## WIA1002/WIB1002 Data Structure

## Tutorial 1: Programming Fundamentals (Revision)

**Instruction:** Bring your solutions for all the questions below to your tutorial class. You might be asked to present your solutions to the class.

1. Write the definition of a class *Telephone* that contains:
    - An instance variable *areaCode*
    - An instance variable *number*
    - A static variable *numberOfTelephoneObject* that keeps track of the number of *Telephone* objects created
    - A constructor that accepts two arguments used to initialize the two instance variables
    - The accessor and mutator methods for *areaCode* and *number*
    - A method *makeFullNumber* that does not accept any argument, concatenates areaCode and number with a dash in between, and returns the resultant *String*.

    Write the statements to:

    - Instantiate 5 *Telephone* objects and store them in an array. Iterate through the array to print the full number of the 5 *Telephone* objects on the console. Your output should look as below:

    ```
    03-79676300
    03-79676301
    03-79676302
    03-79676303
    03-79676304
    ```

2. What is the output for the following? Explain.

```
class Person {
    public Person() {
        System.out.println("(1) Performs Person's tasks");
    }
}
class Employee extends Person {
    public Employee() {
        this("(2) Invoke Employee's overloaded constructor");
        System.out.println("(3) Performs Employee's tasks ");
    }
    public Employee(String s) {
        System.out.println(s);
```

```
    }
}
public class Faculty extends Employee {
    public Faculty() {
        System.out.println("(4) Performs Faculty's tasks");
    }
    public static void main(String[] args) {
        new Faculty();
    }
}
```

3. What is the output for the following? Explain.

```
public class C {
   public static void main(String[] args) {
    Object[] o = {new A(), new B()};
    System.out.print(o[0]);
    System.out.print(o[1]);
  }
}

 class A extends B {
    public String toString() {
        return "A";
     }
}

 class B {
     public String toString() {
        return "B";
    }
}
```

   a. AB
   b. BA
   c. AA
   d. BB

4. Write a class definition for an abstract class, *Vehicle*, that contains:

- a double instance variable, *maxSpeed*
- a protected double instance variable, *currentSpeed*
- a constructor accepting a double used to initialize the *maxSpeed* instance variable
- an abstract method, accelerate, that accepts no parameters and returns nothing.
- a method *getCurrentSpeed* that returns the value of *currentSpeed*
- a method *getMaxSpeed* that returns the value of *maxSpeed*
- a method *pedalToTheMetal*, that repeatedly calls accelerate until the speed of the vehicle is equal to *maxSpeed*. *pedalToTheMetal* returns nothing.

Can you create an instance of *Vehicle?*

5. Assume the existence of an interface, *Account*, with the following methods:
    - *deposit*: accepts an integer parameter and returns an integer
    - *withdraw*: accepts an integer parameter and return a Boolean

   Define a class, *BankAccount*, that implements the above interface and has the following members:
    - an instance variable named *balance*
    - a constructor that accepts an integer that is used to initialize the instance variable
    - an implementation of the *deposit* method that adds its parameter to the *balance* variable. The new balance is returned as the value of the method.
    - an implementation of the *withdraw* method that checks whether its parameter is less than or equal to the *balance* and if so, decreases the *balance* by the value of the parameter and returns *true*; otherwise, it leaves the *balance* unchanged and returns *false*.