C++ Objects as Physical Objects

In many programming situations, objects in programs represent C++ Objects as Physical Objects: **things that can be felt or seen**. These situations provide vivid examples of the correspondence between the program and the real world.

- Circles as Objects
- Widget Parts as Objects

Declaring Objects: When a class is defined, only the specification for the object is defined; no memory or storage is allocated. To use the data and access functions defined in the class, you need to create objects.

Syntax:

ClassName ObjectName;

Accessing data members and member functions: The data members and member functions of class can be accessed using the dot('.') operator with the object. For example if the name of object is *obj* and you want to access the member function with the name *printName()* then you will have to write *obj.printName()* .

Accessing Data Members

The public data members are also accessed in the same way given however the private data members are not allowed to be accessed directly by the object. Accessing a data member depends solely on the access control of that data member.
This access control is given by Access modifiers in C++. There are three access modifiers : **public, private and protected**.

```
// C++ program to demonstrate
// accessing of data members

#include <bits/stdc++.h>
using namespace std;
class Geeks
{
   // Access specifier
   public:

   // Data Members
   string geekname;

   // Member Functions()
   void printname()
   {
      cout << "Geekname is: " << geekname;
   }
```

```
};

int main() {

    // Declare an object of class geeks
    Geeks obj1;

    // accessing data member
    obj1.geekname = "Abhi";

    // accessing member function
    obj1.printname();
    return 0;
}
```

Output:

Geekname is: Abhi

Member Functions in Classes

There are 2 ways to define a member function:

- Inside class definition
- Outside class definition

To define a member function outside the class definition we have to use the scope resolution :: operator along with class name and function name.

```
// C++ program to demonstrate function
// declaration outside class

#include <bits/stdc++.h>
using namespace std;
class Geeks
{
    public:
    string geekname;
    int id;

    // printname is not defined inside class definition
    void printname();

    // printid is defined inside class definition
    void printid()
    {
        cout << "Geek id is: " << id;
    }
};
```

```
// Definition of printname using scope resolution operator ::
void Geeks::printname()
{
   cout << "Geekname is: " << geekname;
}
int main() {

   Geeks obj1;
   obj1.geekname = "xyz";
   obj1.id=15;

   // call printname()
   obj1.printname();
   cout << endl;

   // call printid()
   obj1.printid();
   return 0;
}
```

Output:

```
Geekname is: xyz
Geek id is: 15
```