SmartCliff Learning Solutions

KIOT-SDE-Readiness-Module I- Assessment 02

Topic: Object oriented Programming, Exception Handling, Generics and collection

Date: 23/05/2024

Time: 03:30 PM – 05:30 PM

Total Marks: 50 Marks

PART A: Multiple Choice Questions (10 X 1 = 10)

PART B: Programming Questions

Implement the Java Code for the following scenario.

1. Write a Java program that prompts the user to enter a secret key for a lock. The key must be an isogram, meaning no letter occurs more than once in the key. If the key is not an isogram, the program should throw a custom exception named NotAnIsogramException. The program should inform the user and ask for another secret key until a valid isogram is entered.

   Sample Input1: **(6 Marks)**

   Secret Key for the Lock: Anees

   Sample Output1:

   The secret key is not an Isogram

   Sample Input2:

   Secret Key for the Lock: Computer

   Sample Output2:

   The secret key is an Isogram.

2. A banking application that involves different types of accounts, such as savings accounts and checking accounts. Both types of accounts share a common attribute, which is the balance, but checking accounts have an additional specific attribute called the overdraft limit.

   To design the classes for these accounts we will define two interfaces**: IBankAccount** and **ICheckingAccount.**

   The **IBankAccount** interface will have a property called Balance, which represents the account balance. It will be implemented by both the savings account and checking account classes.The **ICheckingAccount** interface will have a property called OverdraftLimit, which represents the maximum negative balance that a checking account can have. It will be implemented only by the checking account class.

Using these interfaces, we can achieve multiple inheritances by implementing both interfaces in the appropriate classes **SavingAccount** and **CheckingAccount**, and display the updated balance after the transaction. **(8 Marks)**

Sample Input:

Savings Account:

Initial Balance: ₹2000

Checking Account:

Initial Balance: ₹5000

Overdraft Limit: ₹1000

Performing Transactions:

Deposit ₹500 to the Savings Account

Withdraw ₹2000 from the Checking Account

3. Counting Good Pairs in Event Participant IDs **(8 marks)**

You are organizing a large event where each participant is assigned a unique ID number. However, due to a registration system glitch, some participants ended up receiving the same ID. As a result, you have a list of participant IDs and you need to count how many pairs of participants have the same ID. These pairs are called "good pairs."A pair (i, j) is called good if the IDs are the same and i < j.Task is to Given a list of participant IDs, return the number of good pairs.

Sample Input and Output 1:

Event: Tech Conference

Input: participant_ids = [1, 2, 3, 1, 1, 3]

Output: 4

Explanation: There are 4 good pairs: (0,3), (0,4), (3,4), (2,5) (0-indexed).

Sample Input and Output 2:

Event: Charity Run

Input: participant_ids = [1, 1, 1, 1]

Output: 6

Explanation: Each pair in the array is good: (0,1), (0,2), (0,3), (1,2), (1,3), (2,3).

Sample Input and Output 3:

Event: Workshop

Input: participant_ids = [1, 2, 3]

Output: 0

Explanation: There are no good pairs since all IDs are unique.

Constraints:

1 <= participant_ids.length <= 100

1 <= participant_ids[i] <= 100

4. Detecting Palindromic Permutations in Usernames **(8 marks)**

You are part of the cybersecurity team for a social media platform. Your task is to ensure that usernames can be checked for potential palindromic permutations. A username is considered to have a palindromic permutation if you can rearrange its letters to form a palindrome. A palindrome is a string that reads the same forwards and backwards. Task is given a username, determine if any permutation of the username can form a palindrome.

Sample Input and Output 1 :

UserID: user123

Input: name = "code"

Output: false

Explanation: In this case, no permutation of the string "code" can form a palindrome. Palindromes are strings that read the same forwards and backwards. Since "code" cannot be rearranged into a palindrome (no matter how you shuffle its letters), the output is false.

Sample Input and Output 2:

UserID: user321
Input: name = "aab"
Output: true
Explanation: Here, "aab" can be rearranged into "aba", which is a palindrome. By permuting the letters of "aab", we can form a palindrome, so the output is true.

Sample Input and Output 3:

UserID: user431
Input: name = "carerac"
Output: true
Explanation: The string "carerac" can be rearranged into "racecar", which is a palindrome. Even though the letters are not in palindrome order in the original string, they can be rearranged to form one. So, the output is true.

Constraints:

1 <= s.length <= 5000
s consists of only lowercase English letters.

5. You have been tasked with developing an inventory management system for an online store. The system should effectively store and manage product information, including the product's name, SKU (Stock Keeping Unit), quantity available, and price. To achieve this, you decide to implement a solution using an ArrayList data structure to handle real-time inventory operations.
   Here's a detailed implementation of the system:
   I. Start by defining a Product class that represents a product in the inventory. The class should have attributes such as the product's name, SKU, quantity, and price.
   II. In the main inventory management class, create an ArrayList to store the products. This ArrayList will serve as the container for all product entries.
   III. Implement a method to add a new product to the inventory. This method should take the necessary parameters (name, SKU, quantity, and price) and create a new Product object. Then, add the object to the ArrayList.
   IV. Implement a method to remove a product from the inventory. This method should take the SKU of the product as a parameter. Iterate over the ArrayList and search for a product with a matching SKU. If found, remove the product from the ArrayList.
   V. Create a method to update the details of a product in the inventory. This method should take the SKU as a parameter and allow for modifying the product's attributes such as quantity or price. Iterate over the ArrayList to find the product with the specified SKU and update its details accordingly.
   VI. Develop a method to search for a product in the inventory. This method should take a search query (SKU or name) as a parameter. Iterate over the ArrayList, comparing the query with the SKU or name of each product. If a match is found, return the product object.
   VII. Implement a method to display a list of all products in the inventory. Iterate over the ArrayList and print the details of each product, including the name, SKU, quantity, and price.

By implementing these methods, you will have a functional online product inventory management system using an ArrayList to store and manage the products. This solution allows you to add, remove, update, search, and display products in real time, providing efficient inventory management for the online store. **(10 Marks)**

Sample Input:

Add a new product to the inventory:
Product Name: "T-Shirt"
SKU: "TS001"
Quantity: 50
Price: Rs 1540

Remove a product from the inventory: SKU: "TS001"

Update the details of a product:
SKU: "TS002"
New Quantity: 75
New Price: Rs2499

Search for a product:
Search Query: "TS002"

Display all products in the inventory.

Sample Output:

Add a new product to the inventory:
Product 'T-Shirt' with SKU 'TS001' added successfully.

Remove a product from the inventory:
Product with SKU 'TS001' removed from the inventory.

Update the details of a product:
Product with SKU 'TS002' details updated successfully. New Quantity: 75, New Price: $24.99.

Search for a product:
Product found: Name: 'T-Shirt', SKU: 'TS002', Quantity: 75, Price: $24.99

Display all products in the inventory:
Product: T-Shirt, SKU: TS002, Quantity: 75, Price: $24.99
... (other products in the inventory**)**