int range; public: vehicle(int w, int r) { num\_wheels = w; range = r; } void showv() { cout << "Wheels: " << num\_wheels << '\n'; cout << "Range: " << range << '\n'; } }; class car : public vehicle { int passengers; public: void show() { showv(); cout << "Passengers: " << passengers << '\n'; } }; class truck : public vehicle { int loadlimit; public: void show() { showv(); cout << "Loadlimit: " << loadlimit << '\n'; } }; int main() { car objc(5, 4, 500); truck objt(3000, 12, 1200); cout << "Car:\n"; objc.show(); cout << "Truck:\n"; objt.show(); return 0; } Для конструкторов саг() и truck() объекты должны объявляться следующим образом: car obj(passengers, wheels, range); truck obj(loadlimit, wheels, range);

Задача 8.8. Используя следующую иерархию классов, создайте конструктор класса С так, чтобы он инициализировал переменную k и передавал аргументы конструкторам А() и В().

#include using namespace std; class A { int i; public: A(int a) { i = a; } }; classB { int j; public: B(int a) { j = a; } }; class C : public A, public B { int k; public: /\* Create C() so that it initializes k and passes arguments to both A() and B() \*/ };

Задача 8.9. Создайте исходный базовый класс building для хранения числа этажей и комнат в здании, а также общую площадь комнат. Создайте производный класс house, который наследует класс building и хранит число ванных комнат и число спален. Кроме этого создайте производный класс office, который наследует класс building и хранит число огнетушителей и телефонов.

Задача 8.10. Дан следующий фрагмент программы, впишите детали, как указано в комментариях: #include using namespace std; class planet { protected: double distance ; // miles from the sun int revolve ; // in days public: planet(double d, int r) { distance = d; revolve = r; } }; class earth : public planet { double circumference; // circumference(окружность) of orbit public: /\* Create earth(double d, int r). Have it pass the distance and days of revolution back to planet. Have it compute the circumference of the orbit. (Hint: circumference = 2r \*3.1416.) \*/ /\* Create a function called show() that displays the information.\*/ }; int main() { earth obj(93000000, 365); obj.show(); return 0; }

Задача 8.11. Иерархия классов из Задача 8.7 с классом vehicle. В программе имеется ошибка и код не форматирован. Найдите ошибку. Подсказка: попытайтесь провести компиляцию и изучите сообщения об ошибках.

# include using namespace std; // A base class for various types of vehicles . class vehicle { int num\_wheels ; int range ; public : vehicle (int w, int r) { num\_wheels = w; range = r; } void showv () { cout << " Wheels : " << num\_wheels << '\n'; cout << " Range : " << range << '\n'; } }; enum motor {gas , electric , diesel }; class motorized : public vehicle { enum motor mtr ; public : motorized ( enum motor m, int w, int r) : vehicle (w, r) { mtr = m; } void showm () { cout << " Motor : "; switch (mtr ) { case gas : cout << "Gas \n"; break ; case electric : cout << " Electric \n"; break ; case diesel : cout << " Diesel \n"; break ; } } }; class road\_use : public vehicle { int passengers ; public : road\_use (int p, int w, int r) : vehicle (w, r) { passengers = p; } void showr () { cout << " Passengers : " << passengers << '\n'; } }; enum steering { power , rack\_pinion , manual }; class car : public motorized , public road\_use { enum steering strng ; public : car ( enum steering s, enum motor m, int w, int r, int p) : road\_use (p, w, r), motorized (m, w, r), vehicle (w, r) { strng = s; } void show () { showv (); showr (); showm (); cout << " Steering : "; switch ( strng ) { case power : cout << " Power \n"; break ; case rack\_pinion : cout << " Rack and Pinion \n"; break ; case manual : cout << " Manual \n"; break ; } } }; int main () { car c(power , gas , 4, 500 , 5); c. show (); return 0; }

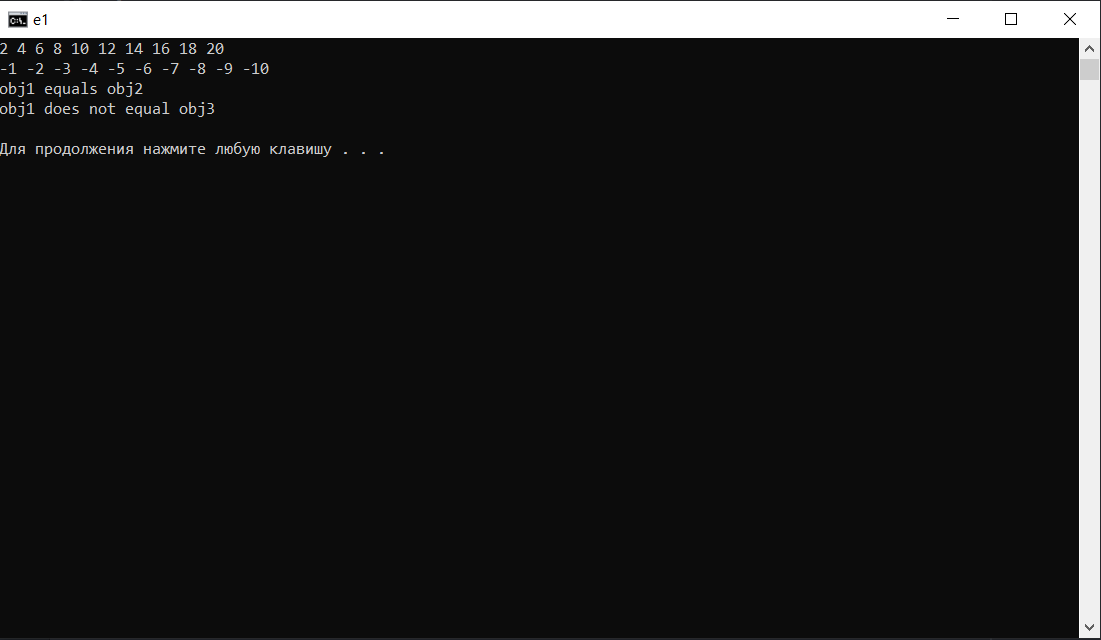
При компиляции программы вы могли увидеть предупреждающее сообщение (или, возможно, сообщение об ошибке), связанное с использованием инструкции switch внутри классов саг и motorised. Почему?

Задача 8.12. Здесь представлена переработанная версия класса coord из предыдущего Практического занятия. Форматирование кода нарушено. Теперь этот класс используется в качестве базового для класса quad, в котором помимо координат хранится номер квадранта, к которому принадлежит точка с этими координатами. Запустите программу и попытайтесь понять и объяснить полученный результат.

#include < iostream>using namespace std; class coord { public: int x, y; // coordinate values coord () { x=0; y=0; } coord (int i, int j) { x=i; y=j; } void get\_xy (int &i, int &j) { i=x; j=y; } coord operator +( coord ob2); coord operator -( coord ob2); coord operator =( coord ob2); }; // Overload + relative to coord class . coord coord :: operator +( coord ob2) { coord temp ; cout << " Using coord operator +() \n"; temp .x = x + ob2 .x; temp .y = y + ob2 .y; return temp ; } // Overload - relative to coord class . coord coord :: operator -( coord ob2) { coord temp ; cout << " Using coord operator -() \n"; temp .x = x - ob2 .x; temp .y = y - ob2 .y; return temp ; } // Overload = relative to coord . coord coord :: operator =( coord ob2) { cout << " Using coord operator =() \n"; x = ob2.x; y = ob2.y; return \* this ; // return the object that is assigned to } class quad : public coord { int quadrant ; public : quad () { x = 0; y = 0; quadrant = 0; } quad (int x, int y) : coord (x, y) { if(x >=0 && y >= 0) quadrant = 1; else if(x =0) quadrant = 2; else if(x << " Point in Quadrant : " << quadrant << '\n'; } quad operator =( coord ob2 ); }; quad quad :: operator =( coord ob2 ) { cout << " Using quad operator =() \n"; x = ob2.x; y = ob2.y; if(x >=0 && y >= 0) quadrant = 1; else if(x =0) quadrant = 2; else if(x << "(o1+o2) X: " << x << ", Y: " << y << "\n"; o3 = o1 - o2; // subtract two objects o3. get\_xy (x, y); o3. showq (); cout << "(o1 -o2) X: " << x << ", Y: " << y << "\n"; o3 = o1; // assign an object o3. get\_xy (x, y); o3. showq (); cout << "(o3=o1) X: " << x << ", Y: " << y << "\n"; return 0; }

Переработайте эту программу, чтобы в ней использовались дружественные оператор-функции.

Завдання 8.1



Код :

#include <iostream >

using namespace std;

class array {

int nums[10];

public:

array();

void set(int n[10]);

void show();

array operator +(array obj2){

array temp;

for (int i = 0; i < 10 ; i++) {

temp.nums[i] = nums[i] + obj2.nums[i];

}

return temp;

}

array operator -(array obj2){

array temp;

for (int i = 0; i < 10 ; i++) {

temp.nums[i] = nums[i] - obj2.nums[i];

}

return temp;

}

bool operator ==(array obj2){

int tr = 0 ;

for (int i = 0; i < 10 ; i++) {

if (nums[i] == obj2.nums[i]) {

tr++;

}

}

return (tr == 10)? true : false;

}

};

array::array() {

for (int i=0; i <10; i++) nums[i] = 0;

}

void array::set(int \*n) {

for (int i=0; i<10; i++) nums[i] = n[i];

}

void array::show() {

for (int i=0; i<10; i++)

cout << nums[i] << ' ';

cout << "\n";

}

// Fill in operator functions.

int main() {

array obj1, obj2, obj3;

int i[10] = {1, 2, 3, 4, 5, 6, 7, 8 ,9 ,10};

obj1.set(i);

obj2.set(i);

obj3 = obj1 + obj2;

obj3.show();

obj3 = obj1 - obj3;

obj3. show();

if (obj1 == obj2)

cout << "obj1 equals obj2\n";

else

cout << "obj1 does not equal obj2\n";

if (obj1 == obj3)

cout << "obj1 equals obj3\n";

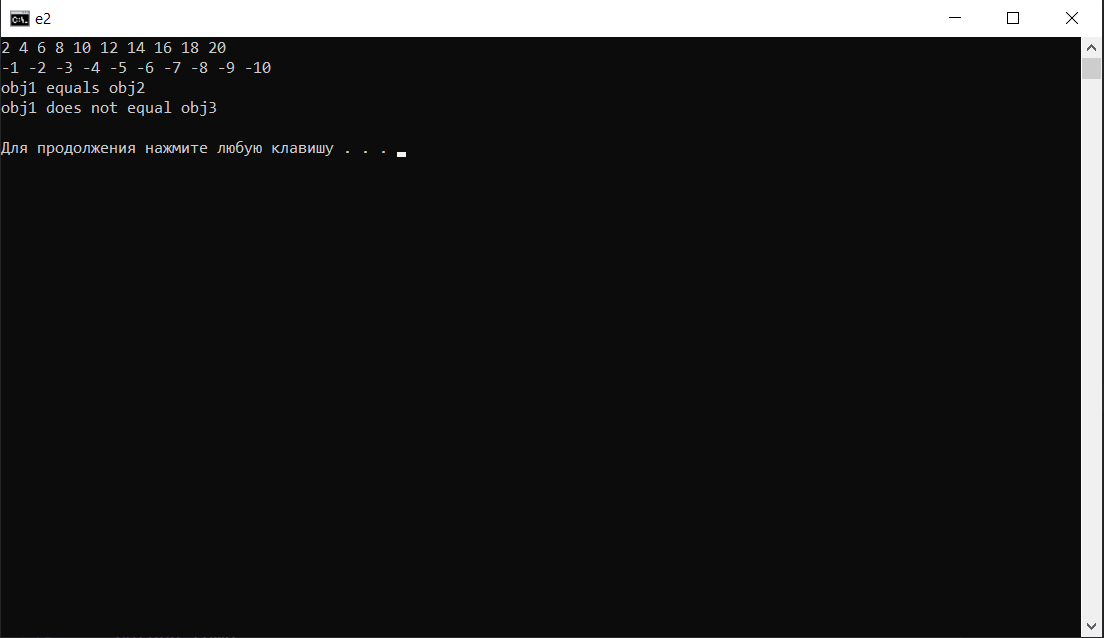
else

cout << "obj1 does not equal obj3\n";

return 0;

}

Завдання 8.2



Код:

#include <iostream >

using namespace std;

class array {

int nums[10];

public:

array();

void set(int n[10]);

void show();

friend array operator +(array obj1 ,array obj2){

array temp;

for (int i = 0; i < 10 ; i++) {

temp.nums[i] = obj1.nums[i] + obj2.nums[i];

}

return temp;

}

friend array operator -(array obj1 ,array obj2){

array temp;

for (int i = 0; i < 10 ; i++) {

temp.nums[i] = obj1.nums[i] - obj2.nums[i];

}

return temp;

}

friend bool operator ==(array obj1 ,array obj2){

int tr = 0 ;

for (int i = 0; i < 10 ; i++) {

if (obj1.nums[i] == obj2.nums[i]) {

tr++;

}

}

return (tr == 10)? true : false;

}

};

array::array() {

for (int i=0; i <10; i++) nums[i] = 0;

}

void array::set(int \*n) {

for (int i=0; i<10; i++) nums[i] = n[i];

}

void array::show() {

for (int i=0; i<10; i++)

cout << nums[i] << ' ';

cout << "\n";

}

// Fill in operator functions.

int main() {

array obj1, obj2, obj3;

int i[10] = {1, 2, 3, 4, 5, 6, 7, 8 ,9 ,10};

obj1.set(i);

obj2.set(i);

obj3 = obj1 + obj2;

obj3.show();

obj3 = obj1 - obj3;

obj3. show();

if (obj1 == obj2)

cout << "obj1 equals obj2\n";

else

cout << "obj1 does not equal obj2\n";

if (obj1 == obj3)

cout << "obj1 equals obj3\n";

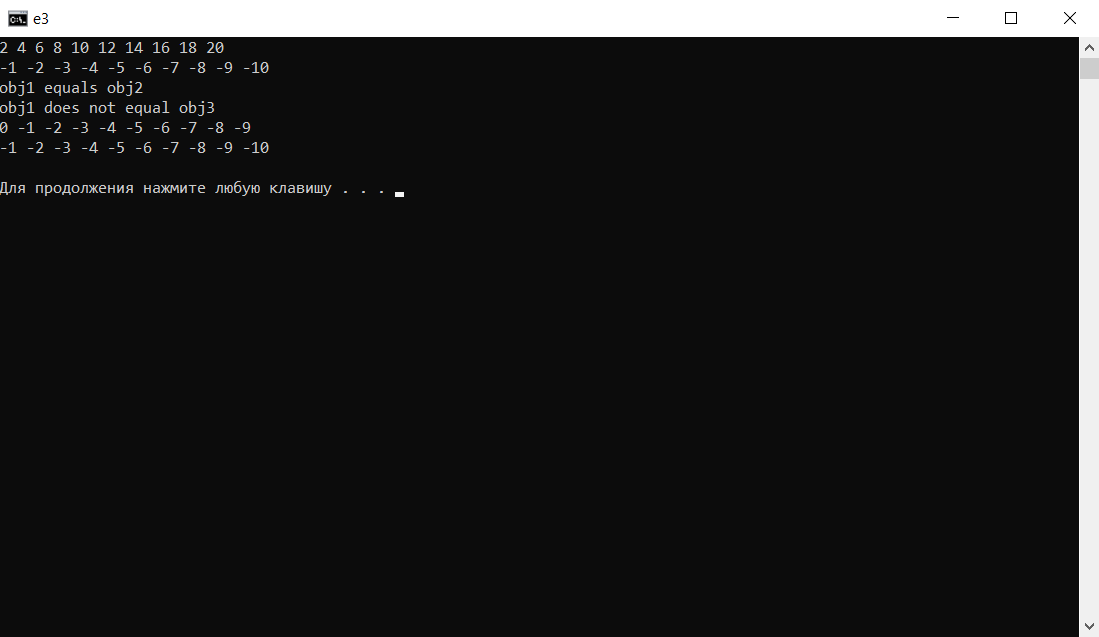
else

cout << "obj1 does not equal obj3\n";

return 0;

}

Завдання 8.3



Код :

#include <iostream >

using namespace std;

class array {

int nums[10];

public:

array();

void set(int n[10]);

void show();

void operator ++(){

for (int i = 0; i < 10 ; i++) {

nums[i] = 1 + nums[i];

}

}

friend void operator --(array &obj2 ){

for (int i = 0; i < 10 ; i++) {

obj2.nums[i] = obj2.nums[i] - 1 ;

}

}

friend array operator +(array obj1 ,array obj2){

array temp;

for (int i = 0; i < 10 ; i++) {

temp.nums[i] = obj1.nums[i] + obj2.nums[i];

}

return temp;

}

friend array operator -(array obj1 ,array obj2){

array temp;

for (int i = 0; i < 10 ; i++) {

temp.nums[i] = obj1.nums[i] - obj2.nums[i];

}

return temp;

}

friend bool operator ==(array obj1 ,array obj2){

int tr = 0 ;

for (int i = 0; i < 10 ; i++) {

if (obj1.nums[i] == obj2.nums[i]) {

tr++;

}

}

return (tr == 10)? true : false;

}

};

array::array() {

for (int i=0; i <10; i++) nums[i] = 0;

}

void array::set(int \*n) {

for (int i=0; i<10; i++) nums[i] = n[i];

}

void array::show() {

for (int i=0; i<10; i++)

cout << nums[i] << ' ';

cout << "\n";

}

// Fill in operator functions.

int main() {

array obj1, obj2, obj3;

int i[10] = {1, 2, 3, 4, 5, 6, 7, 8 ,9 ,10};

obj1.set(i);

obj2.set(i);

obj3 = obj1 + obj2;

obj3.show();

obj3 = obj1 - obj3;

obj3. show();

if (obj1 == obj2)

cout << "obj1 equals obj2\n";

else

cout << "obj1 does not equal obj2\n";

if (obj1 == obj3)

cout << "obj1 equals obj3\n";

else

cout << "obj1 does not equal obj3\n";

++ obj3;

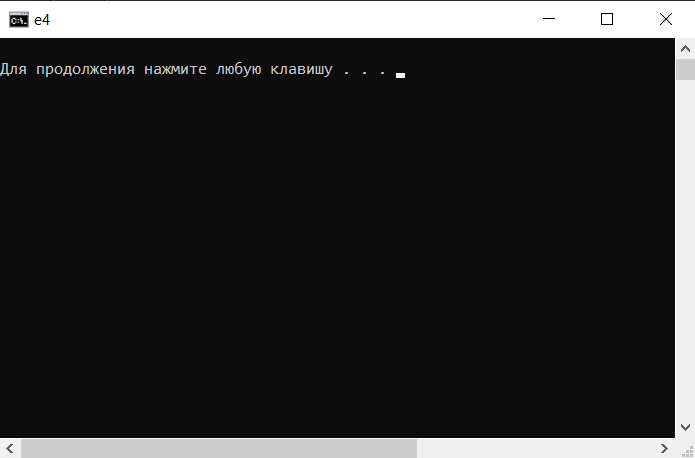
obj3.show();

--obj3;

obj3.show();

}

Завдання 8.4



Код :

#include <iostream >

using namespace std;

class mybase {

int a, b;

public:

int c;

void setab(int i, int j) { a = i; b = j; }

void getab(int &i, int &j) { i = a; j = b; }

};

class derived1 : public mybase {

// ...

};

class derived2 : private mybase {

// ...

};

int main() {

derived1 obj1;

derived2 obj2;

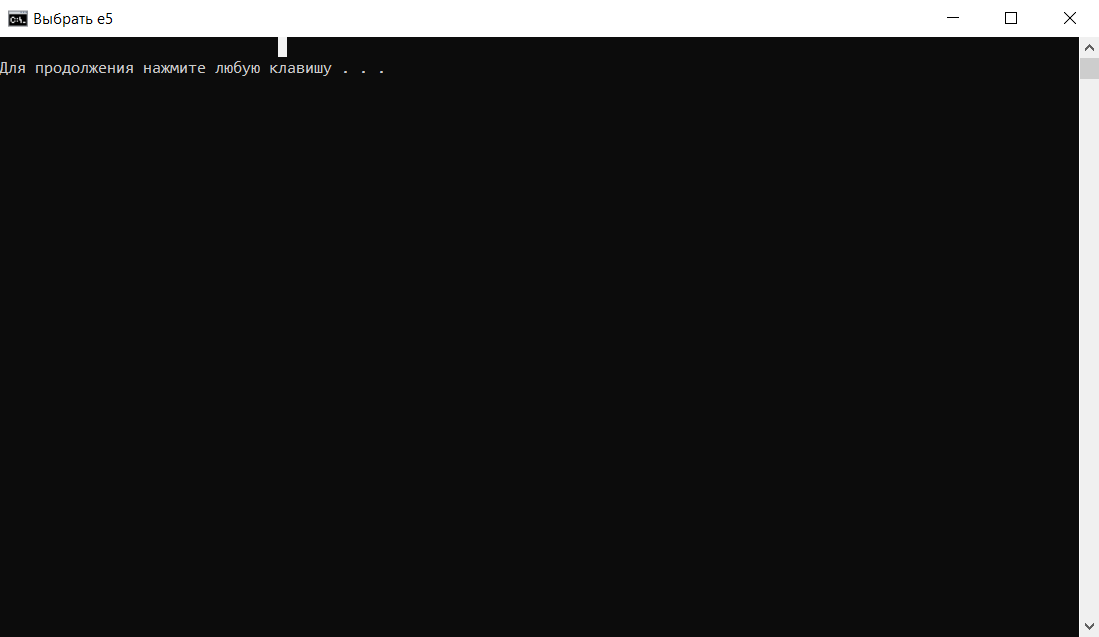
int i = 1, j = 2;

obj1.getab(i,j); // тому що унаслідований public дозоляє , а private -- ні.

obj1.c = 10 ;

}

Завдання 8.5



Код :

#include <iostream >

using namespace std;

class mybase {

protected:

int a, b;

public:

int c;

void setab(int i, int j) { a = i; b = j; }

void getab(int &i, int &j) { i = a; j = b; }

};

class derived1 : public mybase {

// ...

};

class derived2 : private mybase {

// ...

};

int main() {

derived1 obj1;

derived2 obj2;

int i = 1, j = 2;

obj1.getab(i,j);

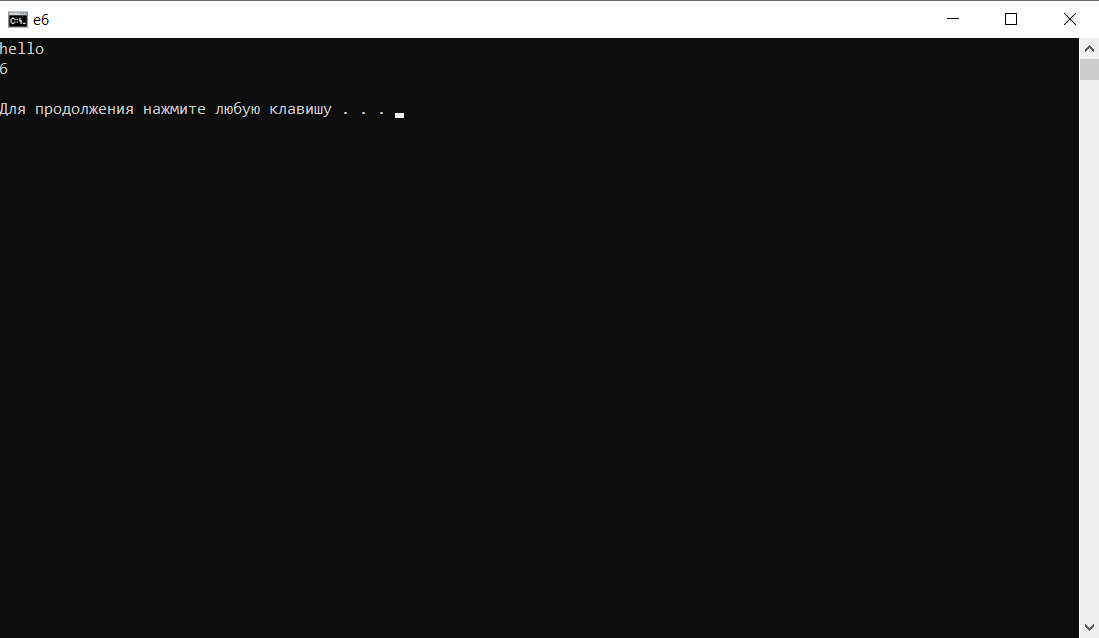
obj1.c = 10 ;

//obj2.getab(i, j); // нічого не зміниться : тип унаслідування private не дозволяє .

//obj2.c = 10 ;

}

Завдання 8.6



Код :

#include <iostream>

#include <cstring>

using namespace std;

class mybase {

char str[80];

public:

mybase(const char \*s){ strcpy (str, s); }

char \*get() { return str; }

};

class myderived : public mybase {

int len;

public:

myderived(const char \*s):mybase(s){

len = strlen(s)+1 ;

}

int getlen() { return len; }

void show() { cout << get () << '\n'; }

};

int main() {

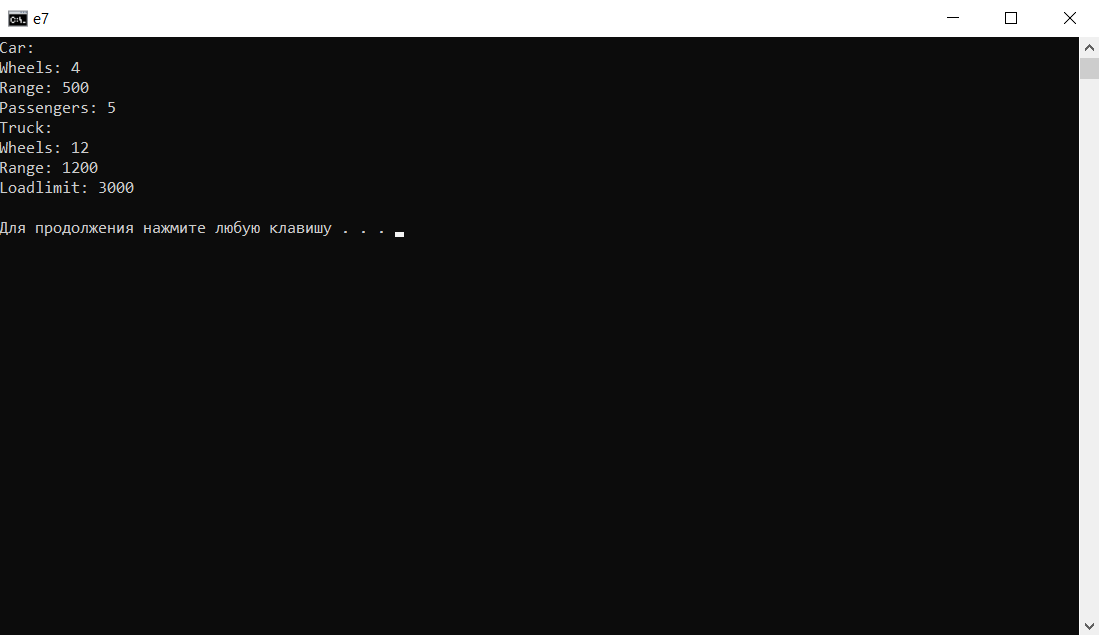
myderived ob("hello");

ob.show();

cout << ob.getlen() << '\n';

}

Завдання 8.7



Код :

#include <iostream >

using namespace std;

class vehicle {

int num\_wheels;

int range;

public:

vehicle(int w, int r) {

num\_wheels = w;

range = r;

}

void showv() {

cout << "Wheels: " << num\_wheels << '\n';

cout << "Range: " << range << '\n';

}

};

class car : public vehicle {

int passengers;

public:

car(int p,int w, int r ):vehicle(w,r){

passengers = p ;

}

void show() {

showv();

cout << "Passengers: " << passengers << '\n';

}

};

class truck : public vehicle {

int loadlimit;

public:

truck(int l,int w, int r):vehicle(w,r){

loadlimit = l ;

}

void show() {

showv();

cout << "Loadlimit: " << loadlimit << '\n';

}

};

int main() {

car objc(5, 4, 500);

truck objt(3000, 12, 1200);

cout << "Car:\n";

objc.show();

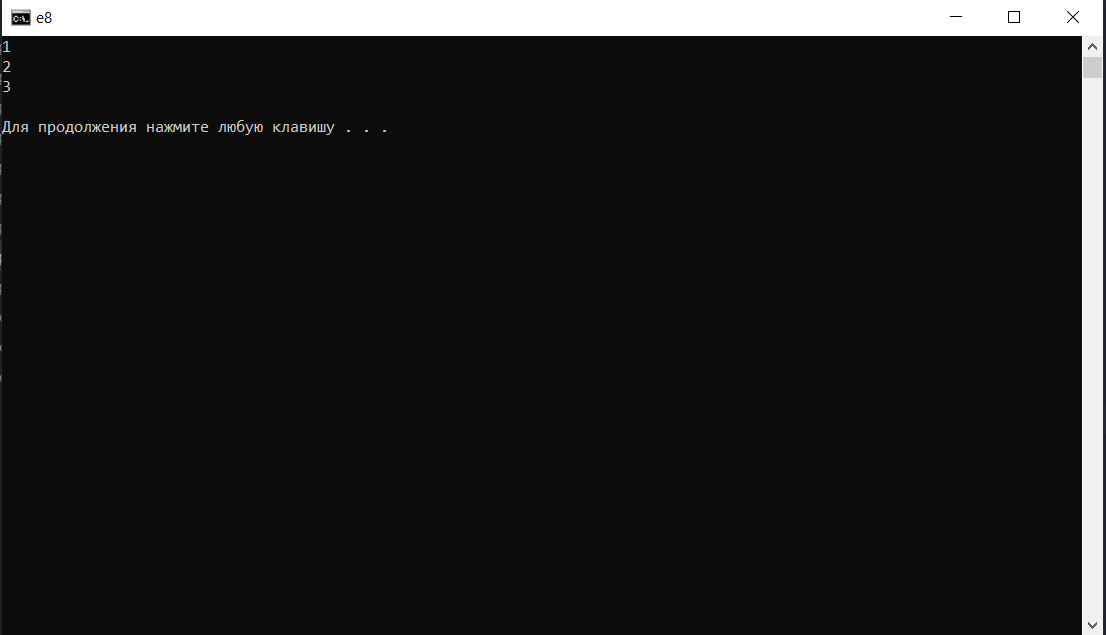
cout << "Truck:\n";

objt.show();

return 0;

}

Завдання 8.8



Код :

#include <iostream >

using namespace std;

class A {

int i;

public:

A(int a) { i = a; }

o1(){

printf("%d\n", i );

}

};

class B {

int j;

public:

B(int a) { j = a; }

o2(){

printf("%d\n", j );

}

};

class C : public A, public B {

int k;

public:

C(int l , int i , int j ) : B(j) , A(i){

k = l ;

}

o3(){

printf("%d\n", k );

o1();

o2();

}

};

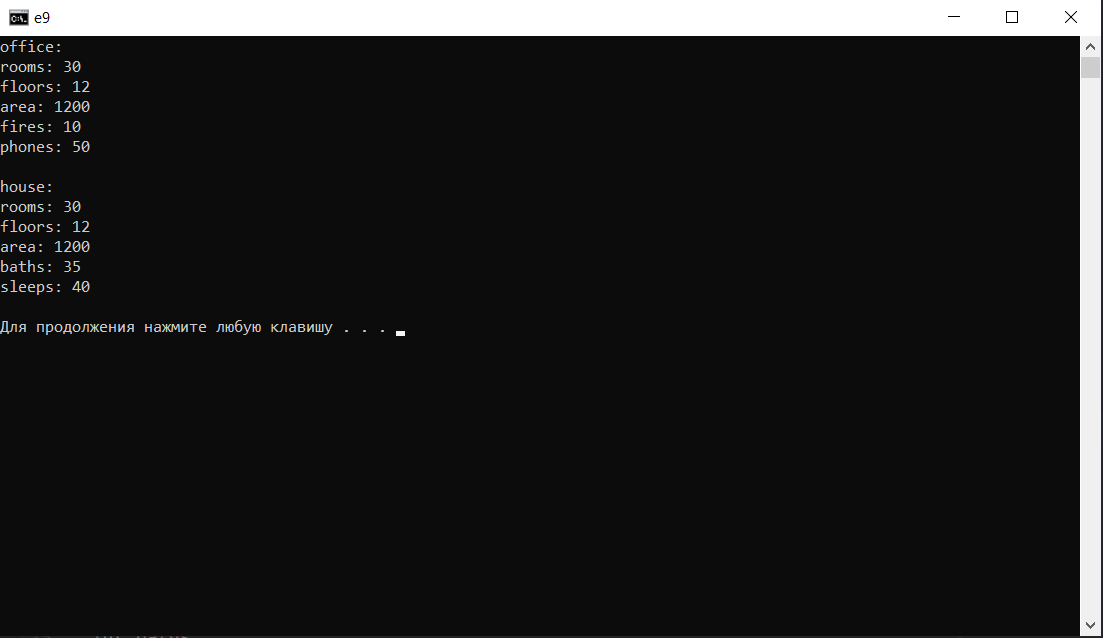
int main (){

C ob(1,2,3);

ob.o3();

}

Завдання 8.9

Код :

#include <iostream >

using namespace std;

class building {

int rooms;

int floors;

int area ;

public:

building(int ro, int fl , int ar ) {

rooms = ro;

floors = fl;

area = ar ;

}

void showv() {

cout << "rooms: " << rooms << '\n';

cout << "floors: " << floors << '\n';

cout << "area: " << area << '\n';

}

};

class house : public building {

int baths ;

int sleeps ;

public:

house(int sl , int ba , int ro , int fl , int ar ):building(ro,fl,ar){

baths = ba ;

sleeps = sl ;

}

void show() {

showv();

cout << "baths: " << baths << '\n';

cout << "sleeps: " << sleeps << '\n';

}

};

class office : public building {

int fires;

int phones ;

public:

office(int fi , int ph, int ro , int fl , int ar ):building(ro,fl,ar){

fires = fi ;

phones = ph;

}

void show() {

showv();

cout << "fires: " << fires << '\n';

cout << "phones: " << phones << '\n';

}

};

int main() {

office obo(10 , 50 ,30, 12, 1200);

house obh(40 , 35 ,30, 12, 1200);

cout << "office:\n";

obo.show();

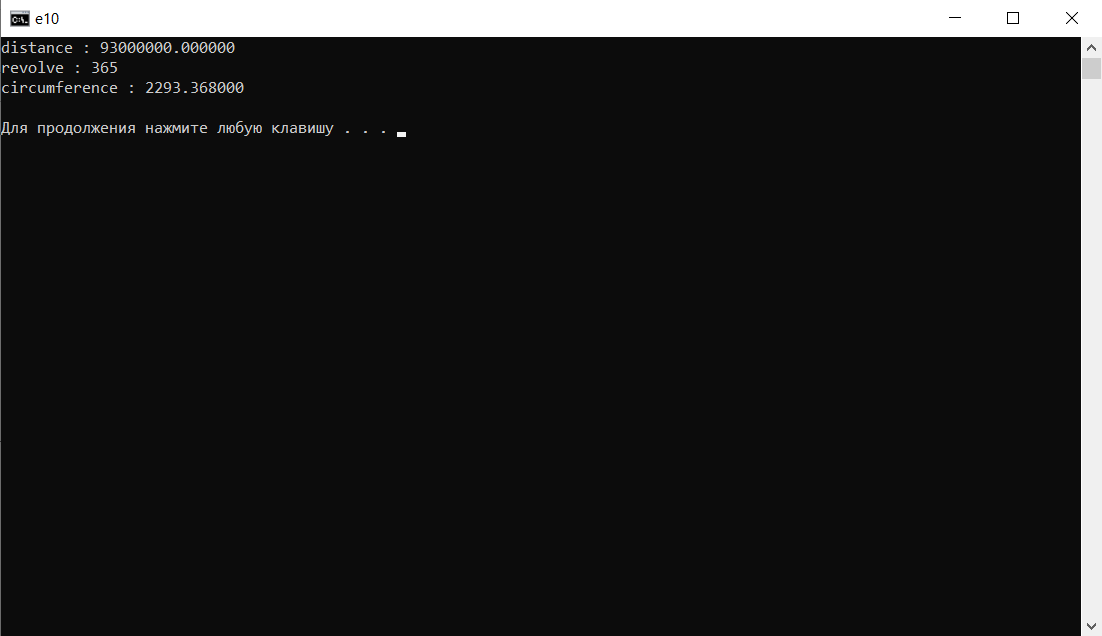
cout << "\nhouse:\n";

obh.show();

return 0;

}

Завдання 8.10



Код :

#include <iostream>

using namespace std;

class planet {

protected:

double distance ; // miles from the sun

int revolve ; // in days

public:

planet(double d, int r) { distance = d; revolve = r; }

};

class earth : public planet {

double circumference; // circumference(окружность) of orbit

public:

earth(double d, int r) : planet(d,r){

circumference = 2 \* r \*3.1416;

}

show(){

printf("distance : %f \nrevolve : %d \ncircumference : %f \n ",distance , revolve , circumference );

}

};

int main() {

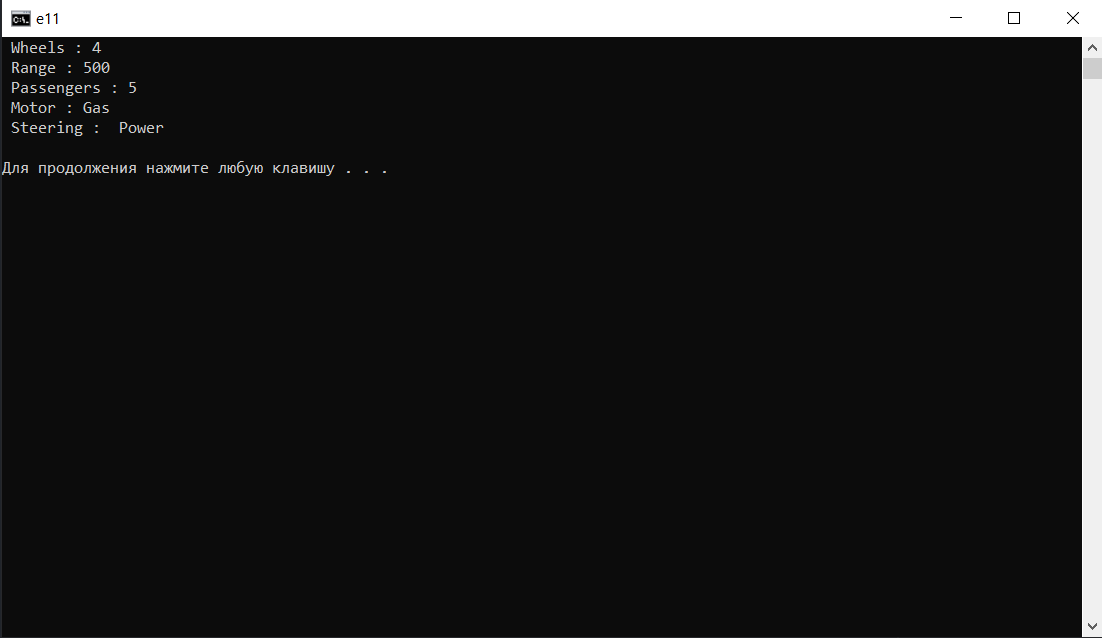
earth obj(93000000, 365);

obj.show();

return 0;

}

Завдання 8.11



Код :

# include <iostream >

using namespace std;

// A base class for various types of vehicles .

class vehicle{

int num\_wheels ;

int range ;

public :

vehicle (int w, int r){

num\_wheels = w;

range = r;

}

void showv (){

cout << " Wheels : " << num\_wheels << '\n';

cout << " Range : " << range << '\n';

}

};

enum motor {gas , electric , diesel };

class motorized {

enum motor mtr ;

public :

motorized ( enum motor m) {

mtr = m;

}

void showm (){

cout << " Motor : ";

switch (mtr ){

case gas : cout << "Gas \n";

break ;

case electric : cout << " Electric \n";

break ;

case diesel : cout << " Diesel \n";

break ;

}

}

};

///

class road\_use {

int passengers ;

public :

road\_use (int p) {

passengers = p;

}

void showr (){

cout << " Passengers : " << passengers << '\n';

}

};

////

enum steering { power , rack\_pinion , manual };

class car : public motorized , public road\_use , public vehicle{

enum steering strng ;

public :

car ( enum steering s, enum motor m, int w, int r, int p) : road\_use (p), motorized (m), vehicle (w, r){

strng = s;

}

void show (){

showv ();

showr ();

showm ();

cout << " Steering : ";

switch ( strng ) {

case power : cout << " Power \n";

break ;

case rack\_pinion : cout << " Rack and Pinion\n";

break ;

case manual : cout << " Manual \n";

break ;

}

}

};

////

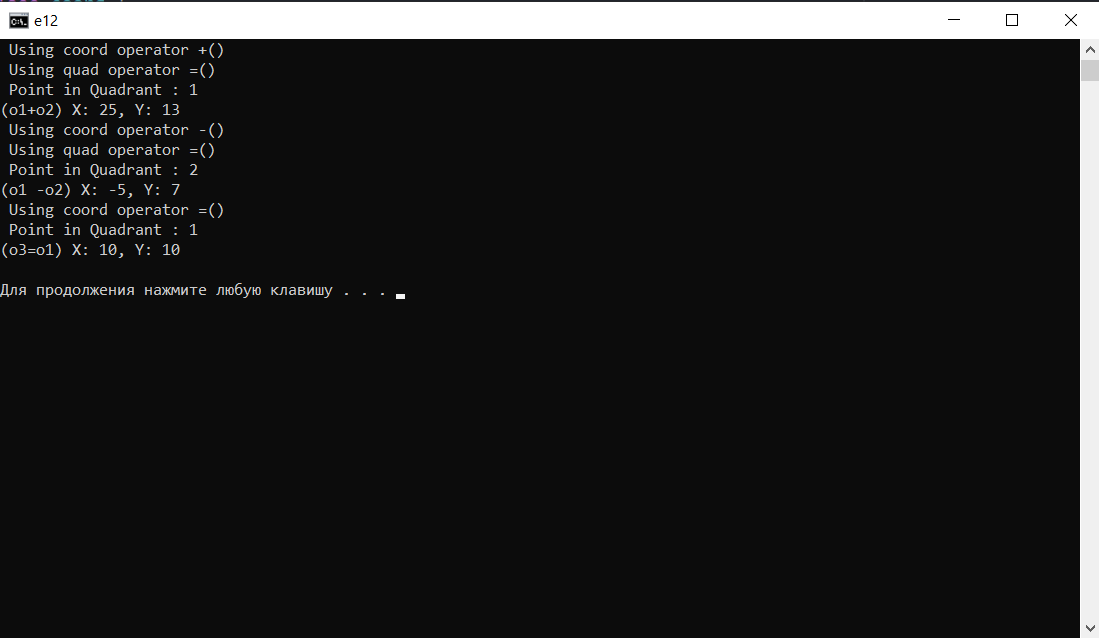
int main () {

car c(power , gas , 4, 500 , 5);

c. show ();

}

Завдання 8.12



Код :

#include <iostream>

using namespace std;

class coord {

public:

int x, y; // coordinate values

coord () { x=0; y=0; }

coord (int i, int j) { x=i; y=j; }

void get\_xy (int &i, int &j) { i=x; j=y; }

friend coord operator +( coord &ob1 , coord &ob2);

friend coord operator -( coord &ob1 , coord &ob2);

coord operator =( coord &ob2);

};

///

// Overload + relative to coord class .

coord operator +( coord &ob1 , coord &ob2){

coord temp ;

cout << " Using coord operator +() \n";

temp.x = ob1.x + ob2.x;

temp.y = ob1.y + ob2.y;

return temp ;

}

// Overload - relative to coord class .

coord operator -(coord &ob1 , coord &ob2) {

coord temp ;

cout << " Using coord operator -() \n";

temp .x = ob1.x - ob2 .x;

temp .y = ob1.y - ob2 .y;

return temp ;

}

// Overload = relative to coord .

coord coord::operator =( coord&ob2){

cout << " Using coord operator =() \n";

x = ob2.x;

y = ob2.y;

return \* this ; // return the object that is assigned to

}

///

class quad : public coord{

int quadrant ;

public :

quad (){

x = 0;

y = 0;

quadrant = 0;

}

quad (int x, int y) : coord (x, y){

if(x >=0 && y >= 0) quadrant = 1;

else if(x <0 && y >=0) quadrant = 2;

else if(x <0 && y <0) quadrant = 3;

else quadrant = 4;

}

void showq (){

cout << " Point in Quadrant : " << quadrant << '\n';

}

quad operator =( coord ob2 );

};

///

quad quad :: operator =( coord ob2 ){

cout << " Using quad operator =() \n";

x = ob2.x;

y = ob2.y;

if(x >=0 && y >= 0) quadrant = 1;

else if(x <0 && y >=0) quadrant = 2;

else if( x <0 && y <0) quadrant = 3;

else quadrant = 4;

return \* this ;

}

int main (){

quad o1 (10 , 10) , o2 (15 , 3) , o3;

int x, y;

o3 = o1 + o2; // add two objects - this calls operator +()

o3. get\_xy (x, y);

o3. showq ();

cout << "(o1+o2) X: " << x << ", Y: " << y << "\n";

o3 = o1 - o2; // subtract two objects

o3.get\_xy (x, y);

o3. showq (); cout << "(o1 -o2) X: " << x << ", Y: " << y << "\n";

o3 = o1; // assign an object

o3. get\_xy (x, y);

o3. showq ();

cout << "(o3=o1) X: " << x << ", Y: " << y << "\n";

}