**[c++ - Defining member functions inside or outside the class definition](http://ecomputernotes.com/cpp/classes-in-c/defining-member-functions)**

BY DINESH THAKUR Category: [Classes in C++](http://ecomputernotes.com/cpp/classes-in-c)

Member functions of a class can be defined either outside the class definition or inside the class definition. **In**both the cases, the function body remains the same, however, the function header is different.

**Outside the Class:**Defining a member function outside a class requires the function declaration (function prototype) to be provided inside the class definition. The member function is declared inside the class like a normal function. This declaration informs the compiler that the function is a member of the class and that it has been defined outside the class. After a member function is declared inside the class, it must be defined (outside the class) in the program.

The definition of member function outside the class differs from normal function definition, as the function name in the function header is preceded by the class name and the scope resolution operator (: :). The scope resolution operator informs the compiler what class the member belongs to. The syntax for defining a member function outside the class is

Return\_type class\_name :: function\_name (parameter\_list)

{

*// body*of the member function

}

To understand the concept of defining a member function outside a class, consider this example.

**Example :**Definition of member function outside the class

class book

{

*II*body of the class

} :

void book :: getdata(char a[],float b)

{

// defining member function outside the class

strcpy(title,a):

price = b:

}

void book :: putdata ()

{

cout<<"\nTitle of Book: "<<title;

cout<<"\nPrice of Book: "<<price;

}

Note that the member functions of the class can access all the data members and other member functions of the same class (private, public or protected) directly by using their names. In addition, different classes can use the same function name.

**Inside the Class:**A member function of a class can also be defined inside the class. However, when a member function is defined inside the class, the class name and the scope resolution operator are not specified in the function header. Moreover, the member functions defined inside a class definition are by default inline functions.

To understand the concept of defining a member function inside a class, consider this example.

**Example**: Definition of a member function inside a class

class book

{

char title[30];

float price;

public:

void getdata(char [],float); *II*declaration

void putdata()//definition inside the class

{

cout<<"\nTitle of Book: "<<title;

cout<<"\nPrice of Book: "<<price;

} ;

In this example, the member function putdata() is defined inside the class book. Hence, putdata()is by default an inline function.

Note that the functions defined outside the class can be explicitly made inline by prefixing the keyword inline before the return type of the function in the function header. For example, consider the definition of the function getdata().

inline void book ::getdata (char a [],float b)

*{*

*body*of the function

}

**Constructor Types:**

<https://www.geeksforgeeks.org/constructors-c/>

**Destructor:**

<https://www.geeksforgeeks.org/destructors-c/>

**Friend Function:**

#include <iostream>

using namespace std;

class Box {

 double width;

 public:

 friend void printWidth( Box box );

 void setWidth( double wid );

};

// Member function definition

void Box::setWidth( double wid ) {

 width = wid;

}

// Note: printWidth() is not a member function of any class.

void printWidth( Box box ) {

 /\* Because printWidth() is a friend of Box, it can

 directly access any member of this class \*/

 cout << "Width of box : " << box.width <<endl;

}

// Main function for the program

int main() {

 Box box;

 // set box width without member function

 box.setWidth(10.0);

 // Use friend function to print the wdith.

 printWidth( box );

 return 0;

}