MODULE 1: PROGRAMMING IN JAVA

Week 8

Most Essential Learning Competencies: *At the end of the course, you must be able to:*

1. Discuss the four fundamental object-oriented programming in Java
2. Apply encapsulation principles to a class
3. Describe how to implement inheritance and develop polymorphism

Reading Activity

The four fundamental OOPs are the following:

1. Inheritance
2. Polymorphism
3. Abstraction
4. Encapsulation

1. To implement encapsulation, you have to take the fields via public methods. The field cannot be used by anyone outside the class if it is declared private. As such, the fields are hidden within the class. **Encapsulation** is also known as ***"data hiding"*** because of this. Encapsulation avoids the code and data from being accessed by the other code defined external to the class. An interface controls access to the data and code. The ability to change the implemented code without interrupting the code of others is one of the advantages of encapsulation. With encapsulation, the code is easy to maintain, flexible and extensive.

   Example:

```
//Save as EncapTest.java
Public class EncapTest{
Private String stud_name;

public String getName(){
return stud_name;
        }
Public void setName(String stud_name){
This.stud_name=stud_name;
        }
}

//save as RunEncap.java
Class RunEncap{
Public static void main(String[] args){
EncapTestsname=new EncapTest();
Sname.setname("Alfie");
System.out.println(sname.getName());
        }
}
```

To compile, type:

```
Javac RunEncap.java EncapTest.java
```

To run, type:

```
Java RunEncap
```

2. Implementing **Inheritance** is the ***kind of software reuse.*** With inheritance a new class is constructed by holding an existing class' member and enhancing them with new or revised capabilities. Programmers save time in program creation by reusing verified and error-free software.

   During the creation of a class, programmers can define that the new class will inherit the members of an existing class. Existing class is known as the ***"superclass"*** while the new class is known as the ***"subclass".***

   A ***subclass*** is more definite that its superclass and signifies a more specialized group of objects because it normally adds its own fields and methods. The superclass from which the subclass openly inherits is called a direct superclass. Any class beyond the direct superclass in the hierarchy is called an ***"inherit class".***

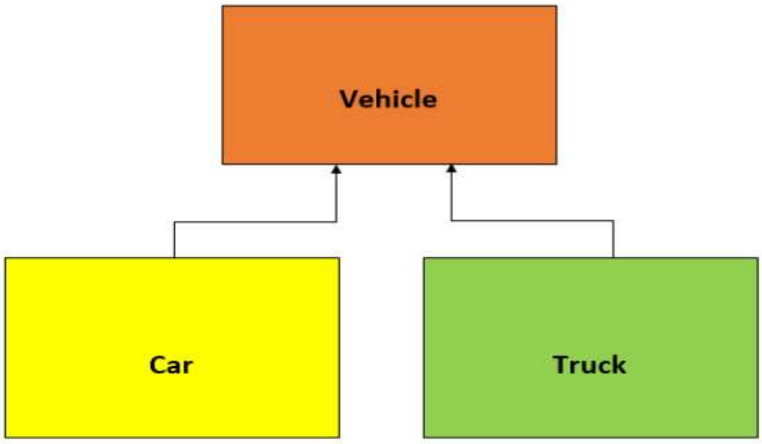

Figure 8.1 Implementing Inheritance

Syntax:

```
Class sub extends Super {

// methods

}
```

Example:

```
Class Worker{
        float wage=20000;
}
Class Carpenter extends Worker{
        Int incentive=5000;
        Public static void main(String args[]){
        Carpenter c=new Carpenter();
                System.out.println("Carpenter wage is:  "+c.wage);
                System.out.println("Incentive of Carpenter is:  "+c.incentive);
}
}
```

In this example, **Carpenter** is the ***subclass*** and **Worker** is the ***superclass***. The relationship between two classes is Carpenter IS-A Worker. Meaning to say, the Carpenter is a type of Worker.

3. Developing **Polymorphism** allows creating programs that process objects sharing the same superclass in a class hierarchy.

   We can construct and employ systems where new classes can be added with few or more alteration using polymorphism. The only parts of a program that should be modified to allow new classes are those that need direct knowledge of the new classes that the programmer adds. Polymorphism arises when programs call upon a method through a superclass variable at run time. The appropriate subclass version of the method is used, based on the type of reference stored in the superclass variable. Identical method name and signature can be utilized to initiate different actions to happen.

Example:

```
B obj = new B( );

//Parent class reference can be assigned to child object

A obj = new B( );
```

In this example, reference of class B can hold object of class B or an object of any subclass of class B. Class B extends A.

Read Additional Resources

1. **Advantages and Disadvantages of Object-Oriented Programming (OOP) PDF**
   https://resources.saylor.org/wwwresources/archived/site/wp-content/uploads/2013/02/CS101-2.1.2-AdvantagesDisadvantagesOfOOP-FINAL.pdf
2. **Java Abstraction PDF**
   https://www.tutorialspoint.com/java/pdf/java_abstraction.pdf
3. **Java Encapsulation PDF**
   https://www.tutorialspoint.com/java/pdf/java_encapsulation.pdf
4. **Java Inheritance PDF**
   https://www.tutorialspoint.com/java/pdf/java_encapsulation.pdf
5. **Java Polymorphism PDF**
   https://www.tutorialspoint.com/java/pdf/java_polymorphism.pdf

Watch Video Resources

1. **Five Basic Programming Concepts of Object-Oriented in Java**
   https://youtu.be/5ApeBo1LujE
2. **Using "super" keyword in constructors**
   https://youtu.be/b-KHSdgsYqY

Laboratory Activity

Laboratory Activity 8.1

1. Open IDE

2. Type the code below as shown

Application:

```
public class A
{
Public void methodx() //Base class method
        {
System.out.println ("This is methodX of class A");
        }
}
public class B extends A
{
Public void methodx()//Derived class method
        {
System.out.println("This is methodX of class B");
        }
{
public class C
{
Public static void main (Strings args []); {
        A obj1 = new A(); //Reference and object A
        A obj2 = new B(); //A reference but B object
obj1.methodx();
obj2.methodx();
        }
}
```

3. Save as **methodX.java**

4. If there are errors, debug the program. Compile it again. Do this repeatedly until no errors will be detected.

5. This will be the final output.

```
This is methodX of class A
This is methodX of class B
```

Insights

Insights 8.1

Instructions: *Write your answer on the Insights Sheet (IS) provided in this module.*

We wanted to know how deep your understandings about the topics presented in the module. Your answer will be graded accordingly using the prescribed rubrics. In your own words (at least 200 words):

1. As a student, how will you engage yourself in studying the concept of Object-Oriented Programming in Java and keeping yourself motivated all the time? (10-points)
2. Do you consider Object-Oriented Programming as a well-adopted programming style that uses interacting objects to model and solve complex programming tasks? Why? Justify your answer.(10-points)

Internet References

1. https://www.youtube.com/c/EJMedia1/about
2. https://www.youtube.com/c/snapppydev/about
3. https://www.youtube.com/
4. https://resources.saylor.org/
5. https://www.tutorialspoint.com
6. Phoenix Publishing House Inc.