Inheritance on Java

On Java we must know 2 main concepts about the inheritance of classes. The class from where the characteristics are inherited is the Parent class while the one which inherits the characteristics is the Child class. The following example shows this relation:

```
 class Parent
{
        int a;
        int insertA(int b)
        {
                a=b;
        }
}
class Child extends Parent
{
……………………………
}
Class Show
{
        Public static void main(String args[])
        {
                Parent P=new Parent();
                Child Ch=new Child();
                Ch.insertA(8);
        }
}
```

Examples

Exercise 1:

```
class Calculation

{

        int z;

        public void addition(int x, int y) {

          z = x + y;

          System.out.println("The sum of the given numbers:"+z);

        }


        public void Subtraction(int x, int y) {

          z = x - y;

          System.out.println("The difference between the given numbers:"+z);

        }

}


public class My_Calculation extends Calculation

 {

        public void multiplication(int x, int y) {

          z = x * y;

          System.out.println("The product of the given numbers:"+z);

        }


        public static void main(String args[]) {

          int a = 20, b = 10;

          My_Calculation demo = new My_Calculation();

          demo.addition(a, b);

          demo.Subtraction(a, b);
          demo.multiplication(a, b);
        }
}
```

Output

The sum of the given numbers:30

The difference between the given numbers:10

The product of the given numbers:200

Example 2:

```
class Box {
            double width;
            double height;
            double depth;
            Box() {
            }
            Box(double w, double h, double d) {
                  width = w;
                  height = h;
                  depth = d;
            }
            void getVolume() {
                  System.out.println("Volume is : " + width * height * depth);
            }
      }

      public class MatchBox extends Box {
            double weight;
            MatchBox() {
        }
        MatchBox(double w, double h, double d, double m) {
                super(w, h, d);
                weight = m;
        }
        public static void main(String args[]) {
                MatchBox mb1 = new MatchBox(10, 10, 10, 10);
                mb1.getVolume();
                System.out.println("width of MatchBox 1 is " + mb1.width);
                System.out.println("height of MatchBox 1 is " + mb1.height);
                System.out.println("depth of MatchBox 1 is " + mb1.depth);
                System.out.println("weight of MatchBox 1 is " + mb1.weight);
        }
}
```

Controlling Access to Members of a Class

A class may be declared with the modifier public, in which case that class is visible to all classes everywhere. At the member level, you can also use the public modifier or no modifier (*package-private*) just as with top-level classes, and with the same meaning. For members, there are two additional access modifiers: private and protected. The private modifier specifies that the member can only be accessed in its own class. The protected modifier specifies that the member can only be accessed within its own package (as with *package-private*) and, in addition, by a subclass of its class in another package. If other programmers use your class, you want to ensure that errors from misuse cannot happen. Access levels can help you do this.

- Use the most restrictive access level that makes sense for a particular member.
  Use private unless you have a good reason not to.
- Avoid public fields except for constants. Public fields tend to link you to a particular implementation and limit your flexibility in changing your code.

```
class A
{
        private int a;
        public int getInt()
        {return a;}
}
class B
{
         private int b;
         void foo()
  {
          A myA = new A();
          b = myA .a;        //Wrong
          b = myA .getInt();  //Right
  }
}
```