Travel Planner

Loops

1. **Calculate Total Trip Duration**

   - **Task:** Write a function that calculates the total duration of a trip by summing up the duration of all the activities in the given array.
   - **Inputs:** An array of activity objects.
   - **Example:**
     - Input:
       ```
       [{ name: "Activity 1", duration: 2 }, { name: "Activity 2", duration: 3 }, { name: "Activity 3", duration: 1 }]
       ```
     - Output: `6`
   - **Tip:** Use a loop to iterate over the activities and accumulate the durations.

2. **Find Longest Activity**

   - **Task:** Write a function that finds the longest activity in the given array and returns its name.
   - **Inputs:** An array of activity objects.
   - **Example:**
     - Input:
       ```
       [{ name: "Activity 1", duration: 2 }, { name: "Activity 2", duration: 3 }, { name: "Activity 3", duration: 1 }]
       ```
     - Output: `"Activity 2"`
   - **Tip:** Use a loop to iterate over the activities and keep track of the longest duration.

3. Find Most Common Activity**

   - **Task:** Write a function that finds the most common activity across all travel plans from an array of travel plan objects and returns its name.
   - **Inputs:** An array of travel plan objects. Each travel plan object has a `name` and an array of `activities`. Each `activity` is an object with a `name`.
   - **Example:**
     - Input:

```
[
    { name: "Plan 1", activities: [{ name: "Activity 1" }, { name:
"Activity 2" }] },
    { name: "Plan 2", activities: [{ name: "Activity 1" }, { name:
"Activity 3" }] }
]
```

    - Output: `"Activity 1"`
  - **Tip:** Use a loop to iterate over the travel plans. Inside that loop, use another loop to iterate over the `activities` array of each travel plan. Create a frequency counter object to count the occurrence of each activity. Return the activity with the highest occurrence.

4. Find Shortest Activity

- **Task:** Write a function that finds the shortest activity in the given array of travel plans and returns its name.
- **Inputs:** An array of travel plan objects. Each travel plan object has a `name` and an array of `activities`. Each `activity` is an object with a `name` and a `duration`.
- **Example:**
  - Input:

```
[
    { name: "Plan 1", activities: [{ name: "Activity 1", duration: 2
}, { name: "Activity 2", duration: 3 }] },
    { name: "Plan 2", activities: [{ name: "Activity 3", duration: 1
}, { name: "Activity 4", duration: 5 }] }
]
```

  - Output: `"Activity 3"`
- **Tip:** Use a loop to iterate over the travel plans. Inside that loop, use another loop to iterate over the `activities` array of each travel plan and keep track of the shortest duration.
-

5. **Find Shortest Segment**

   - **Task:** Write a function that finds the shortest segment in the given array and returns its start and end cities.
   - **Inputs:** An array of segment objects.

- **Example:**
  - Input:
    ```
    [{ start: "City A", end: "City B", distance: 100 }, { start: "City B", end: "City C", distance: 150 }, { start: "City C", end: "City D", distance: 75 }]
    ```
  - Output: `{ start: "City C", end: "City D" }`
- **Tip:** Use a loop to iterate over the segments and keep track of the shortest distance.

6. **Count Segments by Mode of Transportation**

- **Task:** Write a function that counts the number of segments for each mode of transportation in the given array and returns an object with the counts.
- **Inputs:** An array of segment objects.
- **Example:**
  - Input:
    ```
    [{ start: "City A", end: "City B", mode: "Car" }, { start: "City B", end: "City C", mode: "Train" }, { start: "City C", end: "City D", mode: "Car" }, { start: "City D", end: "City E", mode: "Flight" }]
    ```
  - Output: `{ Car: 2, Train: 1, Flight: 1 }`
- **Tip:** Use a loop to iterate over the segments and update the count for each mode of transportation.

7. **Find Average Expense**

- **Task:** Write a function that calculates the average expense of all the expenses in the given array.
- **Inputs:** An array of expense values.
- **Example:**
  - Input: `[50, 100, 75, 125, 200]`
  - Output: `110`
- **Tip:** Use a loop to iterate over the expenses and calculate the sum, then divide by the number of expenses.

8. **Find Highest Expense**

- **Task:** Write a function that finds the highest expense in the given array and returns it.
- **Inputs:** An array of expense values.
- **Example:**

    - Input: `[50, 100, 75, 125, 200]`
    - Output: `200`
  - **Tip:** Use a loop to iterate over the expenses and keep track of the highest value.

9. **Count Expenses within Budget**

  - **Task:** Write a function that counts the number of expenses within a specified budget in the given array.
  - **Inputs:** An array of expense values, and a budget value.
  - **Example:**
    - Input: `[50, 100, 75, 125, 200]`, budget: `150`
    - Output: `3`
  - **Tip:** Use a loop to iterate over the expenses and increment the count if the expense is within the budget.