**Constructors, Destructors, pointer of object**

**Part1:** Constructors

A constructor in C++ is a **special method** that is automatically called when an object of a class is created.

To create a constructor, use the same name as the class, followed by parentheses ():

|  |  |
| --- | --- |
| // MyClass.h  #include <iostream>  using namespace std;  class MyClass {  public:  MyClass(); // Constructor  }; | // MyClass.cpp  #include <iostream>  #include “MyClass.h ”  using namespace std;  MyClass ::MyClass()  {  cout << "Hello World!";  } |
| #include <iostream>  #include “MyClass.h ”  using namespace std;  int main() {  MyClass myObj; // Create an object of MyClass (this will call the constructor)  MyClass myObjx= myObjw; //another way of invoking copy constructor  return 0;  } | |

Constructors can also take parameters (just like regular functions), which can be useful for setting initial values for attributes.

|  |  |
| --- | --- |
| // MyClass.h  #include <iostream>  using namespace std;  class MyClass {  public:  MyClass(int x); // Constructor  }; | // MyClass.cpp  #include <iostream>  #include “MyClass.h ”  using namespace std;  MyClass ::MyClass(int x)  {    cout << "x="<<x;  } |
| #include <iostream>  #include “MyClass.h ”  using namespace std;  int main() {  MyClass myOb(10)j; // Create an object of MyClass (this will call the constructor)  return 0;  } | |

Output:

|  |
| --- |
|  |

**Part2:** Destructors

Destructors are complimentary to constructors. They serve to clean up or do the final finishing task of objects when they are destroyed.

A destructor may be called either when the object goes out of scope or when it is destroyed explicitly using the delete operator. A destructor, like a constructor, has the same name as that of the class but is prefixed by the tilde ('~') symbol. A class cannot have more than one destructor. And destructor can't take arguments or specify a return value. The most common use of the destructor is to de-allocate memory that was allocated for the object by the constructor. A destroctor for above program is can be created as:

In MyClass.h, you need to add the destructor prototype:

~MyClass(); // Constructor

In MyClass.cpp, you need to implement your ~MyClass() function member:

~ MyClass () //destructor

{

cout<<"Destructor called, objects of this class are deleted "<<endl;

}

Update your class and run the client code:

Output

|  |
| --- |
|  |

**Part3: Pointer of object**

A pointer to a C++ class is done exactly the same way as a pointer to a structure and to access members of a pointer to a class you use the member access operator -> operator, just as you do with pointers to structures. Also as with all pointers, you must initialize the pointer before using it.

Example: create class Box with three private variables (double breadth, length and height) and constructor that takes three parameters. Also the Box class has volume method.

The implementation for the constructor as following:

// Constructor definition

Box(double l = 2.0, double b = 2.0, double h = 2.0) {

cout <<"Constructor called." << endl;

length = l;

breadth = b;

height = h;

}

// volume definition

double Volume() {

return length \* breadth \* height;

}

In the client code, try the following example:

int main(void) {

Box Box1(3.3, 1.2, 1.5); // Declare box1

Box Box2(8.5, 6.0, 2.0); // Declare box2

Box \*ptrBox; // Declare pointer to a class.

// Save the address of first object

ptrBox = &Box1;

// Now try to access a member using member access operator

cout << "Volume of Box1: " << ptrBox.Volume() << endl;

// Save the address of second object

ptrBox = &Box2;

cout << "Volume of Box2: " << ptrBox.Volume() << endl;

return 0;

}

Output:

Now, update the class to have only default constructer with no parameters. The new constructor job is to initialize the private members as following:

length = 2.0;

breadth = 2.0;

height = 2.0;

also, create new object box3 of class Box , and use the pointer of object to deal with it.

default constructer

Main function

**Exercises**:

1. Write a program that has a class to represent time. The class should have constructors to initialize data members hour, minute, and second to 0 and to initialize them to values passed as arguments. The class should have a member function to add time objects and return the result as a time object. There should be functions to display time in 12-hour and 24-hour format.
2. Write a program that has a class with a dynamically allocated character array as its data member. One object should contain "Engineers are" and another should contain " Creatures of logic". Member function join() should concatenate two strings by passing two objects as arguments. Display the concatenated string through a member function. Use constructors to allocate and initialize the data member. Also, write a destructor to free the allocated memory for the character array. Make your own function for the concatenation of two strings.
3. Write a class that can store Department ID and Department Name with constructors to initialize its members. Write destructor member in the same class and display the message "Object n goes out of the scope". Your program should be made such that it should show the order of constructor and destructor invocation.

That is the end of labsheet.. Good Luck