MTH 4300: Algorithms, Computers and Programming II

Spring 2026

Section: SMWA

Problem Set 2

I know that it's very easy to just use AI to solve these problems, but please resist! I expect you all to give these problems an honest attempt. Spend about 30 minutes trying each problem yourself first. If you're still not getting to a good place with your solution, then prompt AI to guide you to a solution rather than expect it to solve the entire problem for you.

Problem 1: Number Analyzer

Write a program that reads integers from the user until they enter `-1` (sentinel value). As numbers are read, the program tracks statistics **without** storing all the numbers. Implement the following functions:

- `void update_stats(int value, int& count, int& sum, int& min_val, int& max_val)` – updates all running statistics via references. It should increment `count`, add `value` to `sum`, and update `min_val` and `max_val` if necessary.
- `char classify(int value)` – returns `'P'` for positive, `'N'` for negative, or `'Z'` for zero using if-else statements.

Your `main` function should:

1. Use a `while` loop to read integers until the user enters `-1`
2. For each number, call `update_stats` and `classify`
3. Keep a count of how many positive, negative, and zero values were entered
4. After the loop ends, print the count, sum, average, min, max, and the positive/negative/zero counts
5. If no numbers were entered (the user immediately enters `-1`), print `"No numbers entered."` and exit

Sample output:

```
Enter a number (-1 to stop): 10
Enter a number (-1 to stop): -5
Enter a number (-1 to stop): 0
Enter a number (-1 to stop): 8
Enter a number (-1 to stop): -3
Enter a number (-1 to stop): -1

Count: 5
Sum: 10
Average: 2
Min: -5, Max: 10
Positive: 2, Negative: 2, Zero: 1
```

Problem 2: Interactive Calculator with Input Validation

Write a program that implements a simple calculator using a menu loop:

1. Display a menu with options: `1. Add`, `2. Subtract`, `3. Multiply`, `4. Divide`, `5. Quit`
2. Use a `do-while` loop to keep showing the menu until the user picks Quit
3. Use a `switch` statement to dispatch the selected operation
4. For each operation, write a separate function (e.g., `double add(double a, double b)`)
5. For division, validate that the denominator is not zero – if it is, print an error and use `continue` to skip back to the menu
6. After each valid operation, print the result

Sample output:

```
1. Add  2. Subtract  3. Multiply  4. Divide  5. Quit
Choice: 1
Enter two numbers: 10 3
Result: 13

1. Add  2. Subtract  3. Multiply  4. Divide  5. Quit
Choice: 4
Enter two numbers: 7 0
Error: division by zero!

1. Add  2. Subtract  3. Multiply  4. Divide  5. Quit
Choice: 5
Goodbye!
```