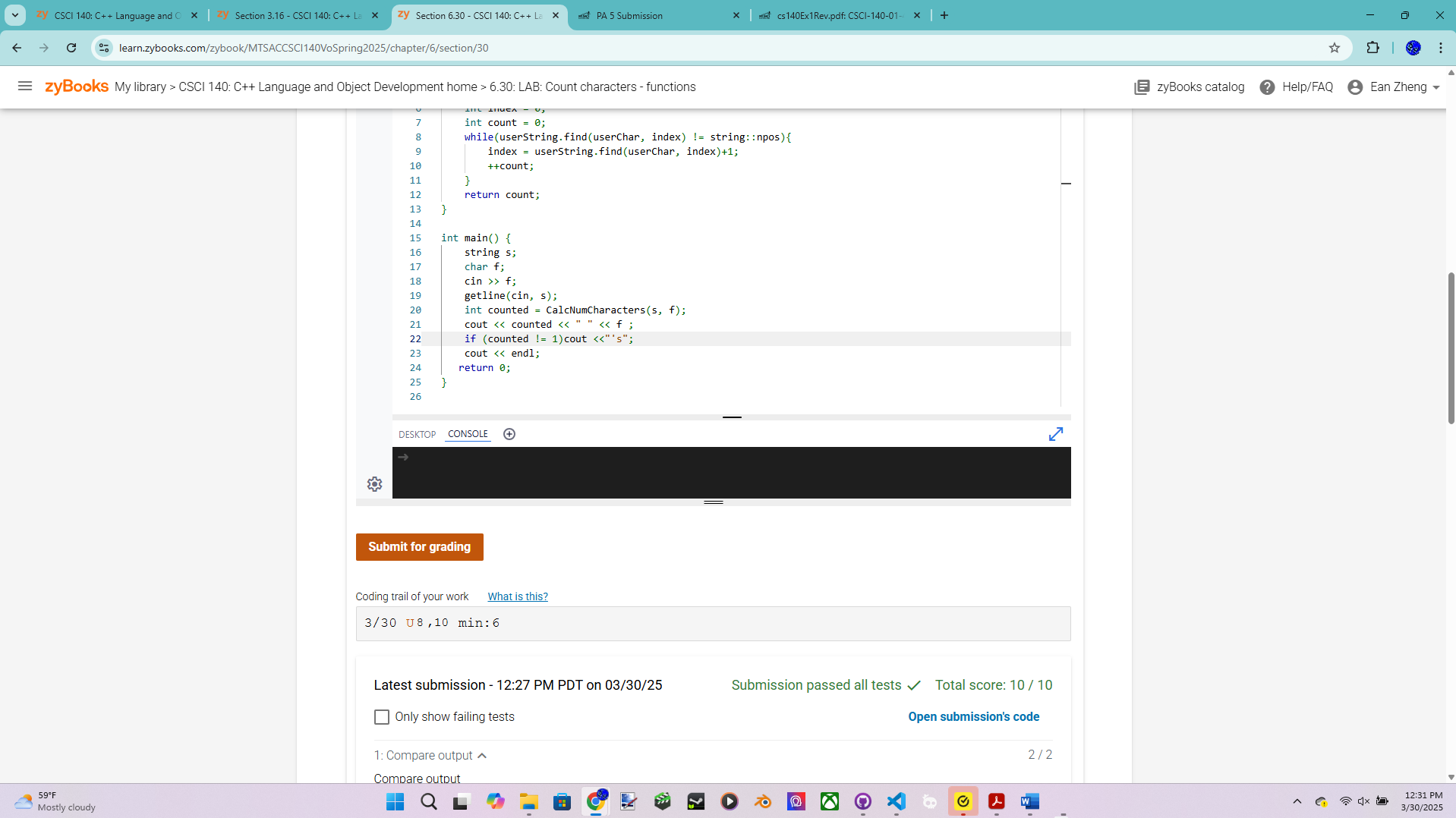
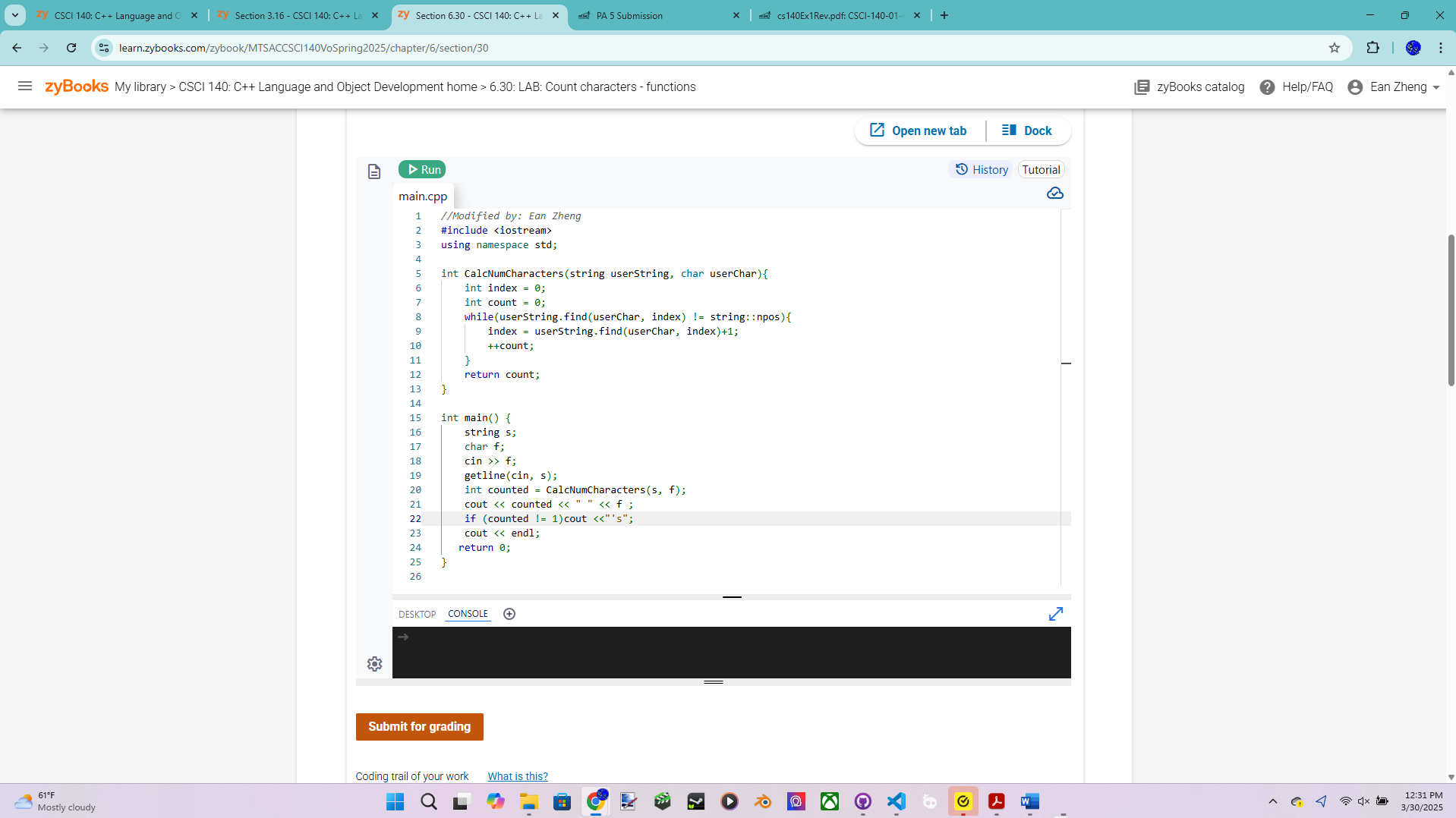
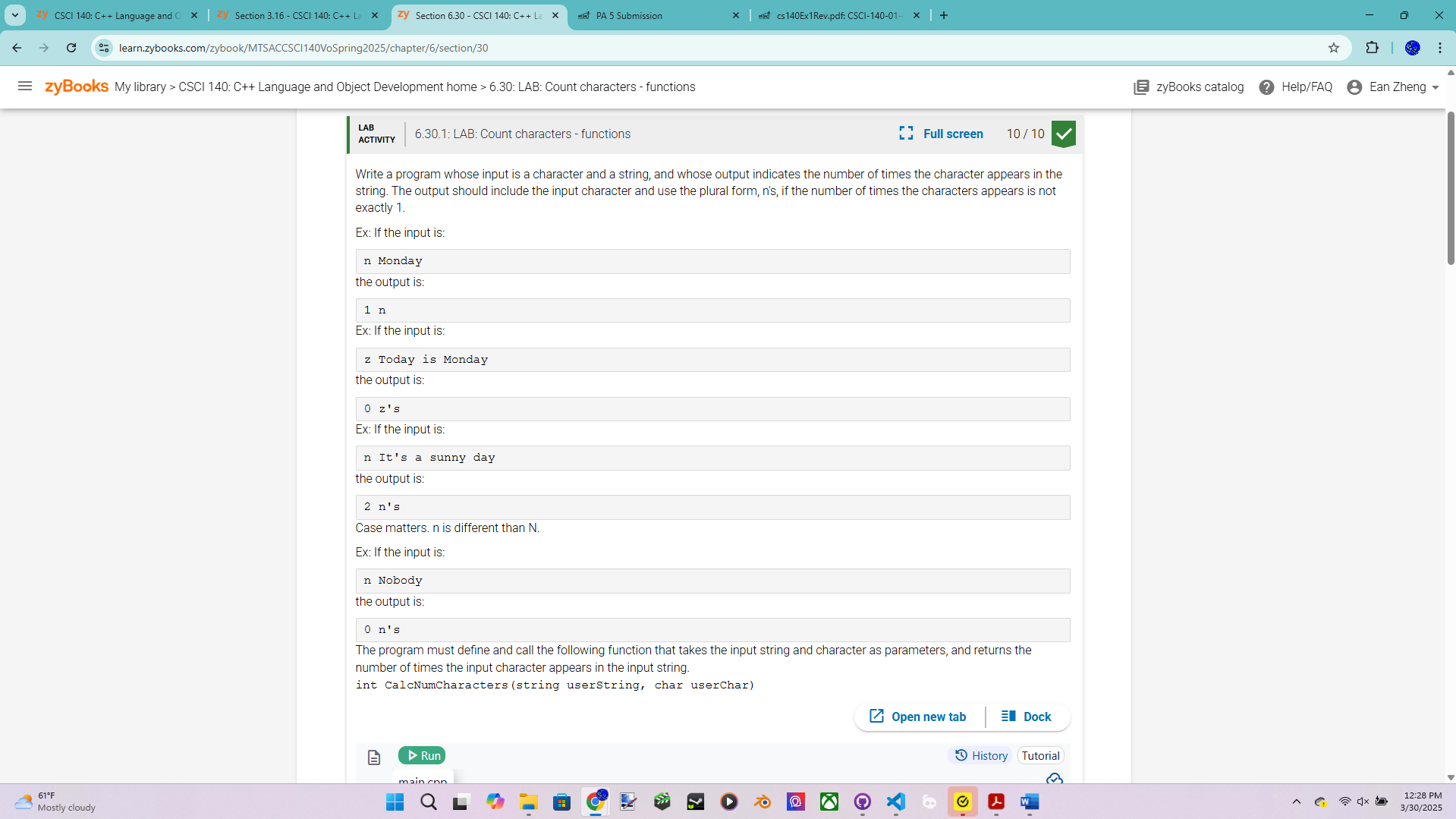
## CSCI 140 PA 5 Submission

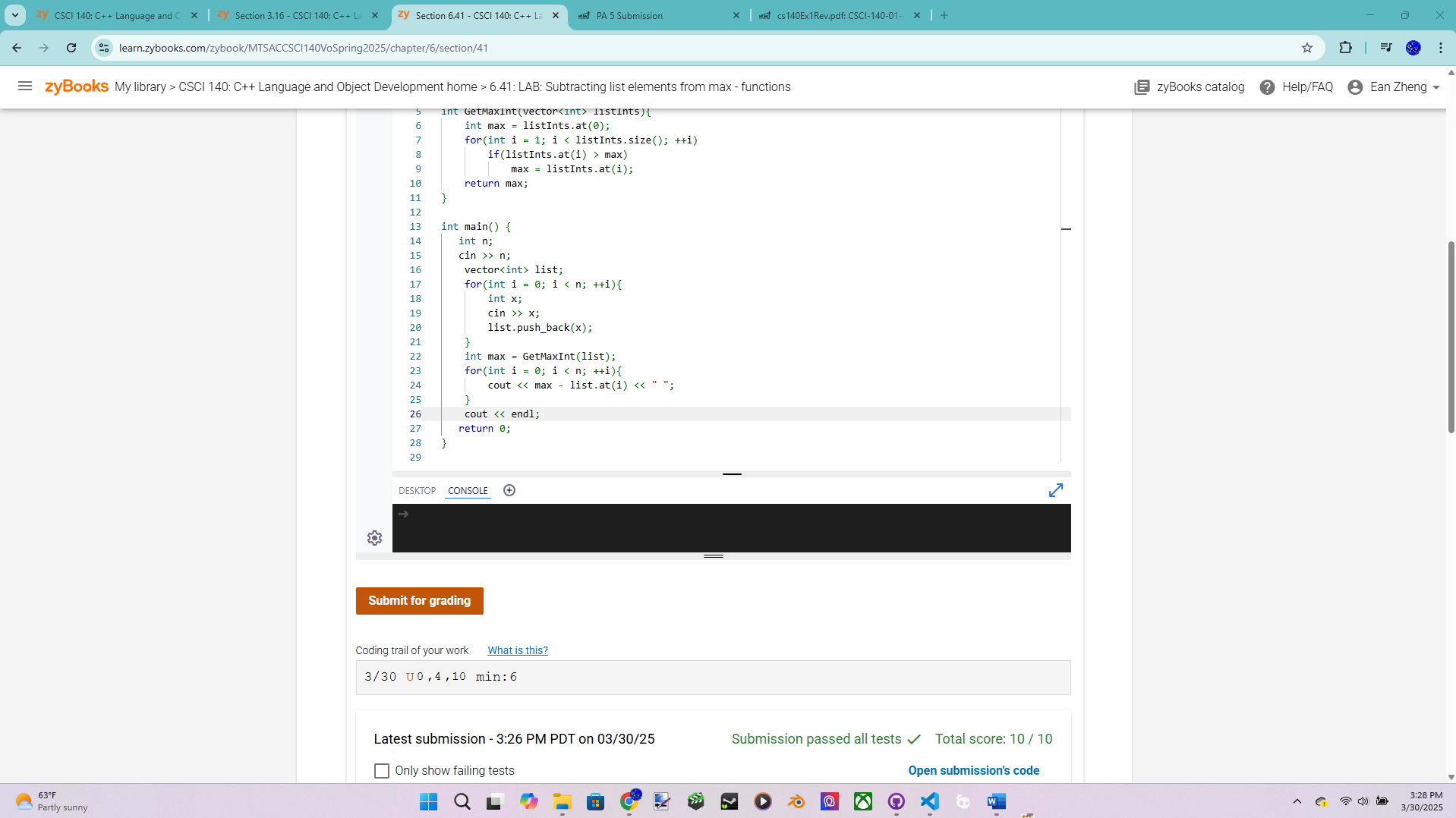
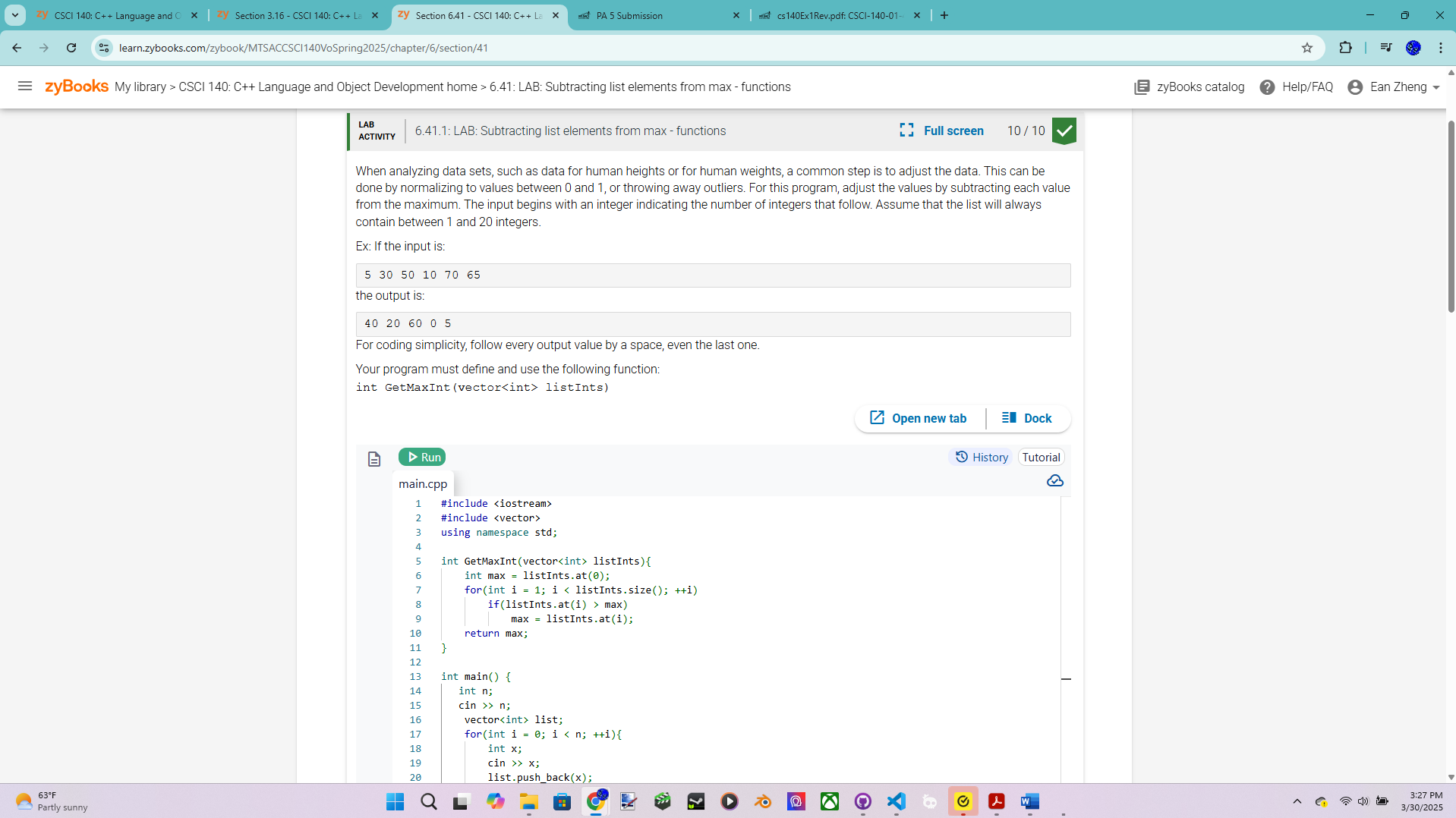
## Due Date:4/2/2025 Late (date and time):\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## Name(s):Ean Zheng

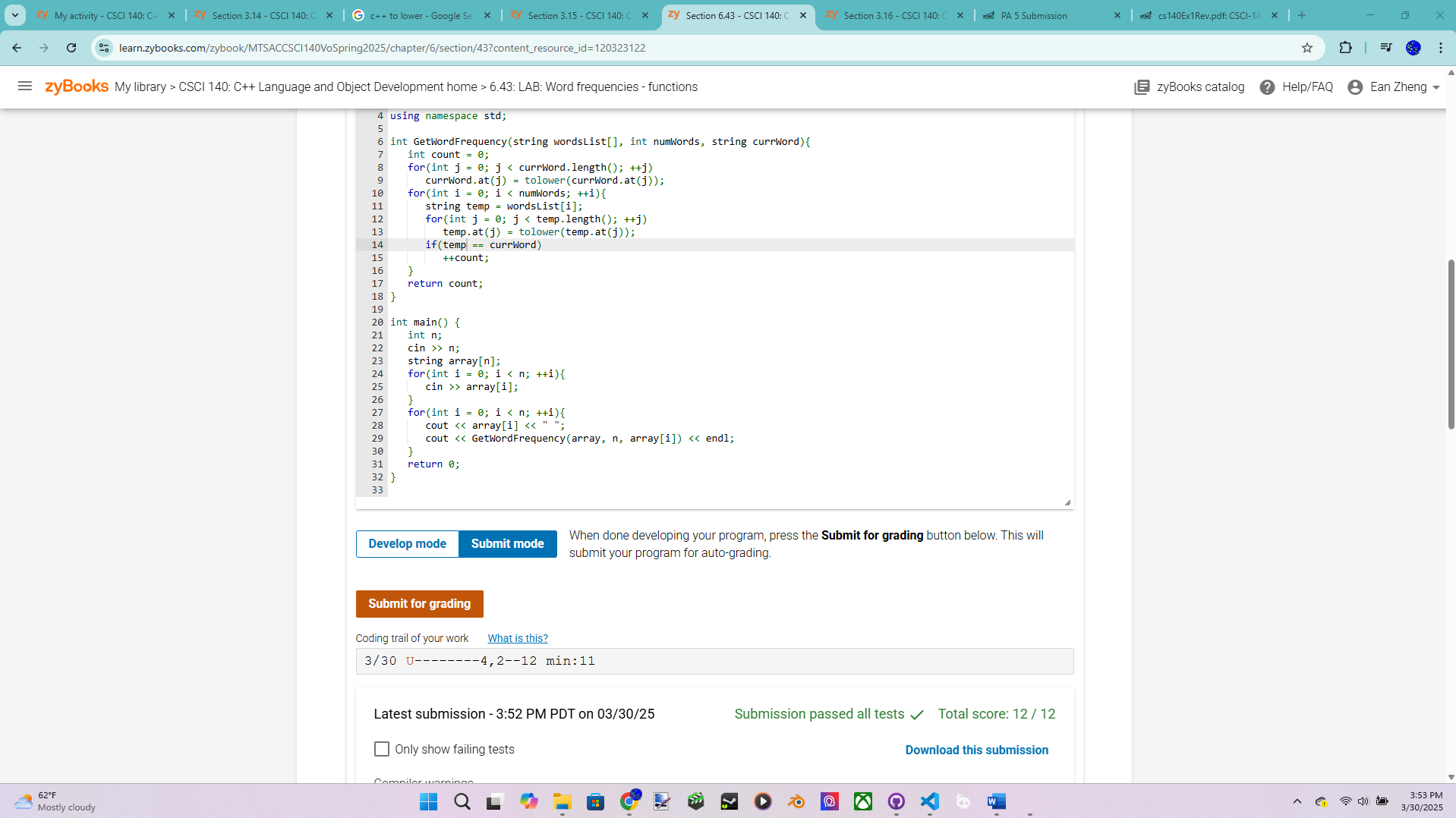
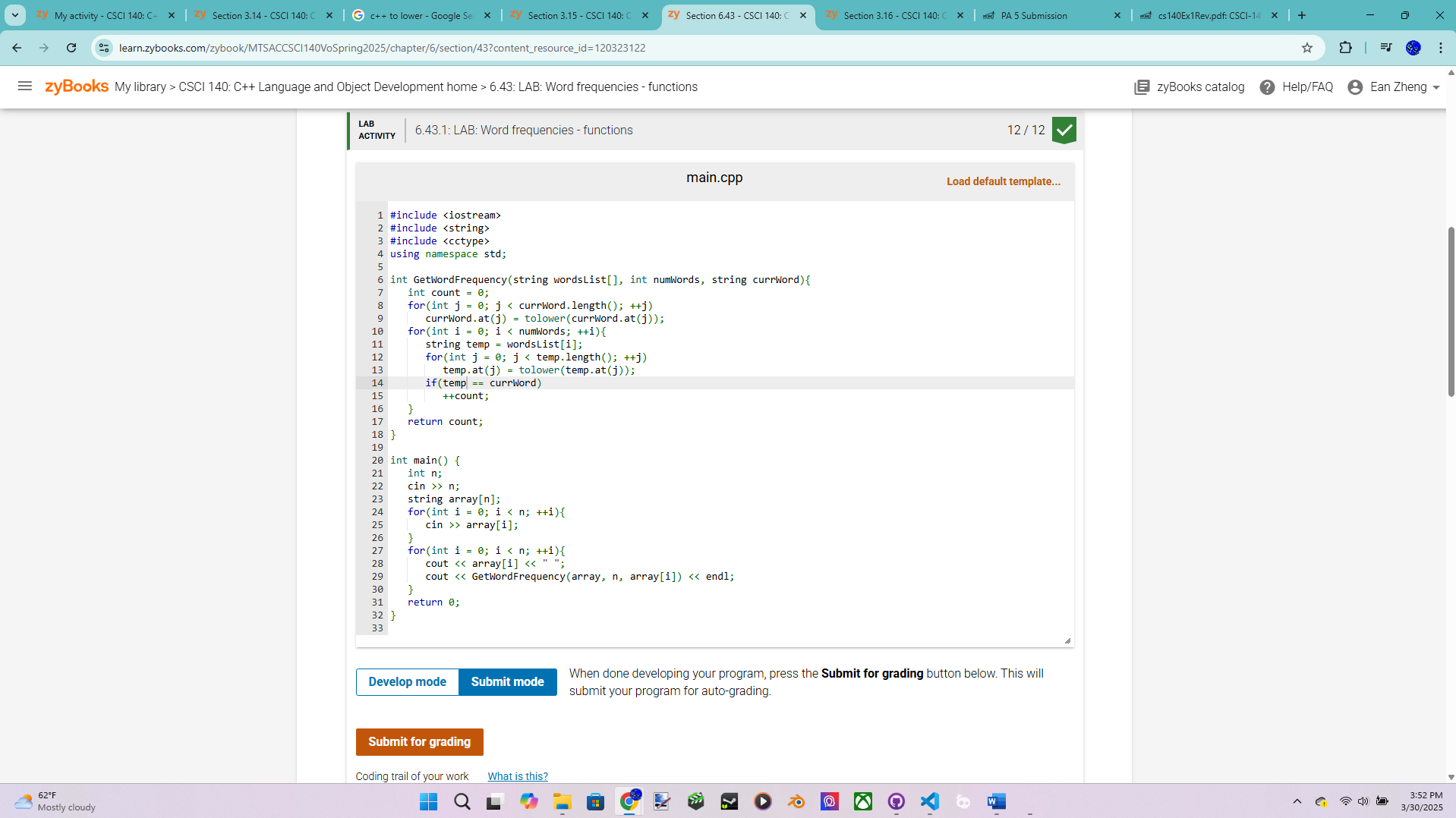
Exercise 1 – 6.30 LAB: Count characters – functions



Exercise 2 – 6.41 LAB: Subtracting list elements from max – functions



Exercise 3 – 6.43 LAB: Word frequencies – functions (classic mode only)



Exercise 4 – Large Integers version 2 – more points for this exercise

Modify your previous version to add two large integers and output the result if it is valid.  
You must utilize functions and here are the three required functions: convert an  
operand as string to an int array or an int vector, add valid operands (two big integers as  
two int arrays or two int vectors, one int array or an int vector as result, return true for  
valid operation and return false for overflow), and output one big integer in required  
format (big integer as one int array or an int vector). Think about the best way to set up  
these functions. Use a sentinel loop to stop the input where 0 % 0 is the sentinel value.

Source code below:

/\* Program: Large Integers Program Version 2 for Exercise 4, PA Submission 5

Author: Ean Zheng

Class: CSCI 140

Date: 3/30/2025

Description:

I certify that the code below is my own work.

Exception(s): N/A

\*/

#include <cstring>

#include <iostream>

using namespace std;

void convertToArray(string value, int array[]){

for (int i = 0; i < value.length(); i++)

{

array[i] = value.at(value.length()-1-i)-48;

}

for (int i = value.length(); i < 25; i++){

array[i] = 0;

}

}

bool addNumbers(int number1[], int number2[], int result[]){

int addon = 0;

for (int i = 0; i < 25; i++)

{

result[i] = number1[i] + number2[i] + addon;

addon = result[i]/10;

result[i] = result[i]%10;

if(i == 24 && addon != 0)

return false;

}

return true;

}

void outputNumber(int number[]){

bool reached = false;

for (int i = 24; i >= 0; i--)

{

if(reached == false && number[i] != 0){

reached = true;

cout << number[i];

}

else if(reached == true){

cout << number[i];

}

}

}

int main()

{

cout << "Author: Ean Zheng" << endl;

string value1;

string value2;

char sign = ' ';

int array1[25];

int array2[25];

int result[25];

while(value1 != "0" && sign != '%' && value2 != "0"){

cout << "Enter an expression --> ";

cin >> value1 >> sign >> value2;

if(sign == '+' && value1.length()<=25 && value2.length()<=25){

convertToArray(value1, array1);

convertToArray(value2, array2);

outputNumber(array1);

cout << " + ";

outputNumber(array2);

cout << " = ";

if(addNumbers(array1, array2, result))

outputNumber(result);

else

cout << "overflow";

cout << endl << endl;

}else if (value1 != "0" && sign != '%' && value2 != "0"){

cout << "Invalid operand(s)" << endl << endl;

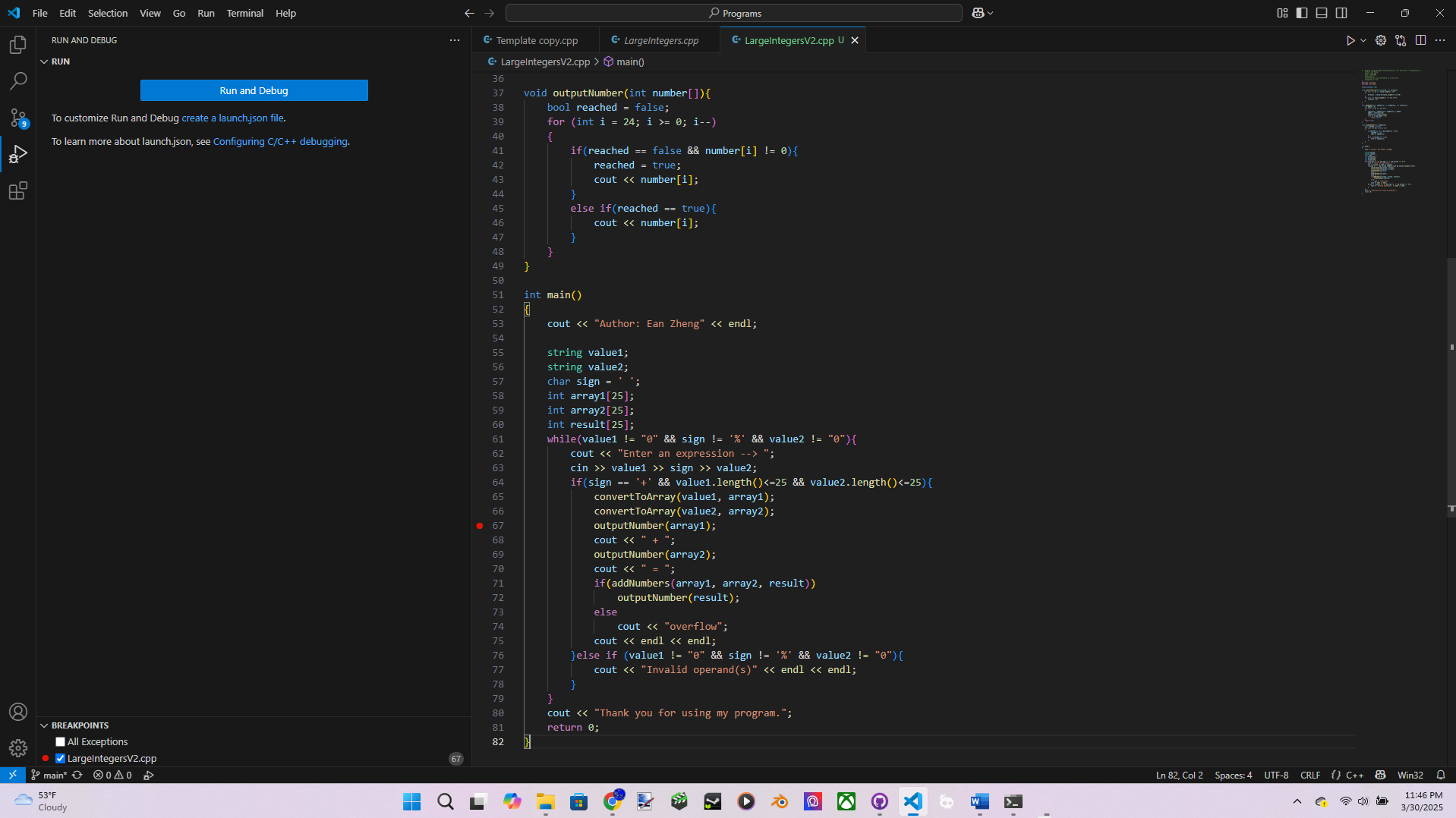
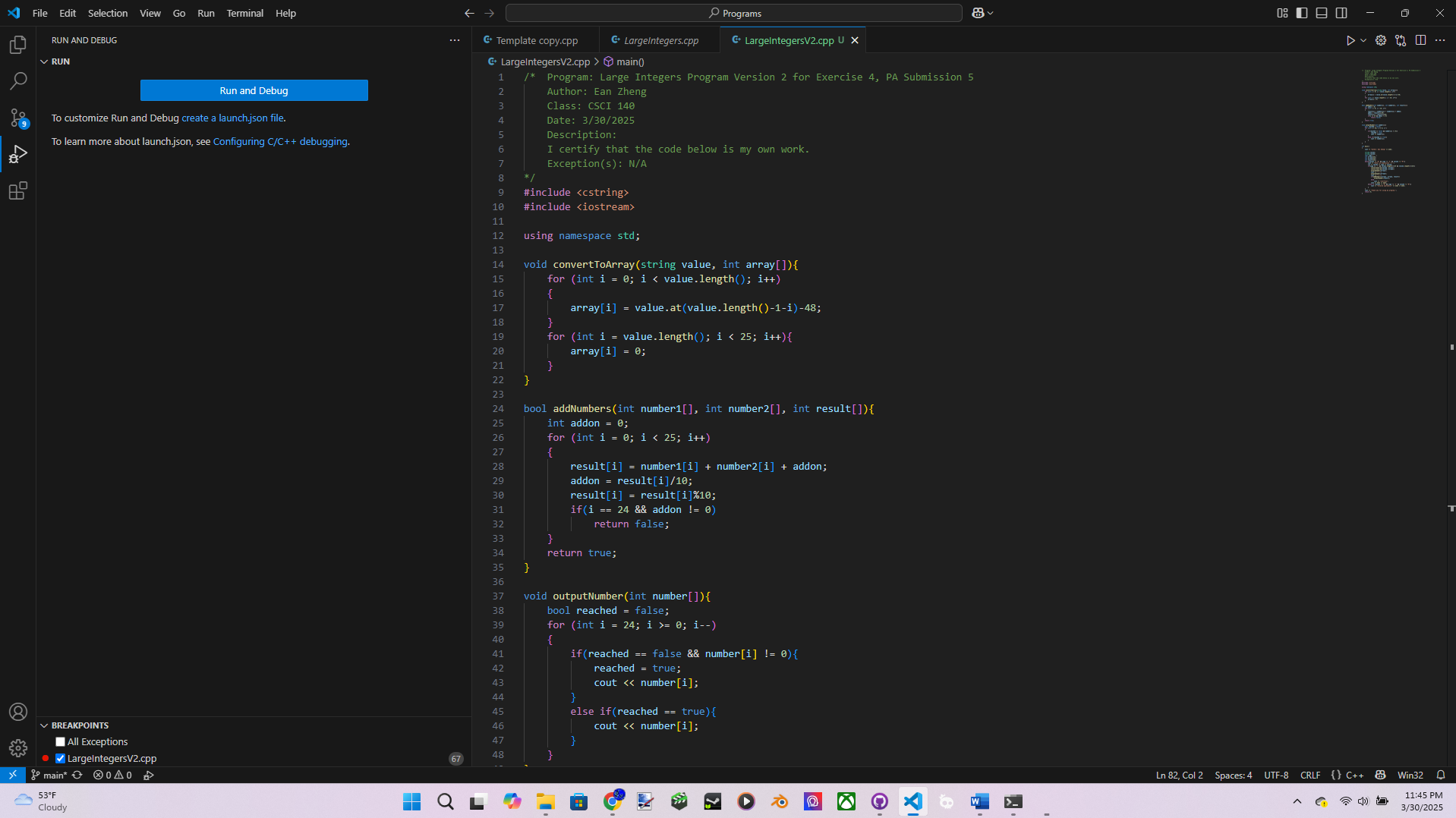
}

}

cout << "Thank you for using my program.";

return 0;

}



Input/output below:

Author: Ean Zheng

Enter an expression --> 1234 + 72

1234 + 72 = 1306

Enter an expression --> 987654321 + 123456789

987654321 + 123456789 = 1111111110

Enter an expression --> 999999999999999999999999 + 1

999999999999999999999999 + 1 = 1000000000000000000000000

Enter an expression --> 9999999999999999999999999 + 1

9999999999999999999999999 + 1 = overflow

Enter an expression --> 99999999999999999999012345 + 123

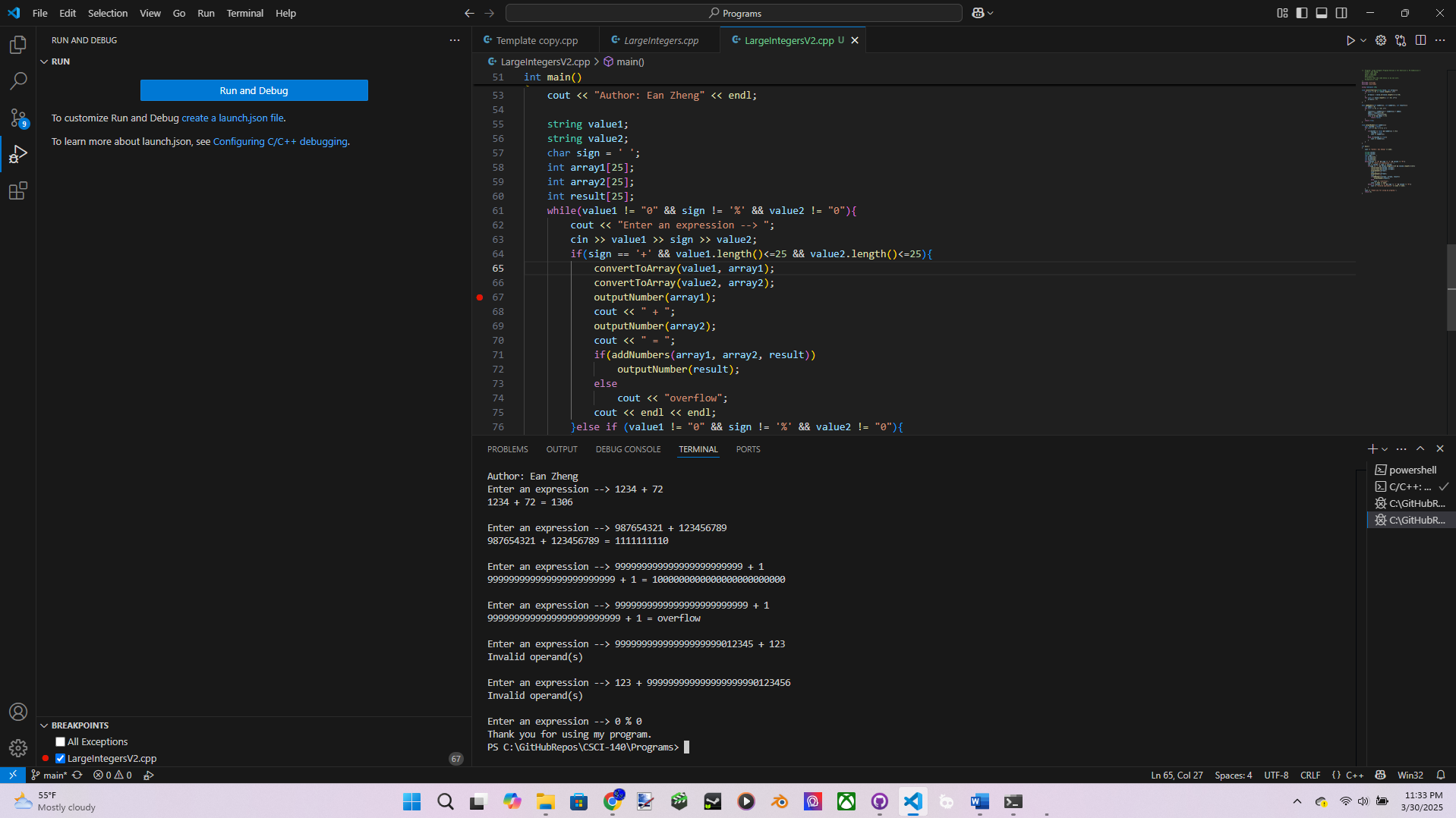
Invalid operand(s)

Enter an expression --> 123 + 999999999999999999990123456

Invalid operand(s)

Enter an expression --> 0 % 0

Thank you for using my program.



Question 1: List some good reasons for creating and using functions in C++ programs.

To make the program more understandable and readable, prevent repetitive code usage, reduce lines of code, and to just make coding easier and more convenient in general.

Question 2: Why is it not a good idea to output a calculated result inside a calculated  
function that returns the calculated result?

Getting the result from the returning function in the main program is more convenient and allows further use, while also most likely requiring less code lines. It is also a good habit to restrict output statements to the main program if you can.

Extra Credit (2 points): Modify Exercise 4 so you can also perform subtraction with  
large number as well by adding a function sub (subtracting one big integer from a big  
integer as two int arrays, one int array as result, return true for valid operation, and return  
false when first operand is less than second operand). You can submit just this version,  
but make sure to show test cases for both addition and subtraction. Follow the interface  
below and you must try the following test cases:  
Enter an expression --> 1234 - 72<Enter>  
1234 - 72 = 1162  
Enter an expression --> 72 - 1234<Enter>  
72 - 1234 = result is negative  
Enter an expression --> 100 - 105<Enter>  
100 - 105 = result is negative  
Enter an expression --> 987654321 - 123456789<Enter>  
987654321 - 123456789 = 864197532  
Enter an expression --> 12345 - 12345<Enter>  
12345 - 12345 = 0  
// 25 digits - 1 digit = 25 digits  
Enter an expression --> 9999999999999999999999999 - 1<Enter>  
9999999999999999999999999 - 1 = 9999999999999999999999998  
Enter an expression --> 0 % 0<Enter>  
Thank you for using my program.

Source Code:

/\* Program: Large Integers Program Version 2 for Exercise 4, PA Submission 5

Author: Ean Zheng

Class: CSCI 140

Date: 3/30/2025

Description:

I certify that the code below is my own work.

Exception(s): N/A

\*/

#include <cstring>

#include <iostream>

#include <cmath>

using namespace std;

void convertToArray(string value, int array[]){

for (int i = 0; i < value.length(); i++)

{

array[i] = value.at(value.length()-1-i)-48;

}

for (int i = value.length(); i < 25; i++){

array[i] = 0;

}

}

bool addNumbers(int number1[], int number2[], int result[]){

int addon = 0;

for (int i = 0; i < 25; i++)

{

result[i] = number1[i] + number2[i] + addon;

addon = result[i]/10;

result[i] = result[i]%10;

if(i == 24 && addon != 0)

return false;

}

return true;

}

bool subtractNumbers(int number1[], int number2[], int result[]){

int subon = 0;

for (int i = 0; i < 25; i++)

{

result[i] = number1[i] - number2[i] - subon;

if(result[i] < 0){

subon = abs(floor(result[i]/10.0));

result[i] = 10 + result[i];

}else

subon = 0;

if(i == 24 && subon != 0)

return false;

}

return true;

}

void outputNumber(int number[]){

bool reached = false;

for (int i = 24; i >= 0; i--)

{

if(reached == false && number[i] != 0){

reached = true;

cout << number[i];

}

else if(reached == true){

cout << number[i];

}

if (reached == false && i == 0){

cout << 0;

}

}

}

int main()

{

cout << "Author: Ean Zheng" << endl;

string value1;

string value2;

char sign = ' ';

int array1[25];

int array2[25];

int result[25];

while(value1 != "0" && sign != '%' && value2 != "0"){

cout << "Enter an expression --> ";

cin >> value1 >> sign >> value2;

if(sign == '+' && value1.length()<=25 && value2.length()<=25){

convertToArray(value1, array1);

convertToArray(value2, array2);

outputNumber(array1);

cout << " + ";

outputNumber(array2);

cout << " = ";

if(addNumbers(array1, array2, result))

outputNumber(result);

else

cout << "overflow";

cout << endl << endl;

}else if(sign == '-' && value1.length()<=25 && value2.length()<=25){

convertToArray(value1, array1);

convertToArray(value2, array2);

outputNumber(array1);

cout << " - ";

outputNumber(array2);

cout << " = ";

if(subtractNumbers(array1, array2, result))

outputNumber(result);

else

cout << "result is negative";

cout << endl << endl;

}else if (value1 != "0" && sign != '%' && value2 != "0"){

cout << "Invalid operand(s)" << endl << endl;

}

}

cout << "Thank you for using my program.";

return 0;

}

Input/Output:

Author: Ean Zheng

Enter an expression --> 1234 - 72

1234 - 72 = 1162

Enter an expression --> 72 - 1234

72 - 1234 = result is negative

Enter an expression --> 100 - 105

100 - 105 = result is negative

Enter an expression --> 987654321 - 123456789

987654321 - 123456789 = 864197532

Enter an expression --> 12345 - 12345

12345 - 12345 = 0

Enter an expression --> 9999999999999999999999999 - 1

9999999999999999999999999 - 1 = 9999999999999999999999998

Enter an expression --> 0 % 0

Thank you for using my program.

