**Technical Design Document Template**

**Name:** Joseph D Sullivan

**Date Created:** February 13, 2025

**Program Description**

Collect expenses from user, calculate expense totals, and display results.

**Functions**

1. **Name**  main

**Description** Entry function for when code is invoked directly. Retrieve expenses, calculate totals, and display results.

**Parameter(s)**

| Name | Type | Description |
| --- | --- | --- |
|  |  |  |

**Logic**

* 1. Retrieve expenses.
  2. If no expenses were entered, display a message and exit.
  3. Calculate lowest expense(s).
  4. Calculate highest expense(s).
  5. Calculate total expenses.
  6. Display results.

**Return(s)**

|  | Type | Description |
| --- | --- | --- |
|  |  |  |

1. **Name**  get\_expenses

**Description** Collects user expenses and returns a list of dictionaries with expense type and amount.

**Parameter(s)**

| Name | Type | Description |
| --- | --- | --- |
|  |  |  |

**Logic**

1. Initialize expenses.
2. Give user instructions.
3. Loop until user enters no type.
   1. Retrieve expense type.
   2. If no expense type was entered, exit loop.
   3. Retrieve expense amount.
   4. Store the valid expense as a dictionary in expenses.
4. Return expenses.

**Return(s)**

|  | Type | Description |
| --- | --- | --- |
|  | List[Dict[str, float]] | A list of dictionaries where each dictionary has:  - "type" (str): The expense type.  - "amount" (float): The expense amount. |

1. **Name**  get\_expense\_type

**Description** Prompts the user to enter an expense type and returns the cleaned input.

**Parameter(s)**

| Name | Type | Description |
| --- | --- | --- |
|  |  |  |

**Logic**

1. Get expense type from user input.
2. Trim whitespace from expense type.
3. Return cleaned expense type.

**Return(s)**

|  | Type | Description |
| --- | --- | --- |
|  | str | The trimmed expense type entered by user. |

1. **Name**  get\_expense\_amount

**Description** Prompts the user to enter a valid expense amount and returns it as a float.

**Parameter(s)**

| Name | Type | Description |
| --- | --- | --- |
|  |  |  |

**Logic**

1. Create prompt text.
2. Loop until user enters valid amount.
   1. Retrieve expense amount from user input.
   2. Trim whitespace from expense amount.
   3. If no expense amount was entered, restart loop.
   4. Validate expense amount as a positive number. Restart loop if validation fails.
   5. Expense amount is valid. Return it.

**Return(s)**

|  | Type | Description |
| --- | --- | --- |
|  | float | The validated positive expense amount entered by the user. |

1. **Name**  get\_lowest

**Description** Finds the expense(s) with the lowest amount.

**Parameter(s)**

| Name | Type | Description |
| --- | --- | --- |
| expenses | List[Dict[str, float]] | A list of dictionaries where each dictionary has:  - "type" (str): The expense type.  - "amount" (float): The expense amount. |

**Logic**

1. If expenses is empty, return an empty list.
2. Find the minimum expense amount.
3. Get all expenses that match the minimum amount.
4. Return lowest expenses.

**Return(s)**

|  | Type | Description |
| --- | --- | --- |
|  | List[Dict[str, float]] | A list containing all expense(s) with the lowest amount. Returns an empty list if expenses is empty. |

1. **Name**  get\_highest

**Description** Finds the expense(s) with the highest amount.

**Parameter(s)**

| Name | Type | Description |
| --- | --- | --- |
| expenses | List[Dict[str, float]] | A list of dictionaries where each dictionary has:  - "type" (str): The expense type.  - "amount" (float): The expense amount. |

**Logic**

1. If expenses is empty, return an empty list.
2. Determine the maximum expense amount.
3. Get all expenses that match the maximum amount.
4. Return highest expenses.

**Return(s)**

|  | Type | Description |
| --- | --- | --- |
|  | List[Dict[str, float]] | A list containing all expense(s) with the highest amount. Returns an empty list if expenses is empty. |

1. **Name**  get\_total

**Description** Calculates the sum total of all expense amounts.

**Parameter(s)**

| Name | Type | Description |
| --- | --- | --- |
| expenses | List[Dict[str, float]] | A list of dictionaries where each dictionary has:  - "type" (str): The expense type.  - "amount" (float): The expense amount. |

**Logic**

1. Sum up the expense amounts.
2. Return sum total.

**Return(s)**

|  | Type | Description |
| --- | --- | --- |
|  | float | The sum of all expense amounts. Returns 0.0 if the list is empty. |

**Logic**

1. main()
2. get\_expenses()
3. get\_expense\_type()
4. get\_expense\_amount()
5. get\_lowest()
6. get\_highest()
7. get\_total()

**Link**

<https://github.com/JosephDSullivan/COP2373/blob/main/src/chapter03/jsulli40_chapter03_assignment01.py>