Oranium Tech

Scenario 1

A teacher has recorded the scores of students from a recent test. To analyze how well the class performed, the teacher wants to perform some calculations on the scores using JavaScript.

The scores are stored in an array:

const scores = [78, 85, 95, 67, 88, 92, 55, 74, 99, 83, 69, 80, 91];

Using this array, help the teacher answer the following questions about the students' performance.

Questions

1. **Find All Passing Scores:**
   Use the filter method to find all scores that are 70 or above (passing marks). Create a new array called passingScores to store these values.
2. **Assign Grades Based on Scores:**
   Use the map method to convert each score to a grade:
   - "A" for scores 90 and above
   - "B" for scores 80–89
   - "C" for scores 70–79
   - "D" for scores below 70

     Create a new array called grades that stores the grades corresponding to each score in the scores array.
3. **Calculate the Average Score:**
   Use the reduce method to calculate the average score of the class.
4. **Count the Number of Failing Scores:**
   Find the number of scores that are below 70. Use the filter method to identify failing scores and then count them.
5. **Find the Highest Score:**
   Use the reduce method to find the highest score in the array.

Scenario 2: Movie Night Analysis

A movie streaming platform has recorded the ratings given by viewers for a selection of popular movies. Each rating is a number between 1 and 10, where 10 is the highest. The platform team wants to analyze these ratings to understand viewer preferences and make recommendations.

The list of ratings is stored in an array:

const movieRatings = [8, 5, 9, 6, 10, 4, 7, 9, 3, 8, 6, 10, 2];

Help the team answer the following questions:

Questions

1. **Identify Highly Rated Movies:**
   Find all movies with ratings of 7 or above and store these ratings in a new array called highRatings.
2. **Assign Rating Categories:**
   Classify each movie rating as:
   - "Excellent" for ratings 9 and above
   - "Good" for ratings between 7 and 8
   - "Average" for ratings between 4 and 6
   - "Poor" for ratings below 4

     Store these classifications in a new array called ratingCategories.
3. **Calculate the Average Rating:**
   Determine the average rating of all movies in the list. This will help the team understand the general viewer satisfaction.
4. **Count Low-Rated Movies:**
   Find out how many movies have a rating below 5, as these are considered poorly rated by viewers.
5. **Find the Highest Movie Rating:**
   Determine the highest rating in the list to identify the best-rated movie on the platform.

Scenario 3: Fitness Tracker Summary

A fitness app records the daily step count for a user over two weeks. The app wants to analyze the data to provide useful insights to the user.

Step Count Array:

const steps = [6500, 8000, 9500, 4000, 12000, 7000, 8500, 10200, 7500, 11500, 9000, 8200, 6000, 11000];

Questions:

1. **Find High Activity Days:**
   Identify all days where the user walked more than 10,000 steps. Create a new array called highActivityDays for these days.

2. **Convert Steps to Miles:**
   Assume that 2,000 steps are roughly equivalent to 1 mile. Create a new array called milesWalked that converts each day's step count into miles.

3. **Calculate Total Steps:**
   Find the total number of steps taken over these two weeks.

4. **Count Rest Days:**
   Count the days where the user walked fewer than 5,000 steps, as these are considered rest days.

5. **Find the Highest Step Count:**
   Determine the highest step count achieved in a single day.

Scenario 4: Diwali Gift Distribution

A group of friends is planning to exchange gifts for Diwali. Each friend has a list of gifts they are willing to buy, and the group wants to analyze the costs and preferences to make informed decisions.

Gift Price Array:

const giftPrices = [1500, 2000, 750, 1200, 3000, 1800, 1000, 450, 850, 600];

Questions:

1. **Find Affordable Gifts:** Identify the gifts that are priced at or below ₹1000. Create a new array called affordableGifts to store these values.

2. **Calculate Total Gift Cost:** Determine the total cost of all the gifts in the giftPrices array. Create a variable to store the total cost.

3. **Discounted Gifts:** Apply a 10% discount to each gift's price. Create a new array called discountedPrices that reflects the new prices after the discount.

4. **Count Gifts Above a Certain Price:** Find out how many gifts are priced above ₹1500. Store this count in a variable called expensiveGiftsCount.

5. **Average Gift Price:** Calculate the average price of the gifts. Create a variable to store the average price.

Scenario 5: Shopping Cart

A user has a shopping cart with various items they intend to purchase. The prices of these items are represented in an array. The user wants to analyze the cart to make better decisions about their purchases. Below are several tasks the user wants to accomplish with the prices in their shopping cart.

Price Array:

const prices = [200, 1500, 300, 800, 1200, 500, 700];

Tasks and Detailed Descriptions:

1. **Identify Affordable Items:** The user wants to find all items in the cart that cost ₹800 or less. This will help them identify the items they can afford without exceeding their budget. Create a new array that includes only these affordable items.
2. **Calculate the Total Cost of the Cart:** The user is interested in knowing the total amount they need to spend to buy all the items in the cart. This involves summing up all the prices in the array to provide a complete picture of the cart's total value.
3. **Determine the Average Price of Items:** To gain insight into their spending, the user would like to calculate the average price of the items in the cart. This requires taking the total price calculated in the previous step and dividing it by the total number of items in the cart.
4. **Count the Number of Expensive Items:** The user wants to keep track of how many items in their cart are considered expensive. Specifically, they want to count the items that are priced above ₹1000. This information can help them make decisions about whether to keep or remove these high-cost items.
5. **Find the Cheapest Item:** Lastly, the user is interested in finding out which item in the cart has the lowest price. Identifying the cheapest item can help them evaluate potential savings or better allocate their budget.