Software Construction and Development

Lab 2

Objectives

- Classes, Abstract Classes, Interfaces
- Inheritance & Polymorphism
- Arrays & Strings

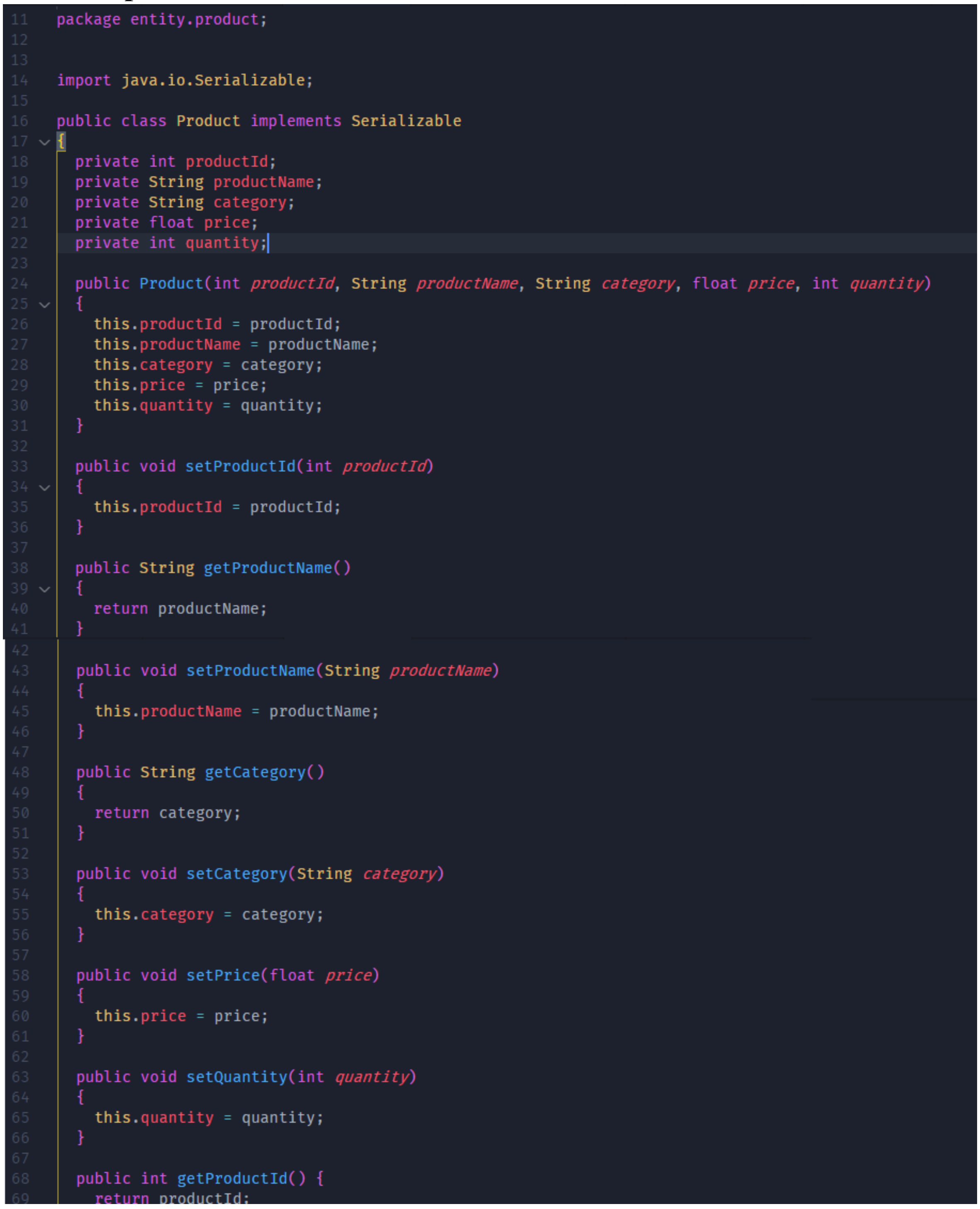
Instructions

- **A very important lab for your rest of the course, work hard on it!**
- **Deadline for submission: 11:45 AM**

Code Style

- Meaningful variable names
- **Camel case** for variable & function names, e.g. **calculateTotalBill**
- **Pascal case** for Class, Interface, Enum names e.g. **PromotionCalculator**
- Interfaces can be named like e.g. interface: **IPromotionCalculator**
- Proper indentation and formatting, use auto formatter of IDE
- No random comments, multiple empty lines or commented code
- Define proper constructor, getter/setter and access modifiers (public, private etc)

An Example Java Class is attached:



```java
package entity.product;


import java.io.Serializable;

public class Product implements Serializable
{
  private int productId;
  private String productName;
  private String category;
  private float price;
  private int quantity;

  public Product(int productId, String productName, String category, float price, int quantity)
  {
    this.productId = productId;
    this.productName = productName;
    this.category = category;
    this.price = price;
    this.quantity = quantity;
  }

  public void setProductId(int productId)
  {
    this.productId = productId;
  }

  public String getProductName()
  {
    return productName;
  }

  public void setProductName(String productName)
  {
    this.productName = productName;
  }

  public String getCategory()
  {
    return category;
  }

  public void setCategory(String category)
  {
    this.category = category;
  }

  public void setPrice(float price)
  {
    this.price = price;
  }

  public void setQuantity(int quantity)
  {
    this.quantity = quantity;
  }

  public int getProductId() {
    return productId;
```

Project Format

- Create an empty project on **Eclipse IDE, named: Lab2Tasks**
- Try to separate your classes in meaningful packages
- Have a Main class in root package of the project, this should be your driver class

Submission

- Create a submission folder named as RollNumber_Lab2
- Create a folder for each problem (if it has multiple parts separate those as e.g. Problem2_Part1)
- Place all the code (.java) files for each problem in it folder
- Zip your the submission folder RollNumber_Lab2, submit the Zip file on Google Classroom

Problem 1: Inheritance

Imagine a publishing company that markets both **book** and **audiocassette** versions of its works. Create an abstract class **Publication** that stores the **title** (a string) and **price** (type float) of a **publication**.

In **Publication** class add a public method as:

public void displayDetails();
// This method should properly display attributes of the publication i.e. Title & price

From this class derive two classes: **Book**, which adds a **pageCount** (type int), and **Tape**, which adds a duration **timeInMinutes** (type float).

Book & Tape both should override the public **void displayDetails()** method of **Publication** class, and do following:

- call the super method in it like **super();** super() method calls the parent class (Publication class) displayDetails() method.
- Display the attribute added in **Book & Tape** classes respectively

Create a **Main** class that would be your driver class, implement the following methods (prototypes given):

private Book createABookFromUserInput(); Take input from user for the fields needed to create a **Book**, set the fields using setters return: return a new Book object created from the user input
private Tape createATapeFromUserInput(); Take input from user for the fields needed to create a **Tape**, set the fields using setters return: return a new Tape object created from the user input
public static void main(String[] args);

- Create a Main class object here, and call the method created above using this instance; as main(String[] args) is static function we cannot call non-static functions from here
- Create a **Book** type publication using the createABookFromUserInput() method
- Create a **Tape** type publication using the createATapeFromUserInput() method
- e.g. Book myBook = driver.createABookFromUserInput();
- Finally, use the displayDetails() method of the created publications to display data

Second Part (Polymorphism):
Change the above program to create a **Book** and **Tape** as following:

Publication myBook = driver.createABookFromUserInput();

Similarly, do it for tape. Does it give the same output? Do you understand this?

If it gives error, try this:
Publication myBook = (Book) driver.createABookFromUserInput(); // casting, similarly do it for **Tape, it casting the child object to parent class reference**

Does it work? Output should be the same in both cases. Attach output of both cases.