

# CSCI 140 PA 5 Submission

Due Date: 3/30/23

Name: Ali Mortada

Exercise 1 – **zyBook 6.26 LAB: Driving cost - functions** -- check if completely done in zyBook \_\_X\_\_ ; otherwise, discuss issues below  
Include a screenshot of current status/score



LAB ACTIVITY 6.26.1: LAB: Driving cost - functions 10 / 10 ✓

main.cpp Load default template...

```
1 #include <iostream>
2 #include <iomanip>           // For setprecision
3 using namespace std;
4
5 double DrivingCost(double milesPerGallon, double dollarsPerGallon, double milesDriven) {
6     return (dollarsPerGallon / milesPerGallon) * milesDriven;
7 }
8
9 int main() {
10     double milesPerGallon, dollarsPerGallon;
11     cin >> milesPerGallon >> dollarsPerGallon;
12
13     cout << fixed << setprecision(2);
14     cout << DrivingCost(milesPerGallon, dollarsPerGallon, 10) << " " << DrivingCost(milesPerGallon, dollarsPerGallon
15
16     return 0;
17 }
```

Exercise 2 – **zyBook 6.29 LAB: Flip a coin** -- check if completely done in zyBook \_\_X\_\_ ; otherwise, discuss issues below  
Include a screenshot of current status/score

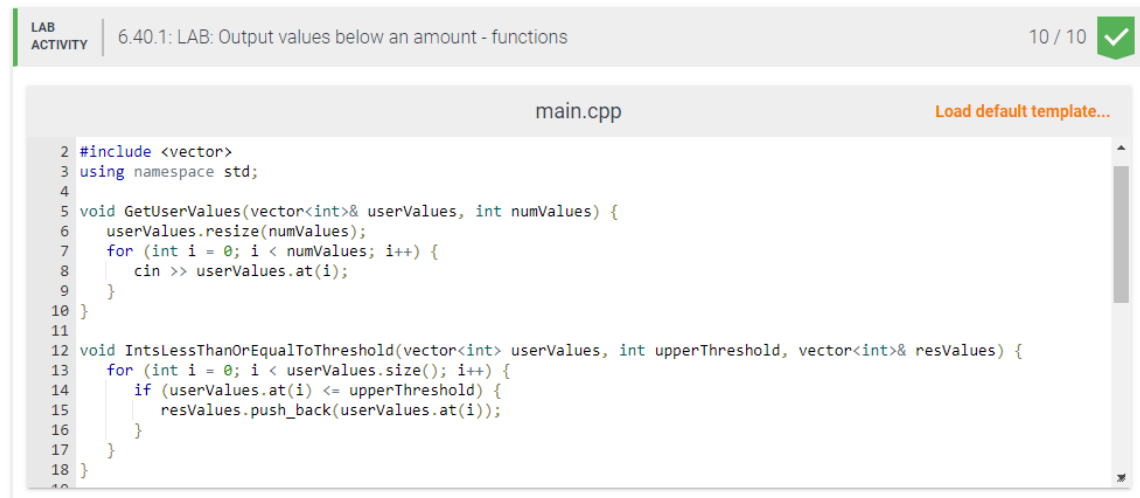


LAB ACTIVITY 6.29.1: LAB: Flip a coin 10 / 10 ✓

main.cpp Load default template...

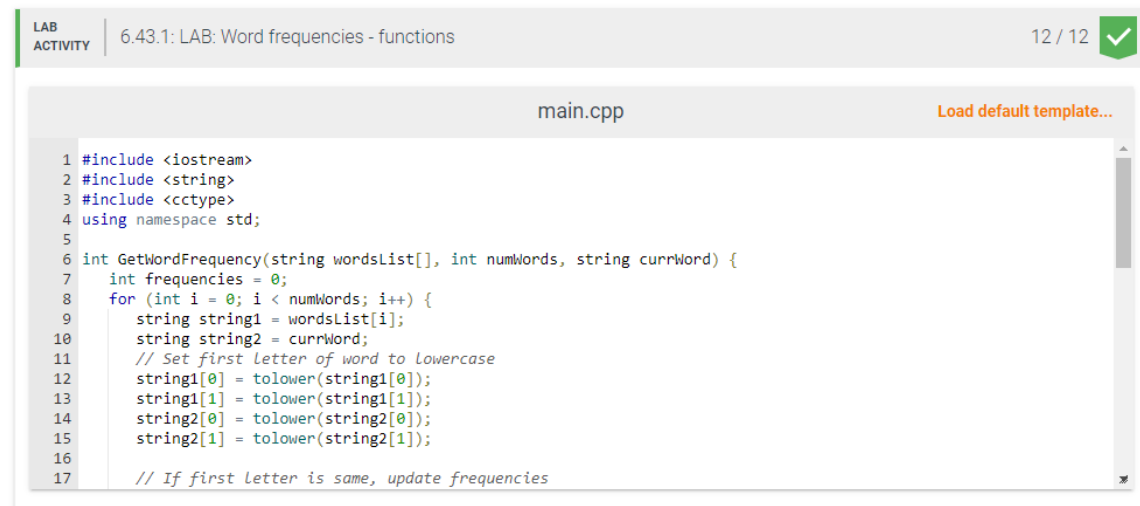
```
1 #include <iostream>
2 #include <cstdlib>
3 using namespace std;
4
5 string CoinFlip() {
6     int randValue = rand() % 2;
7
8     if (randValue == 1) {
9         return "Heads";
10    } else {
11        return "Tails";
12    }
13 }
14
15 int main() {
16     int numFlips;
17 }
```

Exercise 3 – **zyBook 6.40 LAB: Output values below an amount - functions** -- check if completely done in zyBook \_\_X\_\_ ; otherwise, discuss issues below  
Include a screenshot of current status/score



```
2 #include <vector>
3 using namespace std;
4
5 void GetUserValues(vector<int>& userValues, int numValues) {
6     userValues.resize(numValues);
7     for (int i = 0; i < numValues; i++) {
8         cin >> userValues.at(i);
9     }
10 }
11
12 void IntsLessThanOrEqualToThreshold(vector<int> userValues, int upperThreshold, vector<int>& resValues) {
13     for (int i = 0; i < userValues.size(); i++) {
14         if (userValues.at(i) <= upperThreshold) {
15             resValues.push_back(userValues.at(i));
16         }
17     }
18 }
```

Exercise 4 – **zyBook 6.43 LAB: Word frequencies - functions** -- check if completely done in zyBook \_\_X\_\_ ; otherwise, discuss issues below  
Include a screenshot of current status/score



```
1 #include <iostream>
2 #include <string>
3 #include <cctype>
4 using namespace std;
5
6 int GetWordFrequency(string wordsList[], int numWords, string currWord) {
7     int frequencies = 0;
8     for (int i = 0; i < numWords; i++) {
9         string string1 = wordsList[i];
10        string string2 = currWord;
11        // Set first letter of word to lowercase
12        string1[0] = tolower(string1[0]);
13        string1[1] = tolower(string1[1]);
14        string2[0] = tolower(string2[0]);
15        string2[1] = tolower(string2[1]);
16
17        // If first letter is same, update frequencies
```

Exercise 5 -- **Large Integers Version 2** -- check if completely done \_\_X\_\_ ; otherwise, discuss issues below

Source code below:

```
/*
    Program: Large Integers V2
    Author: Ali Mortada
    Class: CSCI 140
    Date: 3/30/23
    Description: Takes two large integers as strings, converts them into
    vectors, adds them up,
    then outputs the result. Uses three functions: one to store the
    integers as vectors, one to add
    them up, and one to output the result. Terminates if one input has
    more than 25 digits, or if
    the total digits of both integers is greater than 25.
    I certify that the code below is my own work.
    Exception(s): N/A
*/

#include <iostream>
#include <vector>
#include <string>
using namespace std;

// Function prototypes
void stringToVector(string, vector<int>&);
bool addOperands(vector<int>&, vector<int>&, vector<int>&, int, int);
void outputLargeInteger(string, string, vector<int>&, bool);

const int MAX_DIGITS = 25;
int main() {
    vector<int> num1(MAX_DIGITS);
    vector<int> num2(MAX_DIGITS);
    vector<int> result(MAX_DIGITS);
    string input1, input2;
    char op;

    cout << "Author: Ali Mortada" << endl;

    // Prompt user for input
    cout << "Enter an expression --> ";
    cin >> input1 >> op >> input2;
```

```

    // If either input1 or input2 has more than 25 digits, terminate
program
    if (input1.size() > MAX_DIGITS || input2.size() > MAX_DIGITS) {
        cout << "Invalid operand(s)" << endl;
        return 0;
    }

    // Convert input strings to vectors
    stringToVector(input1, num1);
    stringToVector(input2, num2);

    // Add operands, then output result
    outputLargeInteger(input1, input2, result, addOperands(num1, num2,
result, input1.size(), input2.size()));

    return 0;
}

// Function definitions
// Converts input string into a vector
void stringToVector(string input, vector<int>& num) {
    int numDigits = input.size();
    int j = 0;
    for (int i = numDigits - 1; i >= 0; i--, j++) {
        num.at(i) = input.at(j) - '0';
    }
}

// Add vector 1 and 2 together
bool addOperands(vector<int>& num1, vector<int>& num2, vector<int>&
result, int numDigits1, int numDigits2) {
    int carry;
    for (int i = 0; i < MAX_DIGITS; i++) {
        // If digits add up to less than 10, simply add. Otherwise, carry
the 1.
        if (result.at(i) + num1.at(i) + num2.at(i) < 10) {
            result.at(i) += num1.at(i) + num2.at(i);
            carry = 0;
        } else {
            result.at(i) = (result.at(i) + num1.at(i) + num2.at(i)) % 10;
            // If next index is valid, carry the 1
            if (i + 1 < MAX_DIGITS)
                result.at(i + 1) = 1;
            carry = 1;
        }
    }
}

```



```
Author: Ali Mortada  
Enter an expression --> 999999999999999999999999 + 1  
999999999999999999999999 + 1 = overflow
```

```
Author: Ali Mortada  
Enter an expression --> 999999999999999999012345 + 123  
Invalid operand(s)
```

### Answer for Question 1

One good reason for creating and using functions is to modularize your code. This makes it so that if the function is used multiple times and there is a problem in one instance, it is easier to fix the problem for all instances as well. Another reason is that it reduces clutter in your main function, making it easier to follow along with the code. A third reason is that it helps with creating a program incrementally, which is very helpful when creating large programs.

### Answer for Question 2

It is not a good idea to output a calculated result inside a calculated function that returns a value because it could cause some confusion. A better idea would be to output the calculated result after the function has been called.

Extra Credit – provide if applicable.