WEEK 3 – LAB ASSIGNMENTS FOR JAVA LAB

1. Financial Data Analysis

As a data analyst at a financial firm, you've been given a dataset that contains the daily closing prices of a company's stock over the past year. Your task is to write a Java program to calculate the total sum and average closing price. The company's CEO wants to use this information to analyze the company's financial performance over the year. Can you accurately compute the required metrics to aid in their decision-making?

2. Web Browser Navigation Simulation

Your program should be able to push new URLs onto the stack as users navigate, pop URLs when users hit the "back" button, and handle edge cases like an empty stack when there are no more pages to go back to. You're tasked with simulating the "back" and "forward" navigation functionality of a web browser. To do this, you need to implement a stack using an array in Java. How will you ensure that your stack-based navigation is efficient and error-free?

3. Customer Service Queue Management

Imagine you're developing a customer service application for a large retail store. The system needs to manage customers waiting in line using a queue implemented with an array. As customers arrive, they are added to the end of the queue, and the service desk attends to them in a first-come, first-served manner. Can you write a Java program to simulate this queue management system and ensure that it can handle both busy and quiet periods effectively?

4. Image Processing with Matrices

You're working on a simple image processing task where two images are represented as 2D arrays of pixel values. To apply certain filters, you need to calculate the sum of these two matrices, representing the pixel-wise addition of two images. Your task is to write a Java program that accurately computes the resulting image after summing these two 2D arrays. Can you ensure that your solution handles different image sizes correctly?

5. Data Analysis in Marketing

As a data analyst in a marketing firm, you're given a 1D array containing customer satisfaction scores. Your goal is to find the range of these scores, which is defined as the difference between the maximum and minimum values in the array. This range will help the marketing team understand the variability in customer satisfaction. Can you develop a Java program that quickly identifies the range of values in this dataset?

6. Product Search in Inventory Management

You're part of the IT team at an e-commerce company, and you're tasked with implementing a search feature in the company's inventory management system. Given an array of product IDs, write a Java program that allows the user to search for a specific product ID. If the product exists in the array, your program should return its position; otherwise, indicate that the product is not found. How will you ensure that your search algorithm is both efficient and reliable?

7. Data Filtering in Sensor Networks

In a network of environmental sensors, you receive an array of integer readings representing temperatures recorded by the sensors. Your task is to write a Java program that calculates the sum of all even temperature readings. This information will be used to detect anomalies in the environment. How can you efficiently filter out the even numbers and calculate their sum to aid in environmental monitoring?

8. Matrix Diagonal Sum for Engineering Calculations

In an engineering application, you're given a 2D array representing a square matrix. Your task is to write a Java program to compute the sum of the diagonal elements of this matrix. This calculation is crucial for certain matrix-based operations used in structural analysis. Can

you ensure that your program handles both small and large matrices effectively while delivering accurate results?

9. In-Place Array Reversal for Data Encryption

As a security engineer, you're developing a basic encryption algorithm that requires reversing the elements of an integer array without using additional memory. Your task is to write a Java program that performs this reversal in place. How will you ensure that your program is both memory-efficient and able to handle large datasets?

10. Finding the Smallest Value in a Sensor Array

In a network of IoT devices, you receive an array of readings from different sensors. To identify potential issues, you need to find the smallest reading in this array. Write a Java program that takes n elements from the user and determines the smallest value. How will you ensure your solution is robust enough to handle noisy or inconsistent data from the sensors?

11. Sum of Odd Values in a Matrix for Image Processing

You're tasked with processing a grayscale image represented as a 2D array of pixel values. Your job is to find the sum of all odd-valued pixels in the image. This sum will be used to apply a special filter in image processing. Can you write a Java program that efficiently computes this sum while handling large images with varying pixel intensities?

12. Matrix Transpose for Data Transformation

You're working on a data transformation task where you need to transpose a matrix, represented as a 2D array. Transposing a matrix involves swapping its rows and columns. Your task is to write a Java program that performs this operation. How will you ensure that your program handles matrices of different sizes and shapes accurately?

13. Sparse Matrix Detection in Big Data Analysis

In a big data analysis project, you encounter a matrix where most of the elements are zero. Such a matrix is called a sparse matrix. Your task is to write a Java program that determines whether a given matrix is sparse. How will you ensure that your program accurately counts non-zero elements and classifies the matrix correctly?

14. Prime Number Count in Scientific Data

You're working on a scientific project where an array contains various integer data points. Your task is to count how many of these numbers are prime. Write a Java program to iterate through the array and count the prime numbers. How can you optimize your algorithm to handle large datasets efficiently?

15. Finding the Second Highest Score in a Competition

In a coding competition, you receive an array of scores from participants. Your task is to find the second highest score in this array. Write a Java program that accurately identifies this score, ensuring that your solution handles cases where there might be duplicates. How will you deal with edge cases such as all participants having the same score?

16. Non-Zero Element Count in Industrial Data

In an industrial control system, you receive an array of sensor readings, where zero might indicate a malfunction or missing data. Your task is to count the non-zero readings in the array. Write a Java program that accurately counts these non-zero elements, helping engineers quickly identify and filter out faulty readings. How will you ensure that your program is both efficient and accurate?

17. Merging Float Arrays in Scientific Simulations

In a scientific simulation, you receive two arrays of floating-point numbers representing different datasets. Your task is to merge these two arrays into one. Write a Java program that accomplishes this while preserving the order of elements. How can you ensure that your program handles floating-point precision and potential data loss during the merge?

18. Index-Wise Addition for Multispectral Image Processing

In multispectral imaging, you have two integer arrays representing different spectral bands of an image. You need to add these arrays index-wise and store the result in a third array. Write a Java program to perform this operation. How will you ensure that your program efficiently handles large arrays, typical in high-resolution imaging?

19. Matrix Multiplication for 3D Transformations

You're working on a 3D graphics engine that requires matrix multiplication for various transformations. Given two matrices, write a Java program to multiply them. This multiplication is essential for operations like rotation, scaling, and translation in 3D space. Can you ensure that your program handles large matrices and delivers accurate results required for smooth rendering?

20. Matrix Subtraction for Image Differencing

In an image differencing algorithm, you need to subtract one matrix from another, where both matrices represent grayscale images. Write a Java program to perform this subtraction, resulting in a difference matrix that highlights changes between the two images. How will you handle edge cases such as different matrix sizes or unexpected negative values in the resulting matrix?

21. Duplicate Element Detection in Big Data Streams

In a big data processing pipeline, you encounter a 1D array containing potentially millions of elements. Your task is to identify any duplicate elements and calculate their frequency of occurrence. Write a Java program that efficiently handles this detection and provides insights into the data's uniqueness. How will you optimize your program to process large arrays in real-time?

22. Alternate Number Extraction in Signal Processing

In a signal processing application, you receive an array of data points. Your task is to extract every alternate number from this array, which might represent periodic samples in a signal. Write a Java program to perform this extraction. How will you ensure that your program accurately extracts the required data, especially when dealing with noisy signals?

23. Merging Sorted Arrays in Database Systems

You're working on a database management system where you need to merge two sorted arrays representing indexes of records. The result should be a single sorted array that includes all elements from the original arrays. Write a Java program to accomplish this. How will you ensure that your merge operation is efficient, especially for large datasets?

24. The Constructor Chronicles: Building the Foundation of a Java Empire

In the bustling world of Java development, constructors are the architects who lay the foundation for every class object. You've been hired by a top-tier software company, **Constructor Dynamics**, renowned for its mastery over object creation. Your challenge is to design a robust Java class called ***Building***, which will serve as the blueprint for all future infrastructure in the company's high-tech city project, **JavaMetropolis**.

Your mission has three critical phases:

1. **Phase 1 - The Default Blueprint**: Create a Building class with a 0-arguments constructor, representing the basic blueprint that initiates all essential structural parameters to their default states. This constructor should print a message confirming that a standard building has been initialized.
2. **Phase 2 - Custom Designs**: Enhance the Building class by introducing a parameterized constructor that allows architects to specify the building's height, width, and depth. This constructor will enable the creation of custom-designed buildings in JavaMetropolis, reflecting the diverse needs of the city's futuristic skyline.

3. **Phase 3 - The Copying Architect**: Implement a copy constructor that can duplicate an existing Building object. This constructor is crucial for replicating successful designs in other areas of the city without altering the original structure.
4. **Bonus Phase - Overloaded Creations**: Demonstrate constructor overloading by adding multiple constructors to the Building class, allowing for various combinations of parameters (e.g., height and width only, height and depth only, etc.). Each overloaded constructor should initialize the building with the provided parameters while setting the rest to default values.

Your task is to merge these phases into a single, cohesive Java program that showcases the power and flexibility of constructors in building the future of JavaMetropolis. Good luck, Architect of Code! The future of JavaMetropolis rests in your hands.

25. **The Grading Enigma: Crafting the Perfect Evaluation System**

In the innovative city of **Evalon**, where every achievement and failure is meticulously recorded and analyzed, the **Performance Council** is tasked with ensuring that the evaluations of their workforce are fair, precise, and meaningful. As the city's top software architect, you have been summoned to design a critical component of their evaluation system: the **Grader**.

The Challenge:

The Performance Council needs a tool to translate raw performance scores into a clear and actionable letter grade. This will help them make informed decisions about employee promotions, raises, and training needs.

Your Mission:

1. **Create the `Grader` Class**:
    - **Attributes and Initialization**: Design a `Grader` class with an instance variable `score` that holds the performance score of an employee. Implement a constructor that ensures this score is properly initialized and falls within the acceptable range of 0 to 100. The constructor should handle edge cases and ensure no invalid scores are accepted.
    - **Determine the Grade**: Add a method called `letterGrade()` to the `Grader` class. This method will convert the numerical score into a letter grade based on Evalon's grading criteria:
        - **O** for Outstanding (90-100)
        - **E** for Excellent (80-89)
        - **A** for Average (70-79)
        - **B** for Below Average (60-69)
        - **C** for Critically Low (50-59)
        - **F** for Failing (0-49)
2. **Develop the Evaluation Scenario**:
    - Implement a `GraderDemo` class to put the `Grader` class to the test. This class should:
        - Prompt the user for a performance score.
        - Validate the input to ensure it's not negative and does not exceed 100. If the input is invalid, inform the user with an appropriate message.
        - Create a `Grader` object with the valid score.
        - Invoke the `letterGrade()` method to determine the employee's grade and display it in a user-friendly format.

The Outcome:
With your solution, the Performance Council of Evalon will have a reliable tool to convert numerical scores into clear, actionable grades, helping them maintain their high standards of evaluation and employee management. Your work will ensure that every employee's performance is assessed fairly and transparently, driving the city towards excellence and innovation. Good luck, Architect of Code! The success of Evalon's evaluation system rests in your hands.

26. The Commission Chronicles: Rewarding Excellence

In the bustling metropolis of **TradeCity**, where every sale is a step towards success, the **Sales Rewards Bureau** has embarked on a mission to enhance their incentive system. The Bureau's latest task is to build a sophisticated tool that will accurately calculate sales commissions and ensure that every sales professional is rewarded fairly for their efforts.

The Challenge:

You, the city's renowned software architect, are tasked with creating the **Commission** class, a pivotal component in this revamped reward system. Your goal is to design a system that calculates and verifies commissions based on sales figures while ensuring that only valid data is processed.

Your Mission:

1. **Design the `Commission` Class**:
    - **Attributes and Initialization**: Develop a `Commission` class with an instance variable `sales` to store the total sales amount. Implement a constructor that initializes this value, ensuring that it is correctly set up and ready for calculations.
    - **Calculate the Commission**: Add a method called `commission()` to the `Commission` class that calculates and returns the commission based on the following criteria:
        - 5% commission for sales up to $5,000
        - 10% commission for sales between $5,001 and $10,000
        - 15% commission for sales exceeding $10,000
2. **Create the Testing Scenario**:
    - Implement a `CommissionDemo` class to test the `Commission` class in action. This class should:
        - Prompt the user to input a sales amount.
        - Validate the input to ensure it is not negative. If the input is negative, display an "Invalid Input" message to the user.
        - Create a `Commission` object with the valid sales amount.
        - Call the `commission()` method to calculate the commission and print the result.

The Outcome:

With your expertly crafted solution, the Sales Rewards Bureau will be equipped with a powerful tool to accurately calculate commissions and manage sales incentives effectively. This will help drive motivation among sales professionals and ensure that their hard work is rewarded fairly. Get ready to transform the way TradeCity calculates and rewards sales performance. Your code will shape the future of sales incentives, ensuring fairness and transparency across the board.