Java Fundamentals
Section 4: Creating an Inventory Project Project

Instructions:

1. **For the first part of the project you are required to think about what your inventory system will store.**
   a. **Think of specific products that lend themselves to be stored in an inventory (for example, products in your home, school, or workplace: they could be from the following categories; office supplies, music CDs, DVD movies, or software). Write a list of at least 6 products that you want to store in your system, this project could be used to store a wide range of products.**
   b. **For each of the products that you identified, complete the following table:**

Attribute	Sample Data
Name of the product (the value that will identify the product in your system).	
Price (this value holds the price that each item will be sold for).	
Number of units in stock (this value will store how many of each product item is currently in stock).	
Item number (used to uniquely identify the product in your system).	

This table gives you an understanding of the type of data that you will want to store for the attributes of each product. It's useful to do this so you have a clear understanding of the data that you will be working with!

2. **The next step is to think about the correct data types that you will use to store the values in your system. To do this add another column to your table that will identify the correct data type for each value that you have identified.**

Attribute	Sample Data	Data Type
Name of the product		
price		

Number of units in stock		
Item number		

3. **Create a project named inventory.**
4. **Create an object class called Product.**
5. **Add the following private instance fields (variables) by using the data types you identified in task 2:**
    a. **item number**
    b. **the name of the product**
    c. **the number of units in stock**
    d. **the price of each unit**
6. **Add a comment above the instance field declarations that states:**
    //Instance field declarations
7. **Create two constructors:**
    a. **A default constructor without parameters that will allow the compiler to initialize the fields to their default values.**
    Add a comment above your constructor that explains the purpose of the code.
    b. **Overload the default constructor by creating a constructor with parameters for all four of the class' instance fields so that they can be initialized with values from the driver class. The parameters should be named; number, name, qty, price. You should use the this.instance_field_name notation to quantify the objects instance field:**

```
this . name  name ;
```

8. **Write getter/accessor and setter/mutator methods for each of the four instance variables. Write getter/accessor and setter/mutator methods for each of the four instance variables. Add comments above them to explain their purpose.**
9. **Override the toString() method from the object class to show a description of each Product object that includes the instance field values in the following format:**

```
Item Nun-ber        1
 Name              Greatest Hits
  Quantity in stock: 25
                   9 . 99
```

10. Create a Java main class called ProductTester.

11 . Create and initialize six Product objects based on the list that you created in task 1 .

a. **Two of the Products should be created using the default constructor.**
b. **The other four should be created by providing values for the arguments that match the parameters of the constructor.**

12. using the ProductTester class, display the details of each product to the console.

13. Save your project.

A. a. List of ProductsLaptop

1. Smartphone
2. Printer
3. Office Chair
4. Desk Lamp
5. External Hard Drive

b. Attributes Table

Attribute	**Sample Data**	**Data Type**
Name of the product	Laptop	String
Price	999.99	double
Number of units in stock	50	int
Item number	101	int

Part 2: Data Types

Attribute	**Sample Data**	**Data Type**
Name of the product	Laptop	String
Price	999.99	double
Number of units in stock	50	int
Item number	101	int

Part 3: Creating the Project

Project Name: inventory

Part 4: Creating the Product Class

Product.java

```
public class Product {
   // Instance field declarations
   private int itemNumber;
   private String name;
   private int quantityInStock;
   private double price;
```

```
// Default constructor
public Product() {
   // Purpose: To initialize the fields to their default values
   this.itemNumber = 0;
   this.name = "";
   this.quantityInStock = 0;
   this.price = 0.0;
}

// Parameterized constructor
public Product(int number, String name, int qty, double price) {
   this.itemNumber = number;
   this.name = name;
   this.quantityInStock = qty;
   this.price = price;
}

// Getter and Setter methods
// Purpose: To get and set the value of itemNumber
public int getItemNumber() {
   return itemNumber;
}

public void setItemNumber(int itemNumber) {
   this.itemNumber = itemNumber;
}

// Purpose: To get and set the value of name
public String getName() {
   return name;
}

public void setName(String name) {
   this.name = name;
```

```
    }

    // Purpose: To get and set the value of quantityInStock
    public int getQuantityInStock() {
        return quantityInStock;
    }

    public void setQuantityInStock(int quantityInStock) {
        this.quantityInStock = quantityInStock;
    }

    // Purpose: To get and set the value of price
    public double getPrice() {
        return price;
    }

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

    // Override toString() method
    @Override
    public String toString() {
        return "Item Number: " + itemNumber + "\n" +
               "Name: " + name + "\n" +
               "Quantity in stock: " + quantityInStock + "\n" +
               "Price: " + price;
    }
}
```

Part 5: Creating the ProductTester Class

ProductTester.java

```
public class ProductTester {
    public static void main(String[] args) {
        // Creating Product objects using default constructor
        Product product1 = new Product();
        Product product2 = new Product();

        // Creating Product objects using parameterized constructor
        Product product3 = new Product(101, "Laptop", 50, 999.99);
        Product product4 = new Product(102, "Smartphone", 30, 699.99);
        Product product5 = new Product(103, "Printer", 20, 199.99);
        Product product6 = new Product(104, "Office Chair", 15, 149.99);

        // Displaying product details
        System.out.println(product1);
        System.out.println();
        System.out.println(product2);
        System.out.println();
        System.out.println(product3);
        System.out.println();
        System.out.println(product4);
        System.out.println();
        System.out.println(product5);
        System.out.println();
        System.out.println(product6);
    }
}
```