CSE 232

Programming with C++

Lecture 10

Object Oriented Programming (3)

Prepared by

Md. Mijanur Rahman, Prof. Dr.
Dept. of Computer Science and Engineering
Jatiya Kabi Kazi Nazrul Islam University, Bangladesh
Email: mijan@jkkniu.edu.bd
Web: www.mijanrahman.com

Contents

Object Oriented Programming

- C++ Inheritance
- Modes of Inheritance in C++
- Types of Inheritance in C++
- Polymorphism in Inheritance

Inheritance in C++

- The capability of a class to derive properties and characteristics from another class is called Inheritance. Inheritance is one of the most important features of Object Oriented Programming in C++.
- **Syntax of Inheritance in C++**

```
class  derived_class_name : access-specifier  base_class_name
{
    //    body ....
};
```

where,

- class: keyword to create a new class
- derived_class_name: name of the new class, which will inherit the base class
- access-specifier: Specifies the access mode which can be either of private, public or protected. If neither is specified, private is taken as default.
- base-class-name: name of the base class.

Inheritance in C++

- Program to Demonstrate the Simple Inheritance of a Class:

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

class Parent {
public:
    int roll;
    void printRoll()
    {
        cout << "Roll: " << roll << endl;
    }
};

class Child : public Parent {
public:
    float marks;
    void printAll()
    {
        cout << "Roll: " << roll << endl;
        cout << "Marks: " << marks << endl;
    }
};

int main()
{
    Child obj1;
    obj1.roll = 101;
    obj1.marks = 85.5;

    obj1.printAll();

    return 0;
}
```

```
Roll: 101
Marks: 85.5
```

Modes of Inheritance in C++

- Mode of inheritance controls the access level of the inherited members of the base class in the derived class. In C++, there are 3 modes of inheritance:
  - Public Mode
  - Protected Mode
  - Private Mode

- **Public Inheritance Mode:**
- If we derive a subclass from a public base class. Then the public member of the base class will become public in the derived class and protected members of the base class will become protected in the derived class.

Modes of Inheritance in C++

- **Protected Inheritance Mode:**
- If we derive a subclass from a Protected base class. Then both public members and protected members of the base class will become protected in the derived class.

- **Private Inheritance Mode:**
- If we derive a subclass from a Private base class. Then both public members and protected members of the base class will become private in the derived class. They can only be accessed by the member functions of the derived class.
- Private mode is the default mode that is applied when we don’t specify any mode.

Modes of Inheritance in C++

- Program to show different kinds of Inheritance Modes and their Member Access Levels:

```
1  class A {
2  public:
3      int x;
4
5  protected:
6      int y;
7
8  private:
9      int z;
10 };
11
12 class B : public A {
13     // x is public
14     // y is protected
15     // z is not accessible from B
16 };
17
18 class C : protected A {
19     // x is protected
20     // y is protected
21     // z is not accessible from C
22 };
23
24 class D : private A // 'private' is default for classes
25 {
26     // x is private
27     // y is private
28     // z is not accessible from D
29 };
```

Modes of Inheritance in C++

- Program to show different kinds of Inheritance Modes and their Member Access Levels:

```
Private Roll = 101
Protected Name = Rahman
Public Marks = 98.25
```

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

class Base {
private:
    int roll = 101;

protected:
    char name[10] = "Rahman";

public:
    float marks = 98.25;
    // function to access private member
    int getRoll() { return roll; }
};

class PublicDerived : public Base {
public:
    // function to access protected member from Base
    char * getName() { return name; }
};

int main()
{
    PublicDerived obj1;
    cout << "Private Roll = " << obj1.getRoll() << endl;
    cout << "Protected Name = " << obj1.getName() << endl;
    cout << "Public Marks = " << obj1.marks << endl;
    return 0;
}
```

Types of Inheritance in C++

- The inheritance can be classified on the basis of the relationship between the derived class and the base class. In C++, we have 5 types of inheritances:
  - Single inheritance
  - Multilevel inheritance
  - Multiple inheritance
  - Hierarchical inheritance
  - Hybrid inheritance

Types of Inheritance in C++

- **Single Inheritance:** In single inheritance, a class is allowed to inherit from only one class. i.e. one base class is inherited by one derived class only.
- Example:

```
class A
{
… .. …
};
class B: public A
{
… .. …
};
```

Types of Inheritance in C++

- **Multiple Inheritance:** Multiple Inheritance is a feature of C++ where a class can inherit from more than one class. i.e one subclass is inherited from more than one base class.
- Example:

```
class B
{
... .. ...
};
class C
{
... .. ...
};
class A: public B, public C
{
... ... ...
};
```

Types of Inheritance in C++

- **Multiple Inheritance:** C++ Program Example

```
Computer Science & Engineering
Jatiya Kabi Kazi Nazrul Islam University
Trishal, Mymensingh, Bangladesh.
```

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

// first base class
class Dept {
public:
    Dept() { cout << "Computer Science & Engineering\n"; }
};

// second base class
class Uni {
public:
    Uni() { cout << "Jatiya Kabi Kazi Nazrul Islam University\n"; }
};

// sub class derived from two base classes
class Address : public Dept, public Uni {
  public:
    Address() { cout << "Trishal, Mymensingh, Bangladesh.\n"; }
};

int main()
{
    Address obj;
    return 0;
}
```

Types of Inheritance in C++

- **Multilevel Inheritance:** In this type of inheritance, a derived class is created from another derived class and that derived class can be derived from a base class or any other derived class. There can be any number of levels.
- Example:

```
class C
{
... .. ...
};
class B : public C
{
... .. ...
};
class A: public B
{
... ... ...
};
```

Types of Inheritance in C++

- **Multilevel Inheritance:** C++ Program Example

```
Computer Science & Engineering
Jatiya Kabi Kazi Nazrul Islam University
Trishal, Mymensingh, Bangladesh.
```

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

// first base class
class Dept {
public:
    Dept() { cout << "Computer Science & Engineering\n"; }
};

// second sub-class derived from first base class
class Uni: public Dept {
public:
    Uni() { cout << "Jatiya Kabi Kazi Nazrul Islam University\n"; }
};

// Third sub-class derived from second sub class
class Address :  public Uni {
  public:
    Address() { cout << "Trishal, Mymensingh, Bangladesh.\n"; }
};

int main()
{
    Address obj;
    return 0;
}
```

Types of Inheritance in C++

- **Hierarchical Inheritance**
- In this type of inheritance, more than one subclass is inherited from a single base class. i.e. more than one derived class is created from a single base class.

- **Example:**

```
class A
{
   // body of the class A.
}
class B : public A
{
   // body of class B.
}
class C : public A
{
   // body of class C.
}
class D : public A
{
   // body of class D.
}
```

Types of Inheritance in C++

- **Hierarchical Inheritance:** C++ Program

```
This is a Vehicle
This Vehicle is Car
This is a Vehicle
This Vehicle is Bus
```

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

// base class
class Vehicle {
public:
    Vehicle() { cout << "This is a Vehicle\n"; }
};

// first sub class
class Car : public Vehicle {
public:
    Car() { cout << "This Vehicle is Car\n"; }
};

// second sub class
class Bus : public Vehicle {
public:
    Bus() { cout << "This Vehicle is Bus\n"; }
};

int main()
{
    Car obj1;
    Bus obj2;
    return 0;
}
```

Types of Inheritance in C++

- **Hybrid Inheritance**
- Hybrid Inheritance is implemented by combining more than one type of inheritance. For example: Combining Hierarchical inheritance and Multiple Inheritance will create hybrid inheritance in C++
- There is no particular syntax of hybrid inheritance. We can just combine two of the previous inheritance types.

- **Example:**

```
class F
{
... .. ...
}
class G
{
... .. ...
}
class B : public F
{
... .. ...
}
class E : public F, public G
{
... .. ...
}
class A : public B {
... .. ...
}
class C : public B {
... .. ...
}
```

Types of Inheritance in C++

- **Hybrid Inheritance:** C++ Program

```
This is a Vehicle
Fare of Vehicle
This Vehicle is a Bus with Fare
```

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

class Vehicle {
public:
    Vehicle() { cout << "This is a Vehicle\n"; }
};

class Fare {
public:
    Fare() { cout << "Fare of Vehicle\n"; }
};

class Car : public Vehicle {
  public:
  Car() { cout << "This Vehical is a Car\n"; }
};

class Bus : public Vehicle, public Fare {
  public:
  Bus() { cout << "This Vehicle is a Bus with Fare\n"; }
};

int main()
{
    Bus obj1;
    return 0;
}
```

Polymorphism in Inheritance

- In Inheritance, we can redefine the base class member function in the derived class. This type of inheritance is called Function Overriding.
- Generally, in other programming languages, function overriding is runtime polymorphism but in C++, we can do it at both runtime and complile time. For runtime polymorphism, we have to use the virtual functions.

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

class Parent {
    int roll = 101;
public:
    void Print()
    {
        cout << "Roll: " << roll << endl;
    }
};

class Child : public Parent {
    float marks = 65.50;
public:
    void Print()
    {
        cout << "Marks: " << marks << endl;
    }
};

int main()
{
    Child obj1;
    obj1.Print();
    return 0;
}
```

Lecture 10

Object Oriented Programming (3)

THE END