**Technical Design Document Template**

**Name:** Ethan Back

**Date Created:** 9/28/2025

**Program Description:**

We ask the user for the type and value of each monthly expense they have. We then save these as dictions with two unique keys, within a list. We then take these inputs, finding the highest and lowest values amongst them, whilst finding the total of all the expenses.

**Functions used in the Program (list in order as they are called):**

1. **Function Name:** expense\_lister

**Description:** collects user input to create a list of dictionaries, each containing a monthly expense and its corresponding type. Each dictionary uses keys to store the expense type and value, allowing us to more easily use the paired data later, for formulas and comparisons.

**Parameters:** None

**Variables:**

**expenses** – a list that stores each expense as a dictionary with the keys ‘type’ (str) and ‘amount’ (float)

**amount-** A float representing the amount associated to the expense type, obtained from user input

**expense\_type-** A string storing the users input for the expense name/type

**Logical Steps:**

1. Initialize an empty list expenses to store expense dictionaries.
2. Enter an infinite loop to repeatedly prompt the user for input.
3. Ask the user to input the expense type. If the user presses Enter without typing, exit the loop.
4. Prompt the user to enter the amount for the given expense type. Attempt to convert this input into a float.
5. If the conversion fails (invalid input), display an error message and prompt again.
6. If successful, create a dictionary with 'type' and 'amount' keys and append it to the expenses list.
7. Repeat steps 3–6 until the user exits by entering a blank expense type.
8. Return the list of expense dictionaries.

**Returns:** a list of dictionaries \*expenses= [ ]\*, where each dictionary represents an expense with keys ‘type’ (strg) and ‘amount’ (float)

2. **Function Name:** main()

**Description:** Calls upon our expense\_lister() function to solve the following; total expenses, highest expense, lowest expense. Then displays and labels these amounts, as well as stating the type of the highest and lowest expenses.

**Parameters:** None

**Variables:**

* Expenses – A list of dictionaries returned by expense\_lister(), where each dictionary contains ‘type’ and ‘amount’ keys, representing an expense
* Total – a float representing the sum of all expense amounts, calculated using reduce
* Highest- A dictionary representing the expense with the highest ‘amount’ key
* Lowest- A dictionary representing the expense with the lowest ‘amount’ key

**Logical Steps:**

1. Call the expense\_lister() function to gather a list of expense dictionaries from user input.
2. Check if the returned expenses list is empty; if so, print a message and terminate the program.
3. Use reduce() with a lambda function to sum all the 'amount' values in the expenses list and store the result in total.
4. Use reduce() with a lambda function to find the expense dictionary with the highest 'amount'.
5. Use reduce() with a lambda function to find the expense dictionary with the lowest 'amount'.
6. Print the total expenses formatted to two decimal places.
7. Print the type and amount of the highest expense.
8. Print the type and amount of the lowest expense.

**Returns:** None

**Logical Steps:**

1. expense\_lister()
2. main()

**Link to your repository:** <https://github.com/if-it-Works-it-Works/COP2373>

**Output Screenshot: (make sure big enough so I can see)**

