**CSC202 – Spring 22-23 – Assignment 2**

|  |  |
| --- | --- |
| **Objective and Overview** | **Marks: out of 5 marks (5%)** |
| ▪ To apply the object-oriented programming concepts of inheritance and polymorphism. | |
| **Grading Rules - rubric (See end of document for more details)** | **Individual Submissions** |
| * **Functionality:** full and correct functionality of the system. * **Use of OOP concepts:** correct use of OOP concepts such as inheritance, polymorphism, and association. * **Error handling and validation:** you need to validate user input and account for any exceptions and errors. * **All the above needs to be supported with Test Cases – Test Cases are explained at the end of the document.** | **4 marks** |
| * **Code organization and structure:** Evaluate the overall organization and structure of the code, ensuring it is modular, easy to read, and follows best practices. This includes proper indentation, use of whitespace, and appropriate commenting. * **Code explanation and documentation:** Assesses your ability to explain your code and design choices, through in-code comments and a separate document outlining your thought process, challenges faced, and the rationale behind your implementation choices. | **1 marks** |
| **Submission Rules** | |
| * This is not a group assignment; only individual submissions are accepted. * Export your Java program into a zipped folder and submit the full code on the blackboard along with a full report detailing all the implemented classes with full descriptions supported by screenshots. * **Due date: 20th May 2023** * **Late submissions will be penalized by 0.5 marks per day after the deadline.** * **Plagiarized work will be reported to OAI.** | |

* **Overview**

Your task is to create a Java program that models different types of eco-friendly products using inheritance. Your program should have a superclass called "Product" that contains common attributes of eco-friendly products such as name, brand, price, type, and eco-friendliness. You will also create five derived classes: "ClothingProduct," "SkincareProduct," "CleaningProduct," "ElectronicProduct," and "PlantProduct," each with their own unique attributes.

Additionally, you will create a class called "ProductInventory" that will manage an array of products. This class will have methods to add, remove, and search for products based on name, brand, or price.

* **Objectives**
* Create a superclass with common attributes of eco-friendly products
* Create derived classes with unique attributes and override a method to display all attributes
* Create a class to manage an array of products with methods to add, remove, and search for products
* Utilize inheritance to create derived classes and encapsulation to protect attributes of the classes
* **Requirements**

Your Java program should have the following classes:

* **Product**
* Private attributes: Name (string), Brand (string), Price (float), Type (string), Eco-Friendly (boolean)
* Public getters and setters for all attributes, and print\_info() method that prints the values of all the attributes.
* **ClothingProduct**
* Inherits from Product
* Additional private attributes: Size (string), Material (string), Color (string), Style (string)
* Public getters and setters for all attributes.
* Override the print\_info() method to display all attributes.
* **SkincareProduct**
* Inherits from Product
* Additional private attributes: SkinType (string), TargetConcerns (array of strings), FragranceFree (boolean)
* Public getters and setters for all attributes
* Override the print\_info() method to display all attributes
* **CleaningProduct**
* Inherits from Product
* Additional private attributes: SurfaceType (string), Scent (string), Biodegradable (boolean)
* Public getters and setters for all attributes
* Override the print\_info() method to display all attributes
* **ElectronicProduct**
* Inherits from Product
* Additional private attributes: PowerSource (string), EnergyEfficient (boolean), SmartHomeCompatible (boolean)
* Public getters and setters for all attributes
* Override the print\_info() method to display all attributes
* **PlantProduct**
* Inherits from Product
* Additional private attributes: PotType (string), SunlightNeeds (string), TypeOfPlant (string)
* Public getters and setters for all attributes
* Override the print\_info() method to display all attributes
* **ProductInventory**
* Private attribute: Products (array of Product objects)
* Public methods:
  + add\_product(product): adds a Product object to the array of products
  + remove\_product(product): removes a Product object from the array of products
  + search\_product(search\_term): searches for a Product object in the array of products based on the product name and returns the full product information. If the product is not in the store then it should notify the user.
* All classes should implement default and overloaded constructors.

**Your program should also allow the user to perform the following actions:**

* Display all eco-friendly products of a specific type (clothing, skincare, cleaning, electronic, or plant) in the product inventory, including all their attributes, using the print\_info() method.

**Test your program using the following sample data:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Product Name** | **Brand** | **Price** | **Type** | **Eco-Friendly** | **Additional Attributes** |
| Eco T-Shirt | OrganicWear | 25.99 | Clothing | Yes | Size: M, Material: Organic Cotton, Color: Green, Style: Casual |
| Eco Jeans | Levi's | 89.99 | Clothing | Yes | Size: 32/32, Material: Recycled Denim, Color: Blue, Style: Slim Fit |
| Shampoo | Pantene | 8.99 | Skincare | No | Skin Type: Normal, Target Concerns: None, Fragrance Free: No |
| Moisturizer | Olay | 12.49 | Skincare | No | Skin Type: Sensitive, Target Concerns: Wrinkles, Fragrance Free: No |
| All-Purpose Cleaner | Clorox | 4.99 | Cleaning | No | Surface Type: Multi-Surface, Scent: Lavender, Biodegradable: No |
| Dish Soap | Palmolive | 2.99 | Cleaning | No | Surface Type: Dishes, Scent: Original, Biodegradable: No |
| Smart Speaker | Amazon | 99.99 | Electronic | No | Power Source: AC, Energy Efficient: Yes, Smart Home Compatible: Yes |
| Laptop | Dell | 799.99 | Electronic | No | Power Source: AC, Energy Efficient: No, Smart Home Compatible: No |
| Snake Plant | Plants R Us | 24.99 | Plant | Yes | Pot Type: Ceramic, Sunlight Needs: Low, Type Of Plant: Succulent |
| Fiddle Leaf Fig | Green Thumb | 49.99 | Plant | Yes | Pot Type: Terracotta, Sunlight Needs: High, Type Of Plant: Tree |
| T-Shirt | H&M | 12.99 | Clothing | No | Size: L, Material: Cotton, Color: Black, Style: Basic |
| Skirt | Forever 21 | 29.99 | Clothing | No | Size: S, Material: Polyester, Color: Red, Style: A-Line |
| Face Wash | Neutrogena | 6.99 | Skincare | No | Skin Type: Oily, Target Concerns: Acne, Fragrance Free: No |
| Sunscreen | Banana Boat | 9.99 | Skincare | No | Skin Type: All, Target Concerns: Sun Protection, Fragrance Free: No |
| Glass Cleaner | Windex | 3.99 | Cleaning | No | Surface Type: Glass, Scent: None, Biodegradable: No |
| Laundry Detergent | Tide | 12.99 | Cleaning | No | Surface Type: Clothing, Scent: Fresh, Biodegradable: No |
| Smart Thermostat | Nest | 249.99 | Electronic | Yes | Power Source: AC, Energy Efficient: Yes, Smart Home Compatible: Yes |
| Smartwatch | Apple | 399.99 | Electronic | No | Power Source: Battery, Energy Efficient: Yes, Smart Home Compatible: Yes |
| Succulent | Plants R Us | 12.99 | Plant | No | Pot Type: Plastic, Sunlight Needs: Low, Type Of Plant |

**You need to run at least all the following test cases and show related code and the desired output:**

1. Test Case 1: Test add\_product() function

* Input: A new Product object
* Action: Call add\_product() method with the new Product object
* Expected Output: The Products array in ProductInventory object should contain the new Product object

1. Test Case 2: Test remove\_product() function

* Input: An existing Product object in the Products array
* Action: Call remove\_product() method with the existing Product object
* Expected Output: The Products array in ProductInventory object should not contain the existing Product object

1. Test Case 3: Test search\_product() function with existing product

* Input: A string that matches an existing Product’s Name attribute
* Action: Call search\_product() method with the input string
* Expected Output: The full information of the existing Product object that matches the input string should be returned

1. Test Case 4: Test search\_product() function with non-existing product

* Input: A string that does not match any existing Product’s Name attribute
* Action: Call search\_product() method with the input string
* Expected Output: A message should be returned to notify the user that the product is not found in the store

1. Test Case 5: Test print\_info() function for ClothingProduct object

* Input: A ClothingProduct object
* Action: Call print\_info() method with the ClothingProduct object
* Expected Output: All the attributes of the ClothingProduct object should be printed to the console

1. Test Case 6: Test print\_info() function for SkincareProduct object

* Input: A SkincareProduct object
* Action: Call print\_info() method with the SkincareProduct object
* Expected Output: All the attributes of the SkincareProduct object should be printed to the console

1. Test Case 7: Test print\_info() function for CleaningProduct object

* Input: A CleaningProduct object
* Action: Call print\_info() method with the CleaningProduct object
* Expected Output: All the attributes of the CleaningProduct object should be printed to the console

1. Test Case 8: Test print\_info() function for ElectronicProduct object

* Input: An ElectronicProduct object
* Action: Call print\_info() method with the ElectronicProduct object
* Expected Output: All the attributes of the ElectronicProduct object should be printed to the console

1. Test Case 9: Test print\_info() function for PlantProduct object

* Input: A PlantProduct object
* Action: Call print\_info() method with the PlantProduct object
* Expected Output: All the attributes of the PlantProduct object should be printed to the console

1. Test Case 10: Test print\_info() function for all eco-friendly ClothingProduct objects

* Input: A string “clothing”
* Action: Call print\_info() method with all eco-friendly ClothingProduct objects in the Products array and the input string
* Expected Output: All the attributes of the eco-friendly ClothingProduct objects should be printed to the console

1. Test Case 11: Test print\_info() function for all eco-friendly SkincareProduct objects

* Input: A string “skincare”
* Action: Call print\_info() method with all eco-friendly SkincareProduct objects in the Products array and the input string
* Expected Output: All the attributes of the eco-friendly SkincareProduct objects should be printed to the console

1. Test Case 12: Test print\_info() function for all eco-friendly CleaningProduct objects

* Input: A string “cleaning”
* Action: Call print\_info() method with all eco-friendly CleaningProduct objects in the Products array and the input string
* Expected Output: All the attributes of

*You may run additional test cases to demonstrate any additional functionality that you have implemented in your program.*