RAJIV GANDHI UNIVERSITY OF KNOWLEDGE TECHNOLOGIES, BASAR
Basar, Nirmal (Dist), Telangana - 504107

(A.Y. 2023-2024)

Branch: Computer Science and Engineering

R19

Subject Name:	**Object Oriented Programming Lab**		**Subject Code:**	
Date & Class :	**14/12/23- C1**		**Time:**	**180 Minutes**
Exam:	**Lab Internal**	**SET NO : I**	**Max. Marks:**	**40M**

Note:

i. Answer any two questions, Each question carries **20 marks.**

1. Suppose there is a Kennel offering three dogs for adoption: snoopy, rocky, and snowy, each with specific characteristics. Develop a Java program to assist potential buyers in selecting a dog based on their preferences.

Consider the following details for the dogs:

- snoopy:
  - Breed: Pomeranian
  - Color: White
  - Height: 1 ft
  - Type: Guard dog
- rocky:
  - Breed: Labrador
  - Color: Brown
  - Height: 3 ft
  - Type: Sniffer dog
- snowy:
  - Breed: German Shepherd
  - Color: Black
  - Height: 4 ft
  - Type: Shepherd dog

Create a Java program that prompts the user to input preferred dog characteristics such as breed, type, and height. Then, based on these inputs, suggest a suitable dog from the kennel (i.e., snoopy, rocky, or snowy) to the user.

Class Name: Dog Data Fields:

- name: String
- breed: String
- color: String
- height: double
- type: String Constructors:
- Dog() Methods:

- getBreed()
- getName()

Implement the program logic to compare user inputs with the available dog data and suggest the most suitable dog based on the input criteria.

Test the program by taking user inputs and displaying the name of the recommended dog from the kennel.

2. Design an E-Book stall management system using Java and Object-Oriented Programming concepts. Each book at the stall possesses properties such as Book Name, Book Author, and Book Count. Additionally, customers visiting the stall are identified by properties like Customer ID, Customer Name, and Customer Address. Customers can purchase books from this E-Book stall.

Write a Java program that performs the following:

1. Create a Book class with attributes:
   - bookName: Represents the name of the book.
   - bookAuthor: Represents the author of the book.
   - bookCount: Represents the count or quantity of available books.
2. Create a Customer class with properties:
   - customerID: Represents the unique identification number of the customer.
   - customerName: Represents the name of the customer.
   - customerAddress: Represents the address of the customer.
3. Implement functionality in the main program to simulate a customer purchasing a book from the E-Book stall. Upon the purchase of a book, display the name of the book along with the remaining count of that particular book.
4. Additionally, when prompted to search by the name of a customer, display the details of the text books they bought along with the respective counts.

Your program should demonstrate the concept of buying books from the E-Book stall by customers. When a customer purchases a book, it should correctly display the book's name and the updated count of remaining books of that type. Also, when searching by customer name, the program should display the text books bought and their respective counts by that customer.

Use appropriate object-oriented principles such as encapsulation, classes, and objects to organize the functionality effectively.

Test the program thoroughly by simulating customer purchases and display the book details with the updated count after each purchase. Additionally, ensure the customer search functionality displays the text books bought and their respective counts for the specified customer.

3. Create an Online Shopping Cart system in Java utilizing interfaces and object-oriented programming principles.

1. Define an interface named Product with methods:
   - getProductName(): Retrieves the name of the product.
   - getPrice(): Retrieves the price of the product.

2. Implement classes Book and Electronic that implement the Product interface:
    - Book and Electronic should have attributes such as productName, price, author (for Book), manufacturer (for Electronic), etc.
    - Implement methods to return the name and price of each product.
3. Create a class named ShoppingCart with functionality to:
    - Maintain a list of added products.
    - Add products to the cart.
    - Calculate and display the total price of the items in the cart.
4. Demonstrate the program by simulating shopping scenarios:
    - Add various books and electronics to the shopping cart.
    - Display the list of products and the total price.

Sample output :

```
The Catcher in the Rye added to cart.
Laptop added to cart.
To Kill a Mockingbird added to cart.

Products in the Shopping Cart:
Name: The Catcher in the Rye, Price: $25.0
Name: Laptop, Price: $899.99
Name: To Kill a Mockingbird, Price: $20.0
Total Price: $944.99
```

* * * * *