**Syntax for Pure Virtual Function is:- virtual void show()==0**

A **pure virtual function** is declared by assigning 0 in declaration.

In C++,A class that contain ‘pure virtual function is called ‘**Abstract Class’**.

**COMPILE TIME AND RUNTIME**

**Static Binding**:- things that is created during compile time and stays fixed during the execution of program.

Example:- Function and Operator Overloading

**Dynamic Binding**:- things that are created and can change duringexecution.

Example:- Virtual and Pure Virtual Function.

**Which of the following access specifies is used in a class definition by default?  
Private**

In C++ if you use **class** to define your class, then the default access is **private**. If you define the same class using **struct**, then the default access is **public**.

**Which of the following can access private data members or member functions of a class?**

**Any member function of that class and friend function.**

Private members are accessed only within the class they are declared. Friend function is used to access the private and protected members of different classes.

**Mutable in C++**

Class objects or member function may be declared as **const**, thus making their member data not modifiable.

**In programming, there may arise some situations where** **If it is necessary to make changes in data item in constant object then In such situations, the data item may be declared as mutable.**

But in programming, there may arise some situations where programmers may want to create a constant object but would likely to modify particular data item. In such situations, the data item may be declared as mutable.

Example:

mutable int s;

Although a function that contains k is declared const, the value of k may be modified. This is the special characteristics of the mutable keyword.

**Manipulators**

Manipulators are operators used in C++ for formatting output.

1. setw(x)// sets the minimum field width on output.

2. setfill('\*')

Ex: cout << setw(20) << setfill('\*') << "w3schools.in" << /\*setw(20) << setfill('\*')<<"Test"<< \*/endl;

Other manipulater: <http://www.cplusplus.com/reference/library/manipulators/>

**Variable:**

Read all carefully: <http://faculty.cs.niu.edu/~freedman/241/241notes/241var2.htm>

**Inline function**

When the program executes the function call instruction the CPU stores the memory address of the instruction following the function call, copies the arguments of the function on the stack and finally transfers control to the specified function. The CPU then executes the function code, stores the function return value in a predefined memory location/register and returns control to the calling function. This can become overhead if the execution time of function is less than the switching time from the caller function to called function (callee).

**When the inline function is called whole code of the inline function gets inserted at the point of inline function call. This substitution is performed by the C++ compiler at compile time. Inline function may increase efficiency if it is small.**