OOPS

Classes

In C++, a class is declared using the **class** keyword. The syntax of a class declaration is similar to that of a structure. Its general form is:

```
class class-name{
    // Private functions and Variables
    public:
    // Public functions and variables
} object-list;
```

In a class declaration, the object list is **optional**.

The class name is technically optional. From a practical point of view, it is virtually always needed. The reason is that the class name becomes a new type name that is used to declare objects of the class.

Functions and variables declared inside the class declaration are said to be **members** of the class.

By default, **all member functions and variables are private** to that class. This means that they are accessible only by other members of that class.

To declare public class members, the **public** keyword is used, followed by a colon. All functions and variables declared after the public specifier are accessible both by other members of the class and by any part of the program that contains the class.

```
#include <bits/stdc++.h>
using namespace std;
// Class Declartion
class myclass {
   // Private Members to myclass
   int a;

  public:
   // Public members to myclass
   void set_a(int num);
   int get_a();
};
int main() { return 0; }
```

This class has one private variable, called ***a***, and two public functions ***seta()*** *and* ***geta()***.

Notice that the functions are declared within a class using their prototype forms. The functions that are declared to be part of a class are called member functions.

Since ***a*** is private it is not accessible by any code outside ***myclass***. However, since ***seta()*** *and* ***geta()*** are members of ***myclass***, they have access to ***a*** and as they are declared as public member of ***myclass***, they can be called by any part of the program that contains ***myclass***.

The member functions need to be defined. You do this by preceding the function name with the class name followed by two colons (:: are called scope resolution operator). For example, after the class declaration, you can declare the member functions as

```
#include <bits/stdc++.h>
using namespace std;
// Class Declartion
class myclass {
   // Private Members to myclass
   int a;

  public:
   // Public members to myclass
   void set_a(int num);
   int get_a();
};

// Member Functions
```

```
void myclass::set_a(int num) { a = num; }
int myclass::get_a() { return a; }

int main() {
   myclass c;
   c.set_a(100);
   cout << c.get_a();
   return 0;
}
```

Output:

```
100
```

In general to declare a member function, you use this form:

```
return-type class-name::func-name(parameter-list){
    // Function Body
}
```

Here the class-name is the name of the class to which the function belongs.

The declaration of a class does not define any objects of the type ***myclass***. It only defines the type of object that will be created when one is actually declared. To create an object, use the class name as type specifier. For example,

```
int main() {
   myclass c;
   c.set_a(100);
   cout << c.get_a();
   return 0;
}
```

Remember that an object declaration creates a physical entity of that type. That is, **an object occupies memory space, but a type definition does not**.

Once an object of a class has been created, your program can reference its public members by using the dot operator in much the same way that structure members are accessed as shown above.

It is important to remember that although all objects of a class share their functions, each object creates and maintains its own data.

NO NOTES TO SHOW