COMP2026 Problem Solving Using Object Oriented Programming

Laboratory 12

Part A Discovery Exercises

Task 1: Abstract Class

a) Create the following abstract class in an IntelliJ project.

```
public abstract class ClassA {
    private int x;

    public ClassA(int x){
        this.x = x;
    }

    public void m1(){
        System.out.println("Hello!");
    }

    public abstract int m2(int n);
}
```

b) Create the following subclass in the same IntelliJ project.

```
public class ClassB extends ClassA {
    private double y;

    public ClassB(int x, double y){
        super(x);
        this.y = y;
    }
}
```

What is the error in ClassB? _____.

c) Add the `m2` method into Class B. Does the error exist? _____.

```
public class ClassB extends ClassA {
    private double y;

    public ClassB(int x, double y){
        super(x);
        this.y = y;
    }

    public int m2(int n) {
        return n;
    }
}
```

d) Remove the `m2` method. Change ClassB into an abstract class. Does the error exist? _____.

```
public abstract class ClassB extends ClassA {
    private double y;

    public ClassB(int x, double y){
        super(x);
        this.y = y;
    }
    //m2() removed
}
```

Task 2: More about Method Overriding

a) Create the following abstract class in an IntelliJ project.

```
public abstract class Shape {

    private String name;

    public Shape(String name)
    {
        this.name = name;
    }

    public String getName()
    {
        return name;
    }
}
```

b) Create the following subclass in the same IntelliJ project.

```
public class Circle extends Shape{

    private double radius;

    public Circle(String name, double radius)
    {
        super(name);
        this.radius = radius;
    }

    public double getArea()
    {
        return Math.PI * radius * radius;
    }
}
```

c) Create the following tester class in the same IntelliJ project.

```
public class ShapeTester {

    public static void main(String[] args) {

    }
}
```

d) Add the following statement to create a Shape object. Does it work? Why?_____.

```
public class ShapeTester {
    public static void main(String[] args) {

        Shape a = new Shape( name: "ShapeA");
    }
}
```

e) Remove the statement in (d). Add the following statement to create a Circle object and assign it to a Shape reference. Does it work? Why?_____.

```
public class ShapeTester {
    public static void main(String[] args) {
        //upcast the Circle object to parent type Shape
        Shape b = new Circle( name: "Circle1",  radius: 5);
    }
}
```

f) Add the following statement to call the `getArea()` method. Does it work? Why?_____.

```
public class ShapeTester {
    public static void main(String[] args) {
        //upcast the Circle object to parent type Shape
        Shape b = new Circle( name: "Circle1",  radius: 5);
        b.getArea();
    }
}
```

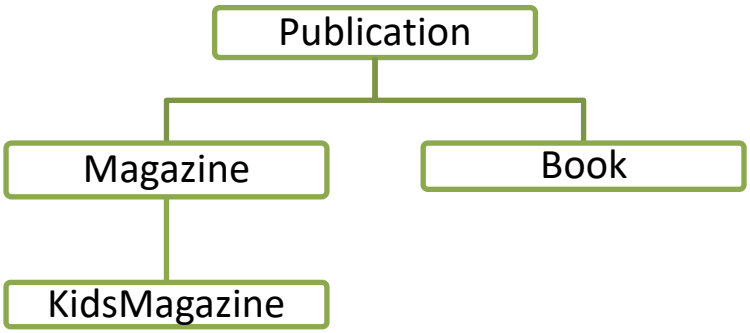
g) Add the following abstract method in the Shape class. Does part (f) work now? Why?_____.

```
public abstract class Shape {
    private String name;

    public Shape(String name)
    {
        this.name = name;
    }

    public String getName()
    {
        return name;
    }

    abstract double getArea();
}
```

Part B Programming Exercises

Task 1: Publication

Implement a class hierarchy consisting of Publication, Magazine, Book, and KidsMagazine classes as follows:



A Publication has a title and a publisher. The class should implement a **`print()`** method that displays all this information. The Publication class, being abstract cannot be instantiated. The **`print()`** method should print the information in the format:

```
Title: title
Publisher: publisher
```

- A Magazine is a kind of publication that has a volume number and an issue number. Magazine should override the **`print()`** method of Publication and display all the information. The **`print()`** method should print the information in the format:

  ```
  Title: title
  Publisher: publisher
  Volume: volumeNumber
  Issue: issueNumber
  ```

- A Book is a kind of publication that has an author. Book should also override the **`print()`** method of Publication. The **`print()`** method should print the information in the format:

  ```
  Title: title
  Publisher: publisher
  Author: author
  ```

- A KidsMagazine is a kind of magazine that has a recommended age range. Again, KidsMagazine should override the **`print()`** method. The **`print()`** method should print the information in the format:

  ```
  Title: title
  Publisher: publisher
  Volume: volumeNumber
  Issue: issueNumber
  Age Range: minimumAge - maximumAge
  ```

Task 2: BookInfo

Complete the following methods in the given BookInfo.java.

a) Write a method called **`addPublications()`** to create the following objects and add them to the array list.

Book:

Title	Publisher	Author
Cindy and the Candy Factory	AA Press	Ben Don
Secret Code	Ma House	Dim Green

Magazine:

Title	Publisher	Volume	Issue
Living	Person	5	3
Cooking	Person	3	10

KidsMagazine:

Title	Publisher	Volume	Issue	Age Range
Tinkering	Teens World	3	10	6-12
Tinkering	Teens World	3	11	6-12
Tinkering	Teens World	3	12	6-12
My Dream	Teens World	8	5	3-6

b) Write a method called **`showAllPublications()`** to print all the publications in the array list. Test the method in the `rupApp()`. Here is the sample output:

```
Title: Cindy and the Candy Factory
Publisher: AA Press
Author: Ben Don

Title: Secret Code
Publisher: Ma House
Author: Dim Green

Title: Living
Publisher: Person
Volume: 5
Issue: 3

Title: Cooking
Publisher: Person
Volume: 3
Issue: 10

Title: Tinkering
Publisher: Teens World
Volume: 3
Issue: 10
Age Range: 6 - 12
```

```
Title: Tinkering
Publisher: Teens World
Volume: 3
Issue: 11
Age Range: 6 - 12

Title: Tinkering
Publisher: Teens World
Volume: 3
Issue: 12
Age Range: 6 - 12

Title: My Dream
Publisher: Teens World
Volume: 8
Issue: 5
Age Range: 3 - 6
```

c) Write a method called **`showAllBooks()`** to print all the books in the array list. Test the method in the `rupApp()`. Here is the sample output:

```
Title: Cindy and the Candy Factory
Publisher: AA Press
Author: Ben Don

Title: Secret Code
Publisher: Ma House
Author: Dim Green
```

d) Write a method called **`showAllMagazines()`** to print all the magazines in the array list. Test the method in the `rupApp()`. Here is the sample output:

```
Title: Living
Publisher: Person
Volume: 5
Issue: 3

Title: Cooking
Publisher: Person
Volume: 3
Issue: 10

Title: Tinkering
Publisher: Teens World
Volume: 3
Issue: 10
Age Range: 6 - 12
```

```
Title: Tinkering
Publisher: Teens World
Volume: 3
Issue: 11
Age Range: 6 - 12

Title: Tinkering
Publisher: Teens World
Volume: 3
Issue: 12
Age Range: 6 - 12

Title: My Dream
Publisher: Teens World
Volume: 8
Issue: 5
Age Range: 3 - 6
```

e) Write a method called **`showAllKidsMagazines()`** to print all the kids magazines in the array list. Test the method in the `rupApp()`. Here is the sample output:

```
Title: Tinkering
Publisher: Teens World
Volume: 3
Issue: 10
Age Range: 6 - 12

Title: Tinkering
Publisher: Teens World
Volume: 3
Issue: 11
Age Range: 6 - 12

Title: Tinkering
Publisher: Teens World
Volume: 3
Issue: 12
Age Range: 6 - 12

Title: My Dream
Publisher: Teens World
Volume: 8
Issue: 5
Age Range: 3 - 6
```

f) Write a method called **`showKidsMagazineByAge(int age)`** to print all the kids magazines that are suitable for the specified age. Test the method in the `rupApp()`. Here are the sample outputs:

`showKidsMagazineByAge(6)`:

```
Title: Tinkering
Publisher: Teens World
Volume: 3
Issue: 10
Age Range: 6 - 12

Title: Tinkering
Publisher: Teens World
Volume: 3
Issue: 11
Age Range: 6 - 12

Title: Tinkering
Publisher: Teens World
Volume: 3
Issue: 12
Age Range: 6 - 12

Title: My Dream
Publisher: Teens World
Volume: 8
Issue: 5
Age Range: 3 - 6
```

`showKidsMagazineByAge(5)`:

```
Title: My Dream
Publisher: Teens World
Volume: 8
Issue: 5
Age Range: 3 - 6
```

g) Write a method called **`showPuhlicationByTitle(String title)`** to print all the publications that are having the specified title. Test the method in the `rupApp()`. Here are the sample outputs:

`showPublicationByTitle("Secret Code")`:

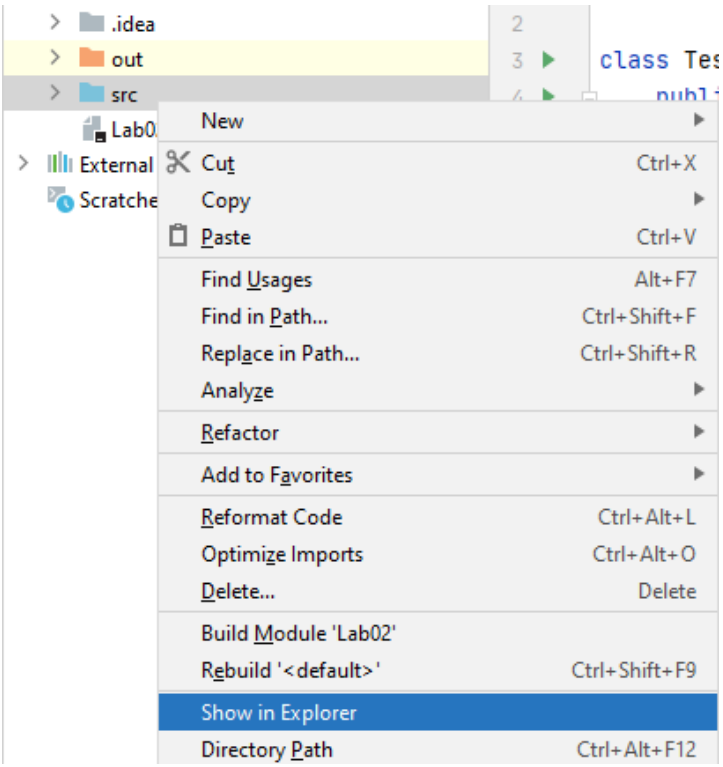
```
Title: Secret Code
Publisher: Ma House
Author: Dim Green
```

`showPublicationByTitle("Tinkering")`:

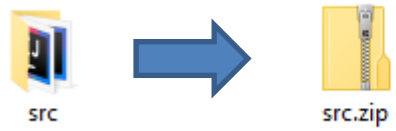
```
Title: Tinkering
Publisher: Teens World
Volume: 3
Issue: 10
Age Range: 6 - 12

Title: Tinkering
Publisher: Teens World
Volume: 3
Issue: 11
Age Range: 6 - 12

Title: Tinkering
Publisher: Teens World
Volume: 3
Issue: 12
Age Range: 6 - 12
```

Part C Submitting Exercises

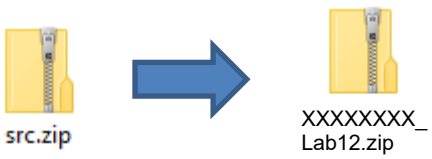
Step 1: Right-click the **src** folder and select **Show in Explorer**



Step 2: Zip the **src** folder into **src.zip**



Step 3: Rename the src.zip file to **XXXXXXXX_lab12.zip** where **XXXXXXXX** is your **student id**



Step 4: Submit **XXXXXXXX_lab12.zip** and **XXXXXXXX_lab12.docx** to Moodle.

References

[1] Bravaco, R., & Simonson, C. (2009). *Java programming: From the ground up*. *Dubuque*, IA: McGraw-Hill.

[2] Dean, J., & Dean, R. (2008). *Introduction to programming with Java: A problem solving approach*. Boston: McGraw-Hill.

[3] Farrell, J. (2012). ***Java programming. Boston, MA: Course Technology Cengage Learning***

[4] Forouzan, B. A., & Gilberg, R. F. (2007). *Computer science: A structured programming approach using C (3rd ed.)*. Boston, MA: Thomson Course Technology.

[5] Gaddis, T. (2016). *Starting out with Java (6th ed.)*. Pearson.

[6] Liang, Y. D. (2013). *Introduction to Java programming: Comprehensive version*. (8th ed.). Pearson.

[7] Schildt, H. (2006). *Java a beginner's guide*. New York: McGraw Hill.

[8] Schildt, H., & Skrien, D. J. (2013). Java programming: A comprehensive introduction. New York: McGraw-Hill.

[9] Wu, C. T. (2010). *An introduction to object-oriented programming with Java*. Boston: McGraw Hill Higher Education

[10] Xavier, C. (2011). *Java programming: A practical approach*. New Delhi: Tata McGraw Hill.

[11] yet another insignificant Programming Notes. (n.d.). Retrieved from https://www3.ntu.edu.sg/home/ehchua/programming

[12] Zakhour, S., Kannan, S., & Gallardo, R. (2013). *The Java tutorial: A short course on the basics (5th ed.)*.