Heaven's Light is Our Guide
Rajshahi University of Engineering & Technology
Department of Computer Science & Engineering

Lab Manual

Course Code: **CSE 1204 (Sec A)**
Course Title: Sessional based on CSE 1203

Module 3 [Inheritance]: (for Week 3)

Problem Statement: You have the create an inheritance among **`Father`**-->**Son** -->**GrandSon** class. The **`father`** class has the following data members

```
class Father{
  private:
    int money;
  protected:
    int gold;
  public:
    int land;
};
```

Now write the **Son** and **GrandSon** classes with **`private`**/**`protected`**/**`public`** access modifier and do the following:

i) Try to access **`money`**, **`gold`** and **`land`** from Son class

ii) Try to access **`money`**, **`gold`** and **`land`** from GrandSon class

iii) Find the values of money, gold and land when different access modifer is used in the following table

Class		In Son class			In GrandSon class		
Son	GrandSon	money	gold	land	money	gold	land
public	public	?	?	?	?	?	?
protected	public	?	?	?	?	?	?
private	public	?	?	?	?	?	?
public	protected	?	?	?	?	?	?
protected	protected	?	?	?	?	?	?
private	protected	?	?	?	?	?	?
public	private	?	?	?	?	?	?
protected	private	?	?	?	?	?	?
private	private	?	?	?	?	?	?

*****

Topic 2 [**Types of Inheritance**]: Learn and Test different types of inheritance in C++. In each inheritance draw the class diagram with class chain and try to access the data members of bases classes from child classes.

i) Single inheritance

```
class A{
  private:
    int x;
 protected:
  int y;
public:
  int z;
}
```

```
class B:public A{
  //write public
method to //access
x,y & z
}
```

```
int main(){
 B b;
//call methods of
class B
return 0;
}
```

ii) Multi-level inheritance

```
class A{
  private:
    int x;
 protected:
  int y;
 public:
  int z;
}
```

```
class B:public A{
  }
```

```
class C:public
B{
  //write public
//method to
//access x,y & z
}
```

```
int main(){
 C c;
//call
//methods of
//class C
return 0
}
```

iii) Multiple inheritance

```
class A{
  private:
    int x;
 protected:
  int y;
 public:
  int z;
}
```

```
class B{
    private:
      int p;
   protected:
    int q;
   public:
    int r;
  }
```

```
class C:public A,
Public B{
  //write public
method //to access
//x,y,z,p,q & r
}
```

```
int main(){
 C c;
//call
//methods of
//class C
return 0
}
```

iv) Heirarchical inheritance

```
class A{
  private:
    int x;
protected:
  int y;
public:
  int z;
}
```

```
class B:public A
{
//write public
method to access
x,y & z
  }
```

```
class C:public A
{
  //write method
public to access
x,y & z
}
```

```
int main(){
 B b;
 C c;
//call
//methods of
//class B & C
return 0
}
```

v) Hybrid (Diamond) inheritance [virtual class]

```
class A{
  private:
    int x;
protected:
  int y;
public:
  int z;
}
```

```
class B:public A
{

 }
```

```
class C:public A
{

}
```

```
class D:public
B, public C {
  //write
public method
to access x,y
& z
}
```

```
int main(){
 D d;
//call
//methods of
//class D
return 0
}
```

Topic 3 [Constructor & Destructor in inheritance]: Write the constructors & destructors for different types of inheritance are given as follows. Also follow and write the sequence of their execution.

i) Single inheritance

```
class A{
  private:
    int ax;
 public:
  //write
  constructor to
  initialize ax
  //Write
  destructor
 }
```

```
class B:public A{
   private:
     int bx;
  public:
//write constructor to
initialize bx
//Write method to sum ax
and bx
//Write destructor
}
```

```
int main(){
 B b;
//call methods of
class B
return 0;
}
```

ii) Multi-level inheritance

```
private:
   int ax;
public:
  //write
  constructor to
  initialize ax
  //Write
  destructor
 }
```

```
class B:public A {
  private:
     int bx;
  public:
    //write
    constructor to
    initialize bx
    //Write
    destructor

  }
```

```
class C:public B {
 private:
     int cx;
  public:
    //write
    constructor to
    initialize cx
//Write method to
sum ax, bx and cx
//Write destructor
}
```

```
int main(){
 C c;
//call
//methods of
//class C
return 0
}
```

iii) Multiple inheritance

```
private:
    int ax;
public:
  //write
  constructor
  to initialize
  ax
  //Write
  destructor
}
```

```
class B{
   private:
     int bx;
  public:
   //write
   constructor
   to
   initialize
   bx
   //Write
   destructor
   }
```

```
class C:public A,
Public B{
 private:
     int cx;
  public:
   //write
   constructor to
   initialize cx
//Write method to sum
ax, bx and cx
//Write destructor
}
```

```
int main(){
 C c;
//call
//methods of
//class C
return 0
}
```

iv) Heirarchical inheritance

```
class A{
 private:
    int ax;
 public:
//write
 constructor to
 initialize ax
//Write
 destructor
 }
```

```
class B:public A
{
 private:
     int bx;
  public:
   //write
   constructor
   to initialize
   bx
   //Write
   destructor
  }
```

```
class C:public A {
 private:
     int cx;
  public:
   //write
   constructor to
   initialize cx
//Write method to
sum ax, bx and cx
//Write destructor
}
```

```
int main(){
 B b;
 C c;
//call
//methods of
//class B & C
return 0
}
```

v) Hybrid (Diamond) inheritance [virtual class]

```
class A{
private:
  int ax;
 public:
//write
 constructor
 to
 initialize
 ax
//Write
 destructor
}
```

```
class B:public
A {
 private:
   int bx;
  public:
//write
 constructor
 to initialize
 bx
//Write
 destructor

 }
```

```
class C:public A
{
   private:
      int cx;
   public:
    //write
    constructor
    to
    initialize
    cx
//Write
destructor

}
```

```
class D:public
B, public C {
    private:
       int dx;
   public:
//write
constructor
to initialize
dx
//Write
method to
sum ax, bx
cx and dx
//Write
destructor
}
```

```
int main(){
 D d;
//call
//methods of
//class D
return 0
}
```