Names :

Belal Eslam 192200222

Habiba Mahmoud 192200356

Salma Emad 192200365

Jana Ibrahim 192200292

Test Scenarios for calculateBill Function

ID	Scenario Name	Description	Test Steps	Preconditions	Expected Result	Actual Result	Status
TC001	Valid Prices Input	Test with all positive prices	Call calculateBill({10.0, 20.0, 30.0}, 3)	Array has 3 valid positive prices	Returns correct total with 8% tax: (10 + 20 + 30) × 1.08 = **64.80**	Returns **64.80**	✅ Pass
TC002	Contains Negative Price	Test behavior with a negative price	Call calculateBill({15.99, 24.50, -3.25}, 3)	One value is negative	Should reject or ignore negative value, or report error	Includes -3.25 → **wrong total** is calculated	❌ Fail
TC003	Loop Off-by-One Error	Test for invalid array access due to loop condition	Call calculateBill({10.0, 20.0, 30.0}, 3) with i <= count	Loop incorrectly runs to i = count	Risk of accessing outside array bounds (undefined behavior)	Accesses prices[3], which is **out of bounds**	❌ Fail

ID	Scenario Name	Description	Test Steps	Preconditions	Expected Result	Actual Result	Status
TC004	Empty Array	Test function behavior with empty array	Call calculateBill({}, 0)	Array is empty, count = 0	Should return 0.0 safely	Loop runs once → accesses prices[0], causing **undefined behavior**	❌ Fail
TC005	Large Price Values	Test with very high price values	Call calculateBill({100000, 200000, 300000}, 3)	Valid large values	Returns correct total: (600000 × 1.08) = **648,000**	Returns **648000**	✅ Pass
TC006	No Rounding Applied	Check if the total is rounded to 2 decimal places	Call calculateBill({14.375}, 1)	Tax creates long decimal result	Should round result to 2 decimal places: **15.53**	Returns **15.525** (not rounded)	❌ Fail
TC007	Bad Output Formatting	Check console output format	Run program with any valid prices	Output is shown using std::cout	Should display with 2 decimal places, e.g., 43.73	Shows raw float (e.g., 43.7292)	❌ Fail
TC008	Invalid Count Parameter	Pass a count value larger than array size	Call calculateBill({10.0, 15.0}, 3)	Count is 3 but array has only 2 elements	Should throw error or prevent access beyond array	Accesses prices[2], which is out-of-bounds → **undefined**	❌ Fail

ID	Scenario Name	Description	Test Steps	Preconditions	Expected Result	Actual Result	Status
						behavior	

✅ **Statement Coverage Analysis:**

Line	Executed?	Why
const double taxRate = 0.08;	✅ Yes	Always runs
double sub = 0.0;	✅ Yes	Always runs
for (int i = 0; i <= count; i++)	✅ Yes	Runs 4 times (0 to 3 inclusive), even though this is wrong
sub += prices[i];	✅ Yes	Runs 4 times, last one accesses out-of-bounds memory
double total = sub * (1 + taxRate);	✅ Yes	Runs once
return total;	✅ Yes	Runs once

✅ **Statement Coverage = 100%.**

Decision Coverage Analysis:

There is **one main decision** in this function:

for (int i = 0; i <= count; i++) // i <= count

Loop condition i <= count:

- **Evaluated True:** ✅ Yes — multiple times (i=0 to i=3)
- **Evaluated False:** ❌ No — never false, we never exit from i == count (3), so i == 4 condition isn't tested

So, decision is only partially covered.

❌ **Decision Coverage = 50%**

Issued Code :

```
#include <iostream>
#include <cmath>

double calculateBill(double prices[], int count) {  // Issue: No size validation
   const double taxRate = 0.08;  // Issue: Hardcoded tax rate
   double sub = 0.0;            // Issue: Poor variable name

   for (int i = 0; i <= count; i++) {  // Issue: Off-by-one error
      sub += prices[i];              // Issue: No negative price check
   }

   double total = sub * (1 + taxRate);
   return total;  // Issue: No rounding
}

int main() {
   double itemPrices[3] = { 15.99, 24.50, -3.25   };  // Issue: Negative price

   // Issue: No validation of array size
   double finalTotal = calculateBill(itemPrices, 3);

   // Issue: Poor output formatting`
```

```
    std::cout << "Your total is: " << finalTotal << std::endl;

    return 0;
```

Fixed Code:

```
#include <iostream>
#include <iomanip>  // for std::setprecision
#include <cmath>    // for std::round

double calculateBill(double prices[], int count, double taxRate = 0.08) {
    if (count <= 0) {
        std::cerr << "Error: No items to calculate.\n";
        return 0.0;
    }

    double subtotal = 0.0;

    for (int i = 0; i < count; i++) {
        if (prices[i] < 0) {
            std::cerr << "Warning: Skipping negative price: " << prices[i] << "\n";
            continue;  // skip negative prices
        }
        subtotal += prices[i];
    }

    double total = subtotal * (1 + taxRate);

    // Round to 2 decimal places
```

```
    total = std::round(total * 100) / 100;

    return total;
}

int main() {
    int count;
    std::cout << "Enter number of items: ";
    std::cin >> count;

    if (count <= 0) {
        std::cerr << "Invalid number of items.\n";
        return 1;
    }

    double* itemPrices = new double[count];  // dynamic array

    for (int i = 0; i < count; i++) {
        std::cout << "Enter price for item " << i + 1 << ": ";
        std::cin >> itemPrices[i];
    }

    double finalTotal = calculateBill(itemPrices, count);

    std::cout << std::fixed << std::setprecision(2);
    std::cout << "Your total is: $" << finalTotal << std::endl;
```

```
delete[] itemPrices;  // cleanup

return 0;
```