**CLASSES AND METHODS**

Structure Vs Class

| **BASIS FOR COMPARISON** | **STRUCTURE** | **CLASS** |
| --- | --- | --- |
| Basic | If access specifier is not declared, by default all member are 'public'. | If access specifier is not declared, by default all members are 'private'. |
| Declaration | struct structure\_name{ type struct\_element 1; type struct\_element 2; type struct\_element 3; . }; | class class\_name{ data member; member function; }; |
| Instance | Instance of 'structure' is called 'structure variable'. | Instance of a 'class' is called 'object'. |
| Polymorphism and inheritance | Not supported | Supports polymorphism and a class can also be inherited. |
| Nature | Value type | Reference type |
| Memory is allocated on | Stack | Heap |
| Null values | Not possible | Can have null values |
| Requires constructor and destructor | No | Yes |

Key Differences Between Structure and Class

1. The main difference between structures and classes is that by default, all member of the structure are public. In contrast, by default, all the members of the class are private.
2. A structure is considered as the value type whereas, a class is a reference type.
3. At the time of instantiating a structure, the memory is allocated on a stack. As against, in class, the memory is alotted on the heap.
4. No structure member can have a null value. Conversely, the variables of a class can have null values.
5. In order to initialize the member of a class, the constructors and destructors are used. On the contrary, the structure can initialize its members automatically.
6. In variables in the structure can not be initialized during the declaration while in case of class, this can be done.

Similarities

* In C++, both the structure and class are syntactically equivalent.
* Both structure and class can declare some of their members private.
* The name of a structure or class can be used as a stand-alone type.
* Both structure and class support the mechanism of inheritance.

### Definition of Class

**Class** in OOP defines a new type which contains data members and member function, that are used to access the data members of the class. The instances of classes are called “**objects**” each of which has the same organization as a class. The class is a logical abstraction, whereas an object has a physical existence.

#### **Syntax**

The class is syntactically similar to the structure. The class can be declared as follow.

#include<iostream.h> // compulsory

…….

class class \_name(i,e user defined name)

{

Access specifier1: //private or public or protected

data members and member functions.

access specifier2:

type data member;

return type **member\_function**(parameter list){ . . }

} ;

object creation;

Here, the class is a keyword which declares to the compiler that a class has been declared. The main feature of OOP is **data hiding** which is accomplished by providing three access specifiers that are “public”, “private”, “protected”. If you do not specify any access specifier in the class while declaring data members or member functions, by default all are considered private.

The public access specifier allows, functions or data to be accessed by other parts of your program. The private members of the class can only be accessed by a member of that class only. The protected access specifier is applied during inheritance. Once you declare the access specifier, it can not be changed throughout the program.

#### **How to access an object?**

The objects are nothing but the instance of the classes. The members of the class are accessed by the object of the same class using a dot (.) operator.

object.mem\_funct(arguments);

* The object can also be passed as an argument to a function.
* The pointer to an object can also be created.

**# include<iostream.h> // compulsory**

**#include<conio.h>**

**class test**

**{**

**private:**

**char name[20]; // Member variables declaration and being static can not be //initialized during declaration. It assigns attributes to the class**

**Int age;**

**public:**

**void set\_data() //member function delaration(sets properties of class)**

**{**

**cout<<”Input your name”<<endl;**

**cin>>name;**

**cout<<”Input Age:”;**

**cin>age;**

**}**

**void disp\_data()**

**{**

**clrscr();**

**cout<<”\n Your Good Name: “<<name;**

**cout<<”\n Your Age is “<<age;**

**}**

**}; // class is terminated**

**void main()**

**{**

**test obj; //object obj is created**

**obj.set\_data(); // member function is called( Message passing)**

**obj.disp\_data(); // “ “ “**

**getch();**

**}**