/\*

Vinay 7118

Program-15 : Use of Static Class member

\*/

#include<iostream>

using namespace std;

class item{ static int count; int number; public:

void getdata(int a){ number=a; count++;

}

void getcount(void){ cout << "Count: "; cout << count << endl;

} }; int item :: count;

int main(){ item a,b,c;

a.getcount();

b.getcount();

c.getcount();

a.getdata(100);

a.getdata(200);

a.getdata(300);

cout << "After reading data" << endl;

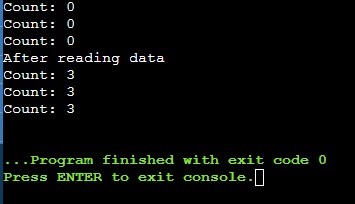
a.getcount();

b.getcount();

c.getcount(); return 0;

}

Output:



/\*

Vinay 7118

Program-16 : Use of static Member function

\*/

#include<iostream>

using namespace std;

class test{ int code; static int count; public:

void setcode(void){ code=++count;

} void showcode(void){ cout << "Object number: " << code << endl;

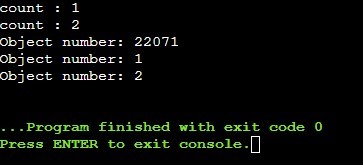
} static void showcount(void){ cout << "count : " << count << endl;

} }; int test :: count; int main(){

test t1,t2; t2.setcode(); test :: showcount(); test t3; t3.setcode(); test :: showcount(); t1.showcode(); t2.showcode(); t3.showcode(); return 0;

}

Output:



/\*

Vinay 7118

Program-17 : Array of Objects

\*/

#include<iostream>

using namespace std;

class employee{ char name[30]; float age; public: void getdata(void); void putdata(void);

}; void employee :: getdata(void){ cout << "Enter name : "; cin >> name; cout << "Enter age : "; cin >> age; } void employee :: putdata(void){ cout << "Name : " << name << endl; cout <<

"Age : " << age << endl;

} int main(){ const int size=3; employee manager[size]; for(int i=0;i<size;i++){ cout << "\n Details of manager " << i+1 << endl; manager[i].getdata();

} cout << "\n"; for(int i=0;i<size;i++){ cout << "\n Manager " << i+1 << "\n"; manager[i].putdata();

} return 0;

}

D3d complier

Output:

Details of manager 1

Enter name : Abhi

Enter age : 25

Details of manager 2

Enter name : Shourya

Enter age : 26

Details of manager 3

Enter name : Aryan

Enter age : 45

Manager 1

Name : Abhi

Age : 25

Manager 2

Name : Shourya

Age : 26

Manager 3

Name : Aryan

Age : 45

/\*

Vinay 7118

Program-18 : Objects as arguments

\*/

#include <iostream> using namespace std;

class thime{ int hours, minutes; public:

void gettime(int h, int m){ hours = h; minutes = m;

} void puttime(){ cout << hours << " hours and "; cout << minutes << " minutes" << endl;

} void sum(thime,thime);

};

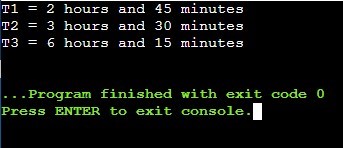
void thime::sum(thime t1, thime t2){ minutes = t1.minutes + t2.minutes; hours = minutes/60; minutes = minutes%60; hours = hours + t1.hours + t2.hours;

}

int main(){ thime t1, t2, t3; t1.gettime(2,45); t2.gettime(3,30); t3.sum(t1,t2); cout << "T1 = "; t1.puttime(); cout <<"T2 = "; t2.puttime(); cout << "T3 = "; t3.puttime(); return 0;

}

Output:



/\*

Vinay 7118

Program-19 : Friend in one class

\*/

#include<iostream>

using namespace std;

class sample{ int a; int b; public:

void setvalue(){ a=25; b=40; } friend float mean(sample s);

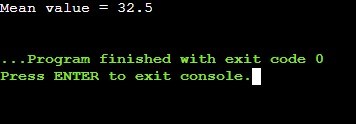
}; float mean(sample s){ return float(s.a + s.b)/2.0;

}

int main(){ sample X; X.setvalue(); cout << "Mean value = " << mean(X) << endl; return 0;

}

Output:



/\*

Vinay 7118

Program-19 : Use in Default Constructor

\*/

#include<iostream> using namespace std;

class rect{ int length,breadth; public:

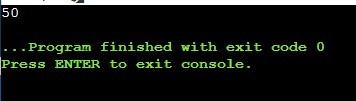
rect(){ length=5; breadth=10;

}; int rectangle(){ return (length\*breadth);

} }; int main(){ rect result; cout << result.rectangle(); return 0;

}

Output:



/\*

Vinay 7118

Program-21 : Parameterized Constructor

\*/

#include <iostream> using namespace std; class Point{ int x, y; public:

Point(int a, int b){ x = a; y = b; } void display()

{ cout << " ( " << x << " , " << y << " ) " << endl;

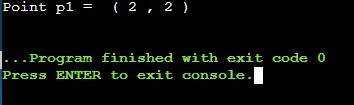
} }; int main()

{

Point p1(2, 2); cout << "Point p1 = "; p1.display(); return 0;

}

Output:



/\*

Vinay 7118

Program-22 : Copy Constructor

\*/

#include<iostream> using namespace std; class code{ int id; public: code(){} code(int a){ id=a; } code(code &x){ id

= x.id; } void display(void){ cout

<< id;

} }; int main(){ code A(100); code B(A); code

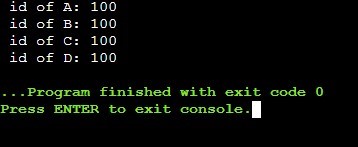
C=A; code D;

D=A;

cout << "\n id of A: "; A.display(); cout << "\n id of B: "; B.display(); cout << "\n id of C: "; C.display(); cout << "\n id of D: "; D.display(); return 0;

}

Output:



/\*

Vinay 7118

Program-23 : Destructor

\*/

#include<iostream>

using namespace std;

int count=0; class test{ public: test(){ count++; cout << "\nConstructor Msg: Object number " << count << " created.. " ;

}

~test(){ cout << "\n\nDestructors Msg: Object number " << count << " destroyed.."; count--;

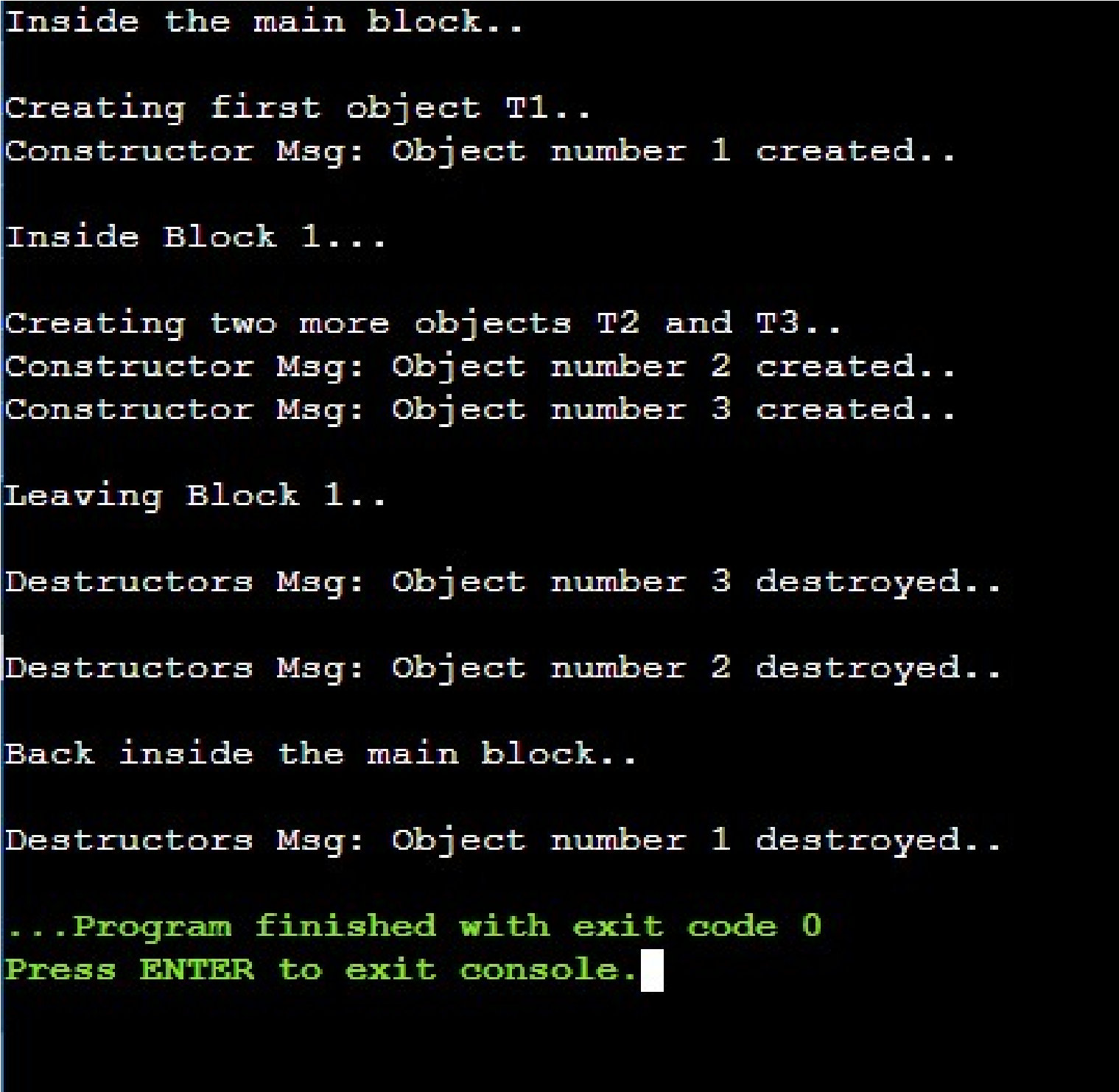
} }; int main(){ cout << "Inside the main block.."; cout << "\n\nCreating first object T1..";

test T1;{ cout << "\n\nInside Block 1..."; cout << "\n\nCreating two more objects T2 and T3.."; test T2,T3; cout << "\n\nLeaving Block 1..";

} cout << "\n\nBack inside the main block.."; return 0;

}

Output:



/\*

Vinay 7118

Program-24 : Constructor with dynamic operation

\*/

#include<iostream> #include<string.h> using namespace std;

class String{ char \*name; int length; public:

String(){ length=0; name=new char[length+1];

}

String(char \*s){ length = strlen(s);

name = new char[length+1];

strcpy(name, s);

} void display(void){ cout << name << endl;

} void join(String &a,String &b);

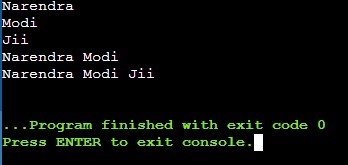
}; void String :: join(String &a,String &b){ length=a.length+b.length; delete name; name = new char[length+1]; strcpy(name, a.name); strcat(name, b.name);

}; int main(){ char \*first="Narendra ";

String name1(first),name2("Modi "),name3("Jii"),s1,s2; s1.join(name1,name2); s2.join(s1,name3); name1.display(); name2.display(); name3.display(); s1.display(); s2.display(); return 0;

}

Output:



/\*

Vinay 7118

Program-25 : Dynamic Object

\*/

#include<iostream>

using namespace std;

class rectangle{ int l,b; public: rectangle(){ cout << "Const with no Parameter\n";

} void read(){ cout << "Enter length & breadth : \n"; cin>>l>>b;

} void area(){ cout << "Area of rectangle is " << l\*b;

}

~rectangle(){ cout << "\nDestruct invoked";

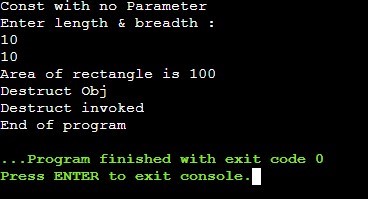
} }; int main(){ rectangle \*ptr; ptr = new rectangle; ptr -> read(); ptr -> area(); cout << "\nDestruct Obj"; delete ptr;

cout << "\nEnd of program"; return

0;

}

Output:



/\*

Vinay 7118

Program-26 : Operator overloading post and prefix increment

\*/

#include<iostream> using namespace std;

class score{ int val; public:

|  |  |
| --- | --- |
| score(){ val=0;  } |  |
| score operator++(){ val=val+1; temp.val=val; return(temp);  } | //prefix overload operator function score temp; |
| score operator++(int){ temp.val=val; val=val+1; return(temp);  } int show(){ return(val);  } | //postfix overload operator function score temp; |

}; int main(){ score s1,s2;

cout << "\nInitial value of s1 object is " << s1.show(); cout << "\nInitial value of s2 object is " << s2.show(); s2 = ++s1; cout << "\ns1 = " << s1.show(); cout <<"\ns2 after prefix operation = " << s2.show(); s2 = s1++; cout << "\ns1 = " << s1.show(); cout << "\ns2 after postfix operation = " << s2.show(); return 0;

}

Output:

