# Virtual Function

- A virtual function is a member function in the base class that we expect to redefine in derived classes.
- It is declared using the virtual keyword.
- A virtual function is used in the base class to ensure that the function is overridden.
- This especially applies to cases where a pointer of base class points to a derived class object.

Example:

```
#include <iostream>
using namespace std;

class Base {
   public:
    virtual void print() {
        cout << "Base Function" << endl;
    }
};

class Derived : public Base {
   public:
    void print() {
        cout << "Derived Function" << endl;
    }
};

int main() {
    Derived derived1;

    // pointer of Base type that points to derived1
    Base* base1 = &derived1;

    // calls member function of Derived class
    base1->print();

    return 0;
}

Output:
Derived Function
```

C++ determines which function is invoked at the runtime based on the type of object pointed by the base class pointer when the function is made virtual.

### Pure Virtual Function:

- A pure virtual function is a virtual function in C++ for which we need not write any function definition and only have to declare it.
- It is declared by assigning 0 in the declaration.

```
class A {
   public:
      virtual void s() = 0; // Pure Virtual Function
};
```

- A pure virtual function (or abstract function) in C++ is a virtual function for which we can implement,
- But we must override that function in the derived class;
- otherwise, the derived class will also become an abstract class.