# Virtual Functions in C++ | C++ Tutorials for Beginners #56

In this tutorial, we will discuss virtual functions in C++

**Virtual Functions in C++**

A member function in the base class which is declared using virtual keyword is called virtual functions. They can be redefined in the derived class. To demonstrate the concept of virtual functions an example program is shown below

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

class BaseClass{
    public:
        int var_base=1;
        virtual void display(){
            cout<<"1 Dispalying Base class variable var_base "<<var_base<<endl;
        }
};

class DerivedClass : public BaseClass{
    public:
            int var_derived=2;
            void display(){
                cout<<"2 Dispalying Base class variable var_base "<<var_base<<endl;
                cout<<"2 Dispalying Derived class variable var_derived "<<var_derived<<endl;
            }
```

```
};
```

Copy

***Code Snippet 1: Virtual Function Example Program***

As shown in code snippet 1,

1. We created a class “BaseClass” which contains public data member “var_base” which has the value “1” and member function “display”. The member function “display” will print the value of data member “var_base”
2. We created another class “DerivedClass” which is inheriting “BaseClass” and contains data member “var_derived” which has the value “2” and member function “display”. The member function “display” will print the values of data members “var_base” and “var_derived”

The code for the main program is shown below

```
int main(){
    BaseClass * base_class_pointer;
    BaseClass obj_base;
    DerivedClass obj_derived;

    base_class_pointer = &obj_derived;
    base_class_pointer->display();
    return 0;
}
```

Copy

***Code Snippet 2: Main Program***

As shown in code snippet 2,

1. We created a pointer “base_class_pointer” of the data type “Baseclass”
2. Object “obj_base” of the data type “BaseClass” is created.

3. Object “obj_derived” of the data type “DerivedClass” is created
4. Pointer “base_class_pointer” of the base class is pointing to the object “obj_derived” of the derived class
5. The pointer “base_class_pointer” is pointed to the object “obj_derived” of the derived class.
6. The function “display” is called using the pointer “base_class_pointer” of the base class.

The main thing to note here is that if we don’t use the “virtual” keyword with the “display” function of the base class then beside of the point that we have pointed our base call pointer to derived class object still the compiler would have called the “display” function of the base class because this is its default behavior as we have seen in the previous tutorial.
But we have used the “virtual” keyword with the “display” function of the base class to make is **virtual function** so when the display function is called by using the base class pointer the display function of the derived class will run because the base class pointer is pointing to the derived class object.
The output of the following program is shown in figure 1