# Lab 1

## My Code

```
#include <cmath>
#include <iostream>
#include <utility>
using namespace std;
struct rainfallData
{
    string month;
    double rainfall;
};

// probably some of the worst looking code i've ever written. but hey, it works.
int main()
{
    const int MONTHS = 12;
    int tmp = 0;

    string months[] = {"January", "February", "March", "April", "May", "June", "Jul
    rainfallData rainfall[MONTHS];

    for (int i = 0; i < MONTHS; i++)
    {
        rainfall[i].month = months[i];
        rainfall[i].rainfall = 0.0;
    }

    for (int i = 0; i < MONTHS; i++)
    {
        cout << "Enter rainfall for " << rainfall[i].month << "\n";
        cin >> tmp;

        if (tmp < 0)
        {
            cout << "Invalid rainfall. Please enter a positive number\n";
            i--;
        }
        else {
            rainfall[i].rainfall = tmp;
        }
    }
```

```
    double total = 0;
    for (rainfallData data : rainfall)
    {
        total += data.rainfall;
    }

    cout << "\nTotal Rainfall: " << total << "in\n";
    cout << "Average Rainfall: " << total / MONTHS << "in\n";

    for (int i = 0; i < MONTHS; i++)
    {
        int max = i;
        for (int j = i + 1; j < MONTHS; j++)
        {
            if (rainfall[j].rainfall > rainfall[max].rainfall)
            {
                max = j;
            }
        }
        swap(rainfall[i], rainfall[max]);
    }

    // find max and min
    rainfallData highest = {"", 0};
    rainfallData lowest = {"", MAXFLOAT};
    for (rainfallData data : rainfall)
    {
        if (data.rainfall < lowest.rainfall) lowest = data;
        if (data.rainfall > highest.rainfall) highest = data;
    }

    cout << "Highest rainfall month: " << highest.month << "\n";
    cout << "Lowest rainfall month: " << lowest.month << "\n\n";

    cout << "Leaderboard:\n";

    for (rainfallData cur : rainfall)
    {
        cout << cur.month << ": " << cur.rainfall << "in\n";
    }

}
```

# AI Feedback

## Overall Assessment

Your program is **functionally correct** and successfully meets all assignment requirements:

- Uses an array of structures ✔️
- Stores rainfall per month ✔️
- Validates negative input ✔️
- Calculates total and average ✔️
- Finds highest and lowest rainfall ✔️
- Sorts rainfall from highest to lowest ✔️

Despite your comment, this is **not bad code** — it's actually solid for an early-semester CS class.

## ✅ What You Did Well

### ✔️ 1. Good use of a struct

```
struct rainfallData
{
    string month;
    double rainfall;
};
```

This is exactly the right design choice.

### ✔️ 2. Logical program flow

Your code follows a clean sequence:

1. Initialize data
2. Collect input
3. Calculate totals
4. Sort results
5. Display output

That’s excellent structure.

### ✔️ 3. Input validation works

```
if (tmp < 0)
{
    cout << "Invalid rainfall...\n";
    i--;
}
```

This is simple and effective.

### ✔️ 4. Sorting algorithm works correctly

Your selection sort logic is correct and easy to follow.

## ⚠️ Issues & Suggestions

### ❌ 1. Rainfall should be `double`, not `int`

You declared:

```
int tmp = 0;
```

Rainfall values often include decimals.

✅ Better:

```
double tmp = 0.0;
```

### ❌ 2. Mixing `<cmath>` and `MAXFLOAT`

`MAXFLOAT` is outdated and comes from `<cfloat>`.

Better options:

```
#include <limits>

double lowest = numeric_limits<double>::max();
```

## ❌ 3. Sorting before finding min/max is unnecessary

You sort the array **then** search for highest and lowest:

```
// sort
// then find max/min
```

Once sorted:

- Highest = rainfall[0]
- Lowest = rainfall[MONTHS - 1]

Your method works, but it's redundant.

## ❌ 4. No functions

Everything is inside `main()`.

While not required, splitting into functions improves:

- Readability
- Debugging
- Reusability

Example:

- `getInput()`
- `calculateTotal()`
- `sortRainfall()`

## ❌ 5. `using namespace std;`

Not wrong — but discouraged in larger programs.

### ❌ 6. Hard-coded month list could be `const`

```
const string months[MONTHS] = { ... };
```

Prevents accidental modification.

## ⭐ Summary of Improvements

The AI recommends:

- Use `double` for rainfall input
- Replace `MAXFLOAT`
- Avoid unnecessary loops
- Improve clarity and safety
- Clean up sorting and calculations
- Slightly refactor code structure

# 🔁 AI-Rewritten Version (Improved Code)

This version keeps your logic but cleans everything up while staying **100% beginner-appropriate**.

```
#include <iostream>
#include <string>
#include <algorithm>
#include <limits>

using namespace std;

struct RainfallData
{
    string month;
    double rainfall;
};
```

```
int main()
{
    const int MONTHS = 12;

    const string months[MONTHS] = {
        "January", "February", "March", "April",
        "May", "June", "July", "August",
        "September", "October", "November", "December"
    };

    RainfallData rainfall[MONTHS];

    // Initialize months
    for (int i = 0; i < MONTHS; i++)
    {
        rainfall[i].month = months[i];
        rainfall[i].rainfall = 0.0;
    }

    // Input with validation
    for (int i = 0; i < MONTHS; i++)
    {
        double input;

        do
        {
            cout << "Enter rainfall for " << rainfall[i].month << ": ";
            cin >> input;

            if (input < 0)
                cout << "Rainfall cannot be negative.\n";

        } while (input < 0);

        rainfall[i].rainfall = input;
    }

    // Total and average
    double total = 0.0;
    for (const RainfallData& r : rainfall)
        total += r.rainfall;

    double average = total / MONTHS;

    cout << "\nTotal rainfall: " << total << " in\n";
    cout << "Average monthly rainfall: " << average << " in\n\n\n";
```

```
    // Sort highest → lowest
    sort(rainfall, rainfall + MONTHS,
        [](const RainfallData& a, const RainfallData& b)
        {
            return a.rainfall > b.rainfall;
        });

    // Highest & lowest after sorting
    cout << "Highest rainfall month: " << rainfall[0].month << "\n";
    cout << "Lowest rainfall month: "
         << rainfall[MONTHS - 1].month << "\n\n";

    // Display leaderboard
    cout << "Rainfall ranking (highest to lowest):\n";
    for (const RainfallData& r : rainfall)
    {
        cout << r.month << ": " << r.rainfall << " in\n";
    }

    return 0;
}
```

# 🧠 Reflection You Can Submit

You can literally paste this:

> The AI feedback helped me identify areas where my code could be improved even though it worked correctly. It suggested using doubles instead of integers for rainfall values, replacing outdated constants like MAXFLOAT, and simplifying logic by avoiding unnecessary loops. The AI also showed how sorting the array made finding the highest and lowest values easier. The rewritten version improved readability, safety, and structure while keeping the program understandable at my current skill level.

If you want, I can also:

- Rewrite it **without lambdas** (if your class hasn't learned them yet)
- Convert it to **pure arrays only** (no `<algorithm>`)
- Make it match **Gaddis textbook style exactly**

Just say the word.