# Tutorial 55 - Pointers to Derived Classes in C++

**Definition:**

- In C++, a **base class pointer** can point to a **derived class object**, allowing access to the base class members.
- However, a base class pointer **cannot access derived class members** directly.
- Example program demonstrates the behavior.

---

### Code Snippet 1: Base and Derived Classes

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

class BaseClass {
public:
    int var_base;
    void display() {
        cout << "Displaying Base class variable var_base: " << var_base << endl;
    }
};

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

```

---

### Key Points:

1. **BaseClass**:
   - Contains a public data member `var_base`.
   - Function `display` prints the value of `var_base`.
2. **DerivedClass**:
   - Inherits from `BaseClass`.
   - Contains an additional data member `var_derived`.
   - Function `display` prints values of `var_base` and `var_derived`.

---

### Code Snippet 2: Main Program

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

    base_class_pointer = &obj_derived; // Base class pointer points to derived class object
    base_class_pointer->var_base = 34;
```

```
    // base_class_pointer->var_derived = 134; // Error: Cannot access derived class member
    base_class_pointer->display(); // Calls BaseClass display

    base_class_pointer->var_base = 3400;
    base_class_pointer->display(); // Updated value of var_base

    DerivedClass* derived_class_pointer = &obj_derived;
    derived_class_pointer->var_base = 9448; // Access base class member
    derived_class_pointer->var_derived = 98; // Access derived class member
    derived_class_pointer->display(); // Calls DerivedClass display

    return 0;
}

```

---

### Key Points from Main Program:

1. A **base class pointer** can:
   - Point to a **derived class object**.
   - Access **base class members** only.
   - Call the **base class version** of the `display` function.
2. A **derived class pointer** can:
   - Point to a **derived class object**.
   - Access both **base class** and **derived class members**.
   - Call the **derived class version** of the `display` function.
3. **Important Notes**:
   - Base class pointer **cannot modify/access derived class members**.
   - Derived class pointer can **modify/access both base and derived class members**.

---

### Output:

1. **Base Class Pointer**:

```
Displaying Base class variable var_base: 34
Displaying Base class variable var_base: 3400

```

2. **Derived Class Pointer**:

```
Displaying Base class variable var_base: 9448
Displaying Derived class variable var_derived: 98

```

---

### Short Notes for Notebook:

#### Pointers to Derived Classes in C++:

1. **Base Class Pointer**:
   - Can point to a **derived class object**.
   - **Cannot access** derived class members.
   - Calls **base class functions**.

2. **Derived Class Pointer**:
   - Can point to a **derived class object**.
   - Can access **both base and derived class members**.
   - Calls **derived class functions**.
3. **Key Behavior**:
   - Base class pointer **only works with base class members**.
   - Derived class pointer allows access to all members and overrides base class functions.

---