1. **Employee Information System.**

You’ve been tasked with developing a Java program for managing employee information. The program should allow users to input details for a employee, including their name, age and salary. Ensure that the age is a non-negative integer and the grade is a non-negative double. Once the details are entered, the program should display the employee’s information.

Input Format:

The first line consists of a String that represents the name.

The second line consist of int which represents age.

The third line consists of a double that represents the salary.

Output Format:

The output should display the student’s deatials.

Sample Input 1:

Rakesh Sharma

45

25000.00

Sample Output 1:

Employee details:

Name: Rakesh Sharma

Age : 45

Salary: 25000.0

Sample Input 2:

Rakesh Sharma

-25

45000.0

Sample Output 2:

Age must be a non-negative integer.

Student details:

Name: Rakesh Sharma

Age: 0

Salary: 45000.00

Sample Input 3:

Rakesh Sharma

45

-1000

Sample Output 3:

Salary must be a non-negative double.

Student details:

Name: Rakesh Sharma

Age: 45

Salary: 0.00

Sample Input 4:

Rakesh Sharma

-45

-1000

Sample Output 4:

Age must be a non-negative integer.

Salary must be a non-negative double.

Student details:

Name: Rakesh Sharma

Age: 0

Salary: 0.00

1. **Book billing system using HashMap**

You are tasked with developing a simple Book bill system for a Supna Book store. The system should allow the user to input the book name and book price they want to purchase. After the data is collected, the system should ask for a total bill price threshold. If the total price of books purchased exceeds the threshold, the customer will receive a 25% discount on the total bill. The program will ensure that only positive price values are accepted and will display the details of the purchased books, the total bill before discount, and the total bill after applying the discount if applicable.

Input Format

The first line of input consists of an integer representing the number of books the customer wants to purchase.

The next two lines consists of the user inputs for each book:

. Book name (String)

. Book price (double, must be a positive number)

After entering the book details, the last line of input consists of the total price threshold.

Output Format:

The output displays the book names and prices of the items (2 decimal values) purchased.

The total bill (2 decimal values) before any discount.

If the total bill exceeds the threshold, the output displays the total bill(2 decimal values) after a 25% discount. If not then “No discount applied as the total bill does not exceed the threshold”.

If no books are purchased, the output prints a message stating that “No books were purchased.

Sample Input 1:

2

Java Complete Reference

100.00

C#

200.00

200.00

Sample Output 2:

Books purchased:

Book name: Java Complete Reference, Price: 100.00

Book name: C#, Price: 200.00

Total bill before discount: 300.00

No discount applied as the total bill does not exceed the threshold.

Sample Input 2:

2

Java Complete Reference

800.00

C#

200.00

200.00

Sample Output 2:

Books purchased:

Book name: Java Complete Reference, Price: 100.00

Book name: C#, Price: 200.00

Total bill before discount: 1000.00

Total bill after 25% discount: 750.00

1. **Electronic Store Management System using HashMap**

You are assigned to develop a Electronic Store Management System in Java using HashMap. The system should enable users to efficiently manage the Store collection of Electronic products.

Functionalities:

Add Product: Users should be able to add new Product to the system by providing the product name and quantity.

Search Product: Users should be able to search for a product by providing the product name. The system should display the corresponding quantity if the product exists; otherwise, it should notify the user that the product does not exist.

Display All Products: Users should have the option to view all products stored in the system, including their product name and quantities.

Total Number of Product: The system should display the total number of products stored in the system.

Clear All Products: Users should be able to clear all products from the library in a single operation.

Input format :

The first line consists of the size of the HashMap.

The next lines consist of values representing products in the following format: "Product name, Product Quantity".

The last input consists of a String that represents the product name to search.

Output format :

The output should contain search result, number of products and product details.

Sample Input 1 :

4

HP ZenBook, 8

Dell Mouse, 5

Dell Precision, 10

MacBook Air, 2

Dell Precision

Sample Output 1 :

Dell Precision Quantity: 10

Total number of products: 4

All Products:

MacBook Air, Quantity=2

Dell Mouse, Quantity=5

HP ZenBook, Quantity=8

Dell Precision, Quantity=10

All Products Cleared

Sample Input 2 :

4

HP ZenBook, 8

Dell Mouse, 5

Dell Precision, 10

MacBook Air, 2

MacBook Pro

Sample Output 2 :

MacBook does not exist

Total number of products: 4

All Products:

MacBook Air, Quantity=2

Dell Mouse, Quantity=5

HP ZenBook, Quantity=8

Dell Precision, Quantity=10

All Products Cleared

1. **Employee Salary Management using HashMap**

You are tasked with developing a simple employee salary management system for a company. The system should allow the HR manager to input the names and salaries of employees. After the data is collected, the system should ask for a salary threshold and then display the names of all employees whose salaries are above that threshold. The program will ensure that only positive salary values are accepted and will display a message if no employees meet the salary criteria.

Input Format

The first line of input consists of an integer representing the number of employees.

The next two lines consist of the user inputs for each employee:

- Employee name (String)

- Employee salary (double, must be a positive number)

After entering employee details, the last line of input consists of the salary threshold.

Output Format

The output displays the names and salaries of employees whose salaries are greater than the provided threshold.

Please note that only up to two decimal places are allowed while printing the salary.

If no employees meet the criteria, the output prints a message stating that "No employees have a salary greater than the threshold".

Sample Input 1

4

Ajay

23500.50

Kiran

30000.00

Sunil

20000.00

Ravi

40000.00

25000.00

Sample Output 1

Employee: Kiran, Salary: 30000.00

Employee: Ravi, Salary: 40000.00

Sample Input 2

4

Ajay

23500.50

Kiran

30000.00

Sunil

20000.00

Ravi

40000.00

55000.00

Sample Output 2

No employees have a salary greater than the threshold

1. **Problem Statement**

You are required to implement the following function:

public static void countOccurrences (int[] array);

The function accepts an array of non-negative integers as its argument. The function should count the occurrences of each number in the array and print the results.

Input Format

The first line should contain an integer representing the size of the array.

The second line should contain an array of integers.

Output Format

The output prints the result in the form of "number: count", where number is an element and count is number of occurrences of an element.

Constraints

The input array contains only non-negative integers.

Sample Input

9

1 2 3 4 5 1 2 3 4

Sample Output

1: 2

2: 2

3: 2

4: 2

5: 1

1. **Uniqueness**

You are tasked with developing a utility for processing user-provided text inputs in an application. One of the requirements is to ensure that any duplicate characters within a string input are removed to maintain clarity and uniqueness in the output. However, the order of appearance of the first occurrence of each character should be preserved to maintain the original context of the input.

Write a Java method that removes duplicate characters from a string while ensuring the order of the first appearance of each character is retained.

Input Format

A single string input String containing the user-provided text

Output Format

A single string with all duplicate characters removed, maintaining the order of the first appearance of each

character. If there is no duplicate then its should display "No duplicate characters found."

Input:

ABACDBA

Output

ABCD

1. **Frequency**

Imagine you are developing a text processing application where you need to analyze the frequency of each character in a user- provided text input. Your application aims to provide statistical insights into the textual content, such as character frequency distributions, without distinguishing between uppercase and lowercase letters using HashMap

You need to implement a Java method that counts the occurrences of each character in a given string, treating uppercase and lowercase versions of the same character as identical.

Input Format

It will take take user input as string

Output Format

Its should print each char and the count of the particular