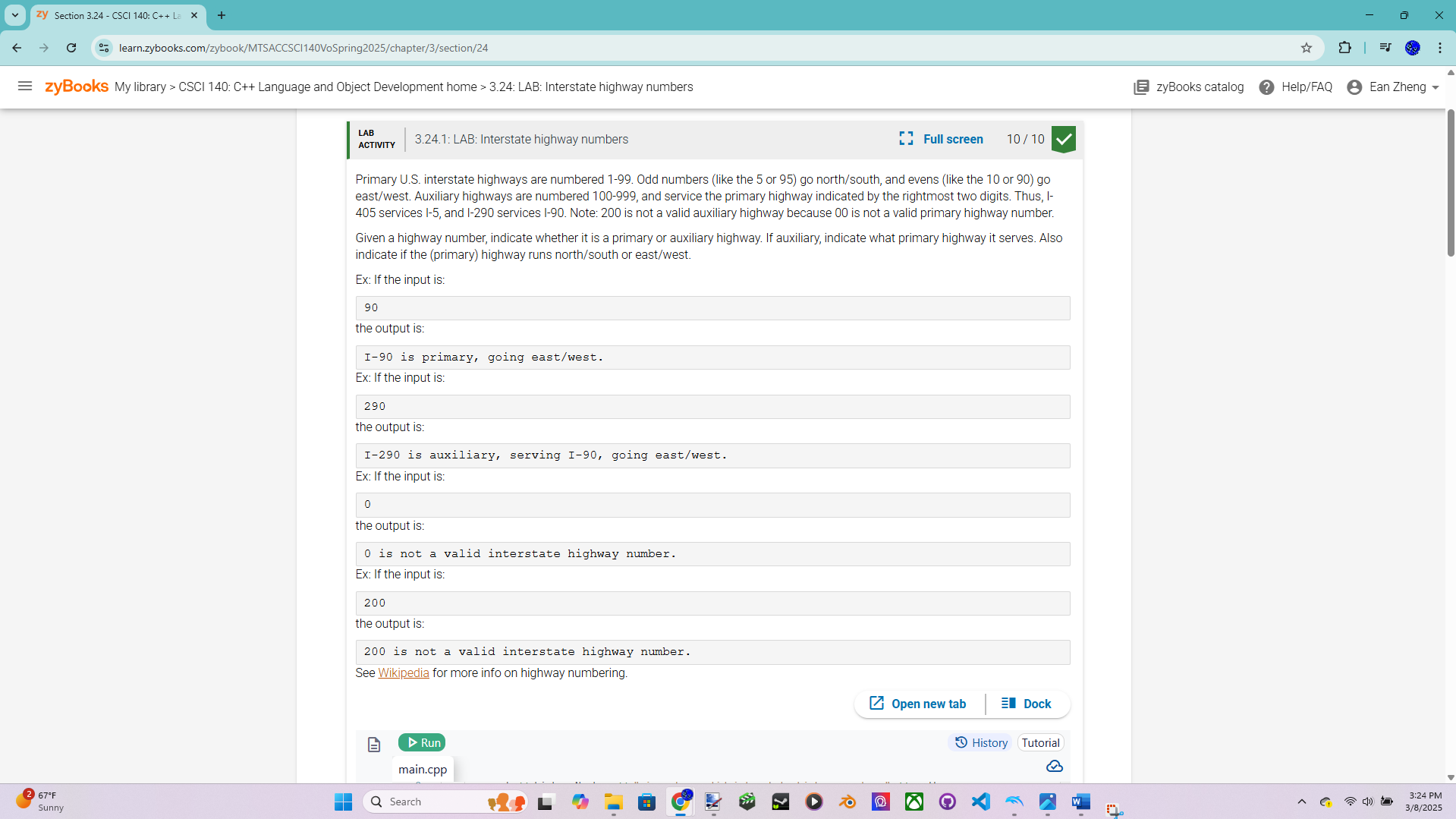
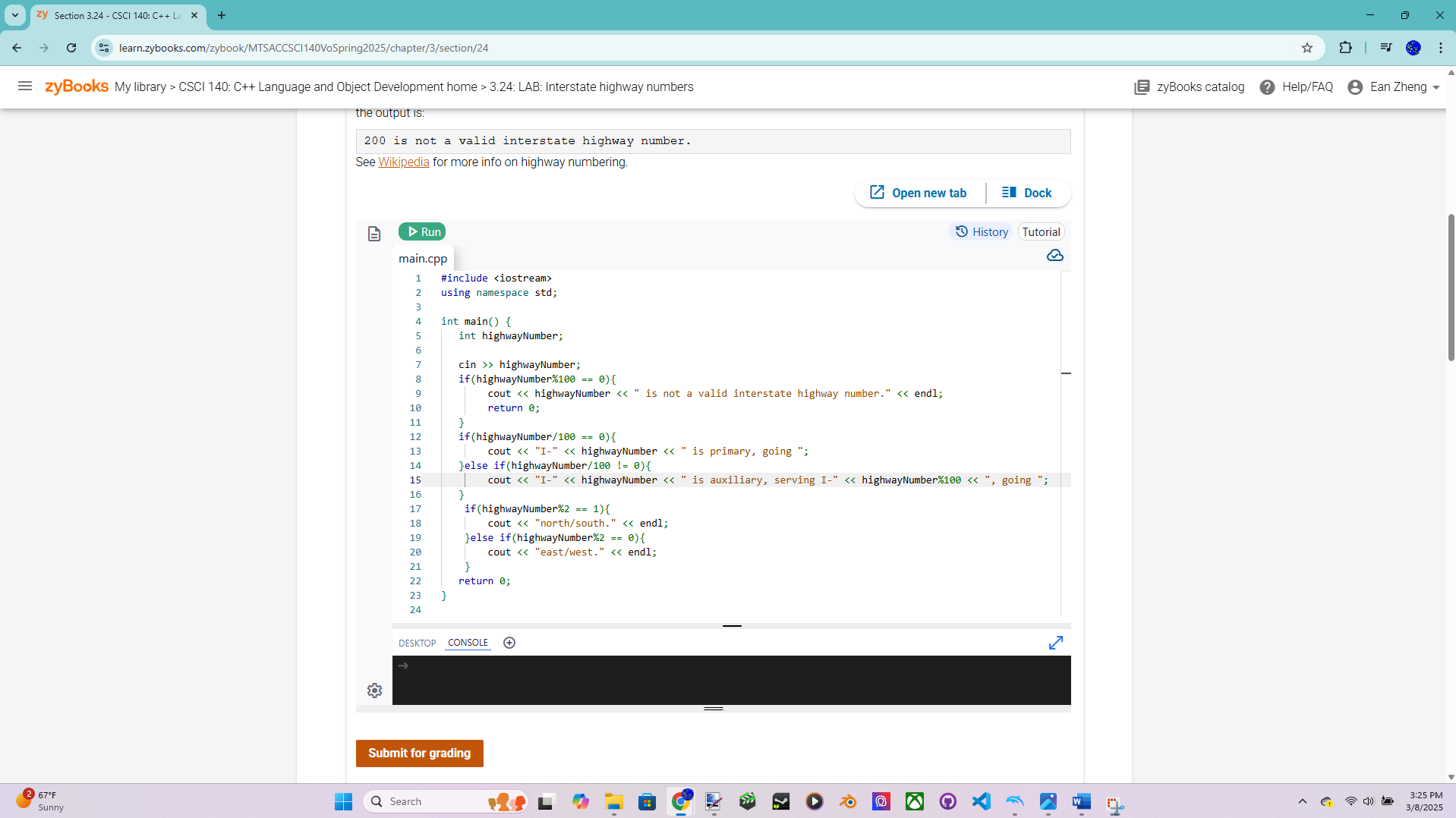
## CSCI 140 PA 2 Submission

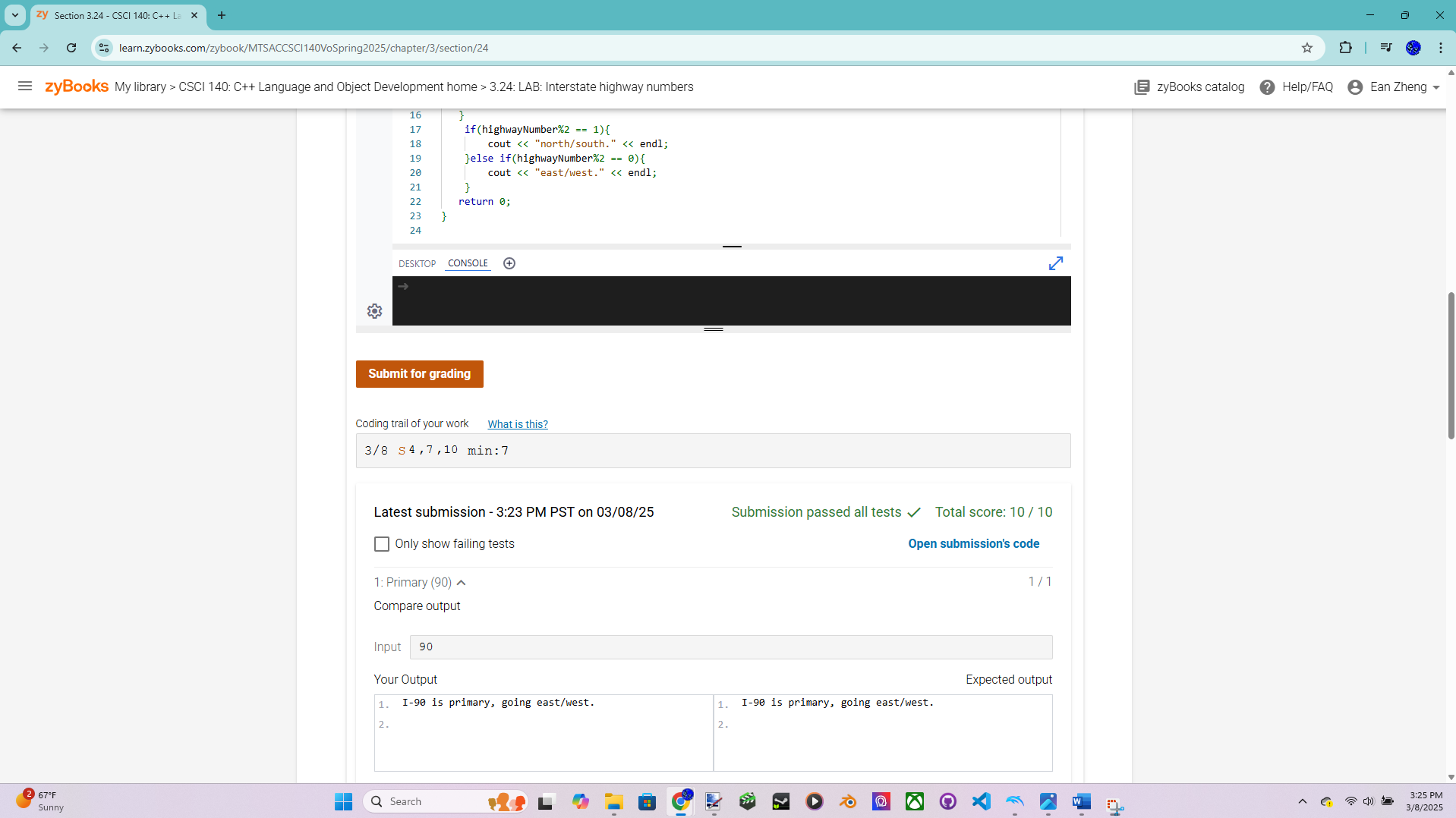
## Due Date:3/10/2025 Late (date and time):\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## Name: Ean Zheng

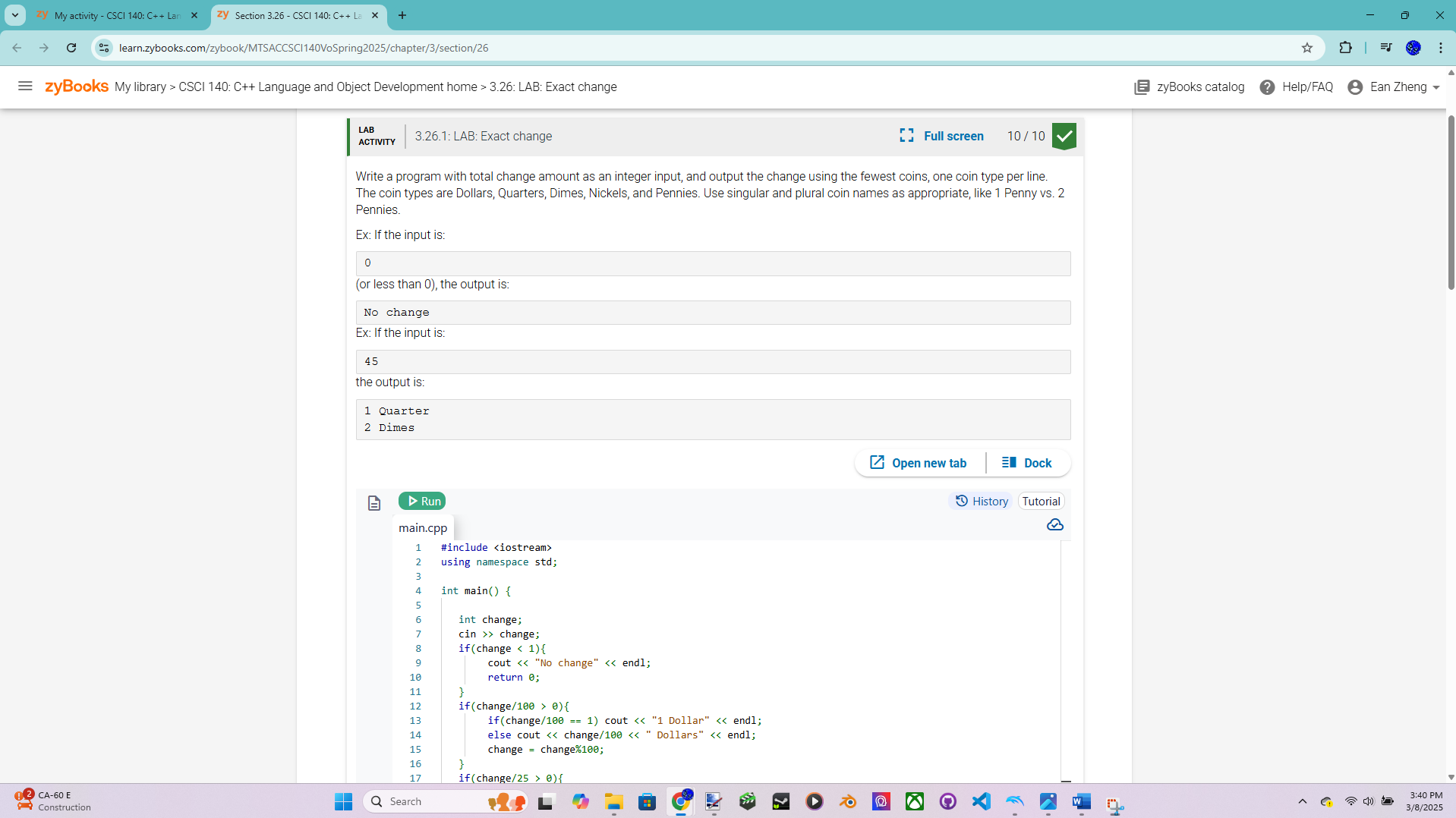
Exercise 1 – zyBook 3.24 LAB: Interstate highway numbers

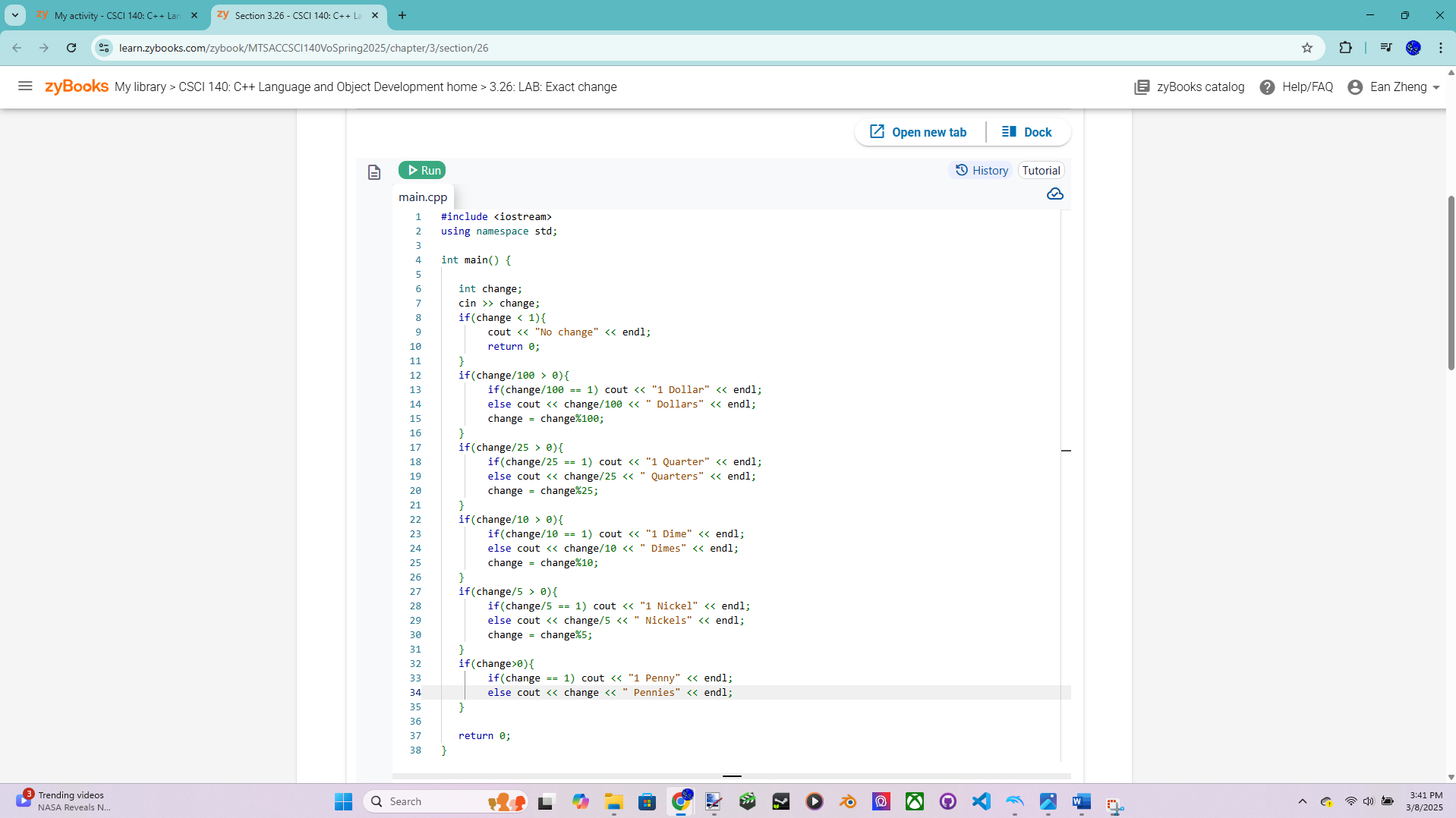


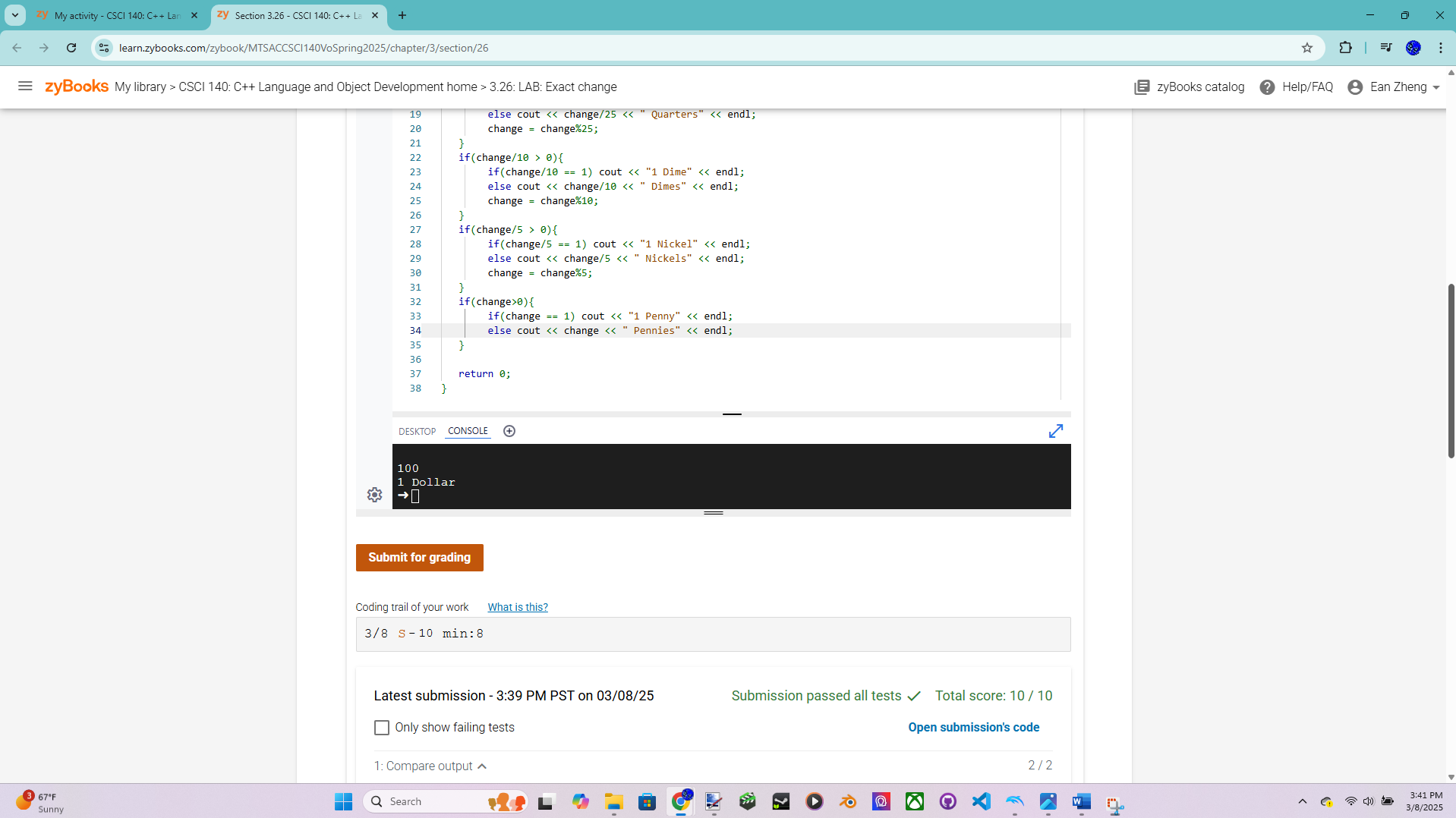




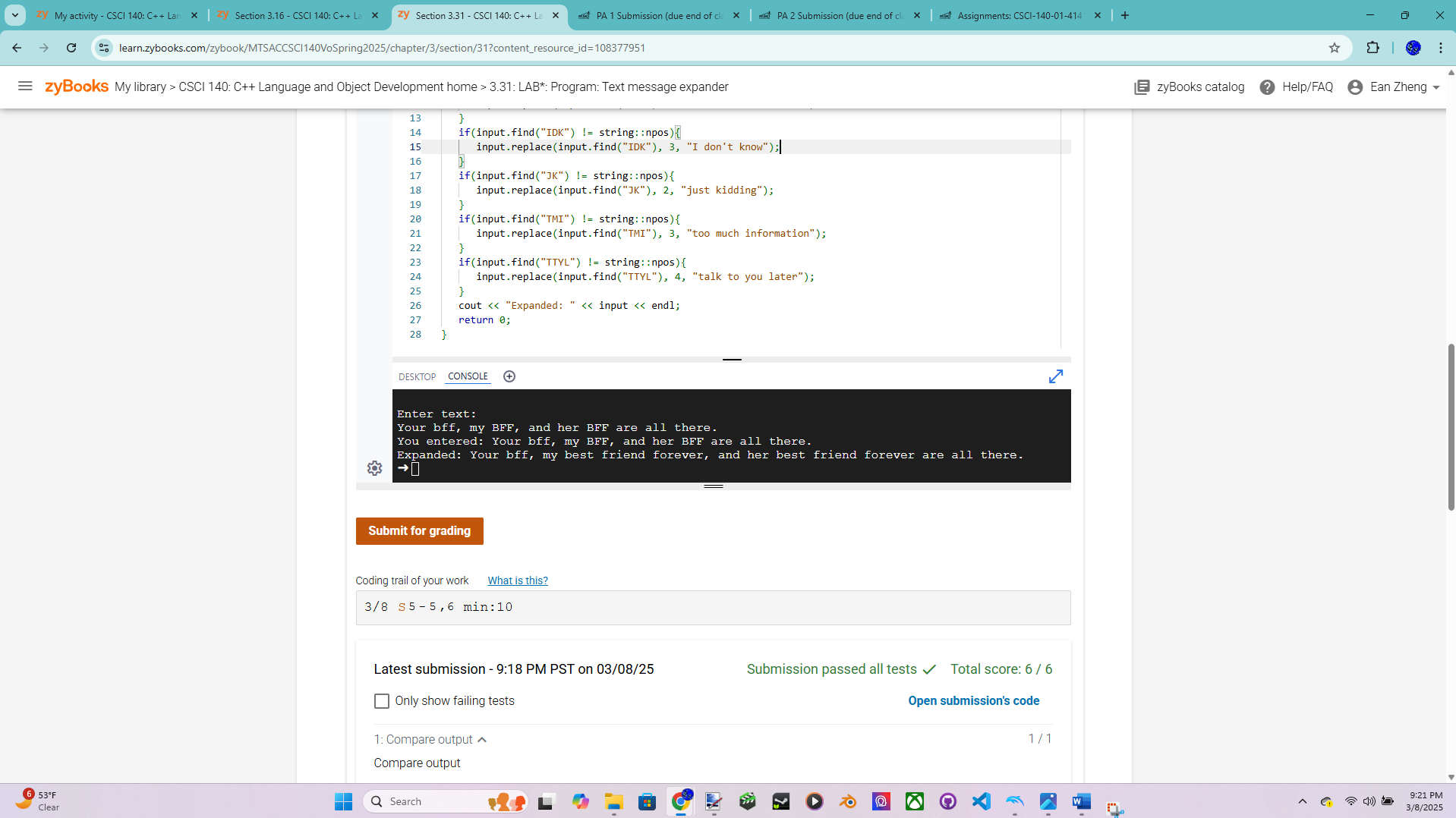
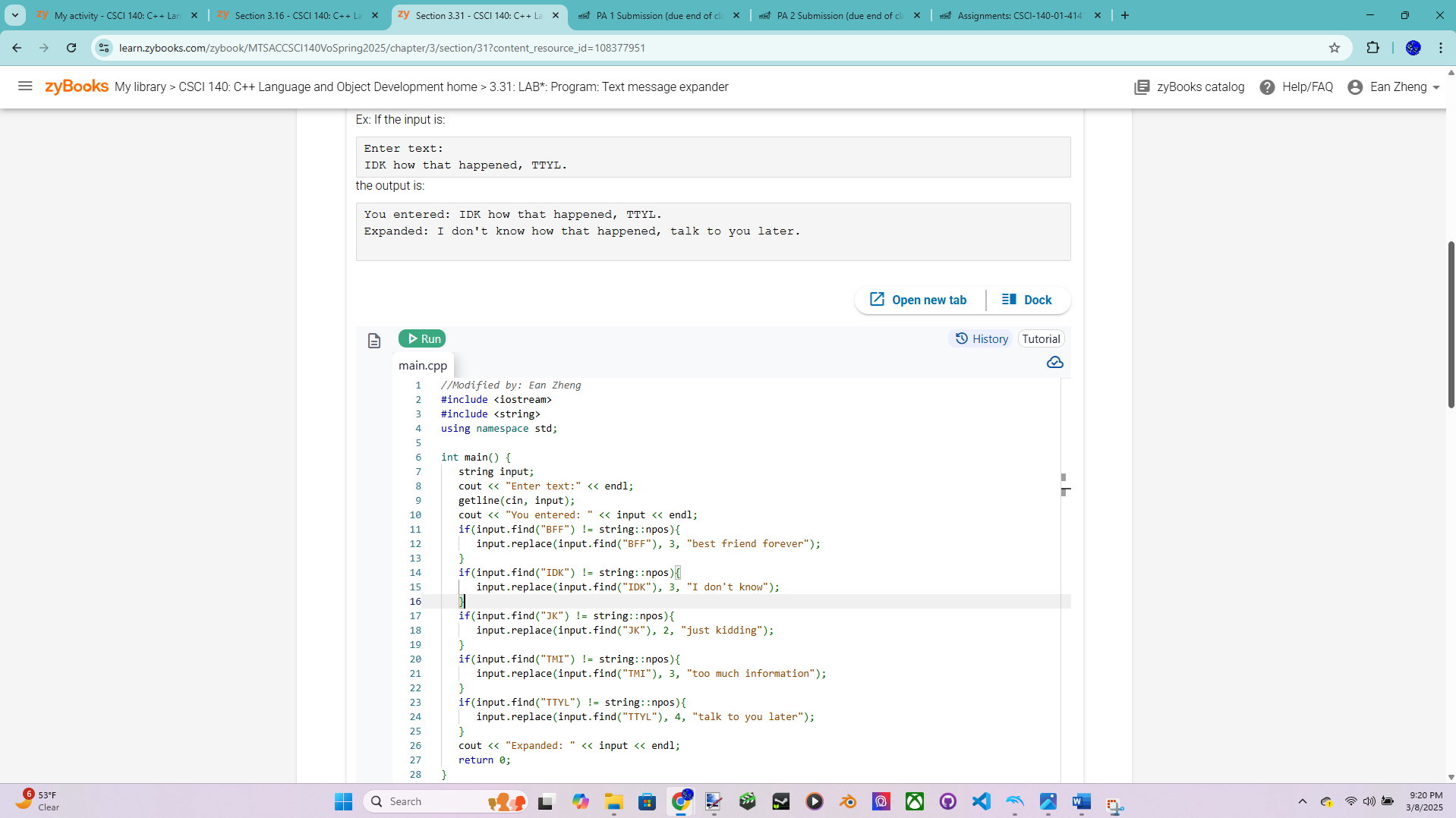
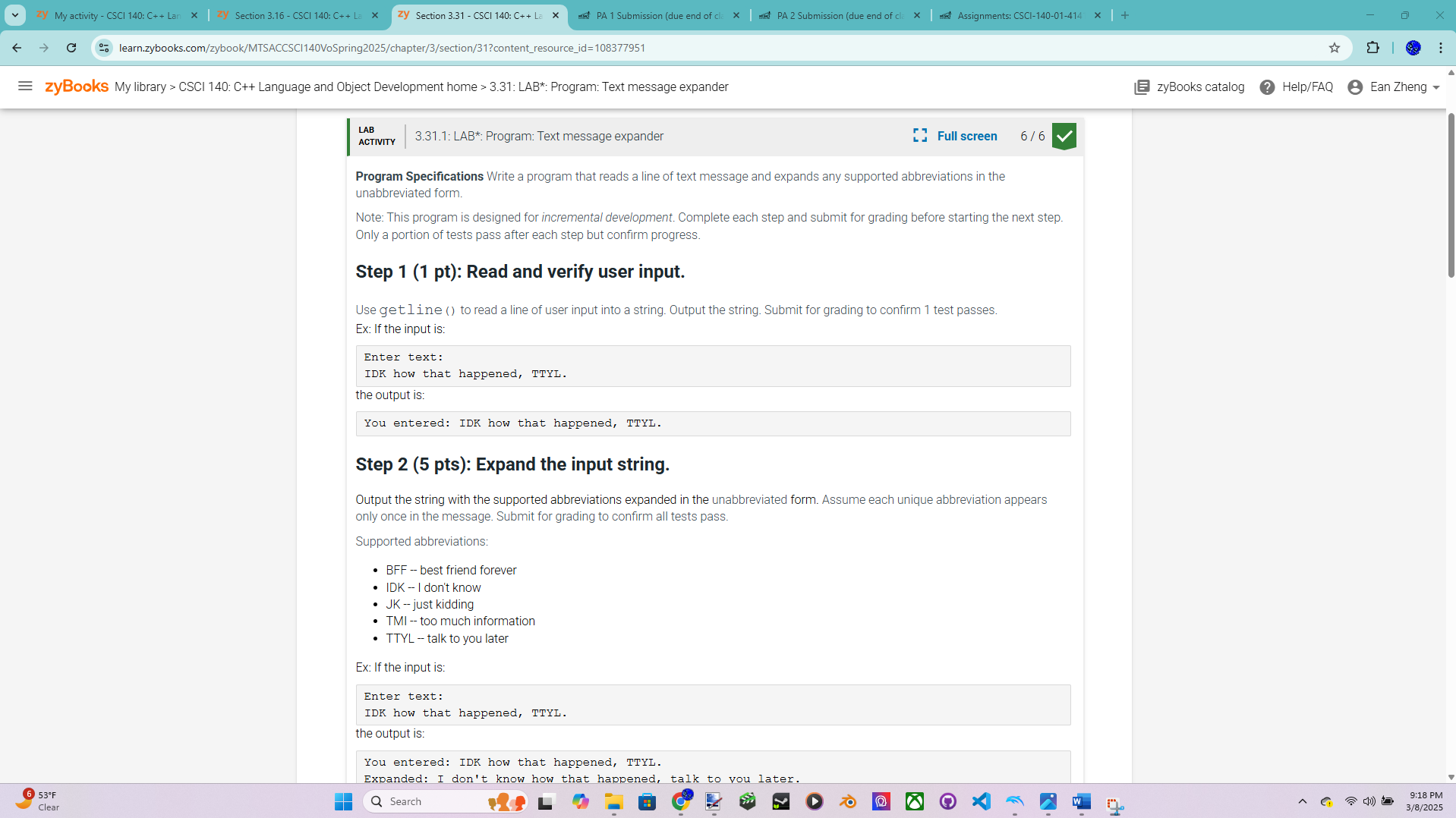
Exercise 2 – zyBook 3.26 LAB: Exact change







Exercise 3 – zyBook 3.31 LAB\*: Program: Text message expander



Exercise 4 – Simple Vending Machine Version 1 – more points for this exercise

Your program is supposed to read an integer value between 0 and 100 (inclusive),  
representing the amount of a purchase in cents from a vending machine. Produce an error  
message if the input value is not in that range. If the input is valid, determine the amount  
of change that would be returned from one dollar, and print the number of quarters,  
dimes, and nickels. Since pennies are not available, round the changes to the nearest  
multiple of 5 as needed. Assume that there are exactly 2 quarters, 2 dimes, and 2 nickels  
in the machine at the beginning. It is important to maximize the coins with the  
highest value first and utilize the next denomination if the current denomination is  
exhausted. Reject a valid purchase if it cannot be processed (i.e., not enough available  
coins to make the change). Follow the format below and you must plan and write  
down the pseudocode before attempting your code on the computer.

Pseudocode below if applicable:

Initiate purchaseamount and change int variables, and initiate quarters, dimes, and nickels int variables with value 0.

Announce program name and author, say default amounts of quarters, dimes, and nickles, add a blank line, and prompt user to input value with valid range.

When user inputs value, assign purchaseamount to inputed value and show it.

Check if purchaseamount isn’t in range, if so, cout invalid amount error and return 0.

Assign change to 100 minus input.

If modulo of change by 5 isn’t 0, print change variable, then see if modulo of change by 5 is less than or equal to 2, or greater than or equal to 3. If less than or equal to 2, subtract modulo of change by 5 from change. If greater than or equal to 5, add 5 subtracted by module of change by 5 to change. (module of change by 5 is parenthesized to assure correct operation.) After that, print current change variable as rounded change.

Initiate remainingchange variable as change

Make while loop for while quarters isn’t 2 and while remainingchange is bigger than or equal to 25, in the loop add 1 to quarter then subtract 25 from remainingchange

Do same while loops 2 times except for dimes and nickels. Compare remainingchange bigger than or equal to 10 in while condition and subtract 10 from remainingchange inside loop for dimes. Compare remainingchange bigger than or equal to 5 in while condition and subtract 5 from remainingchange inside loop for nickels.

Check if remainingchange is 0. If so, print change variable, and that it is given as the quarters, dimes, and nickels variables in separate lines. If not, print insufficient coins meesage with change variable, and that it can’t be processed.

Source code below:

/\* Program: Simple Vending Machine Version 1 for Exercise 4, PA Submission 2

Author: Ean Zheng

Class: CSCI 140

Date: 3/8/2025

Description:

I certify that the code below is my own work.

Exception(s): N/A

\*/

#include <iostream>

using namespace std;

int main()

{

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

int purchaseamount;

int change;

int quarters = 0;

int dimes = 0;

int nickels = 0;

cout << "Vending Machine Version 1 by Ean Zheng" << endl;

cout << "There are 2 quarters, 2 dimes, and 2 nickels." << endl << endl;

cout << "Enter a purchase amount [0 - 100] --> ";

cin >> purchaseamount;

cout << "You entered a purchase amount of " << purchaseamount << " cents." << endl;

if(purchaseamount < 0 || purchaseamount > 100){

cout << "You entered an invalid amount (not between 0 and 100).";

return 0;

}

change = 100-purchaseamount;

if(change%5 != 0){

cout << "Your change of " << change;

if(change%5 <= 2)

change -= change%5;

else if(change%5 >= 3)

change += 5 - (change%5);

cout << " cents is rounded to " << change << " cents." << endl;

}

int remainingchange = change;

while(quarters != 2 && remainingchange>=25){

quarters++;

remainingchange -= 25;

}

while(dimes != 2 && remainingchange>=10){

dimes++;

remainingchange -= 10;

}

while(nickels != 2 && remainingchange>=5){

nickels++;

remainingchange -= 5;

}

if(remainingchange == 0){

cout << "Your change of " << change << " cents is given as:" << endl;

cout << "quarter(s): " << quarters << endl;

cout << "dime(s): " << dimes << endl;

cout << "nickel(s): " << nickels << endl;

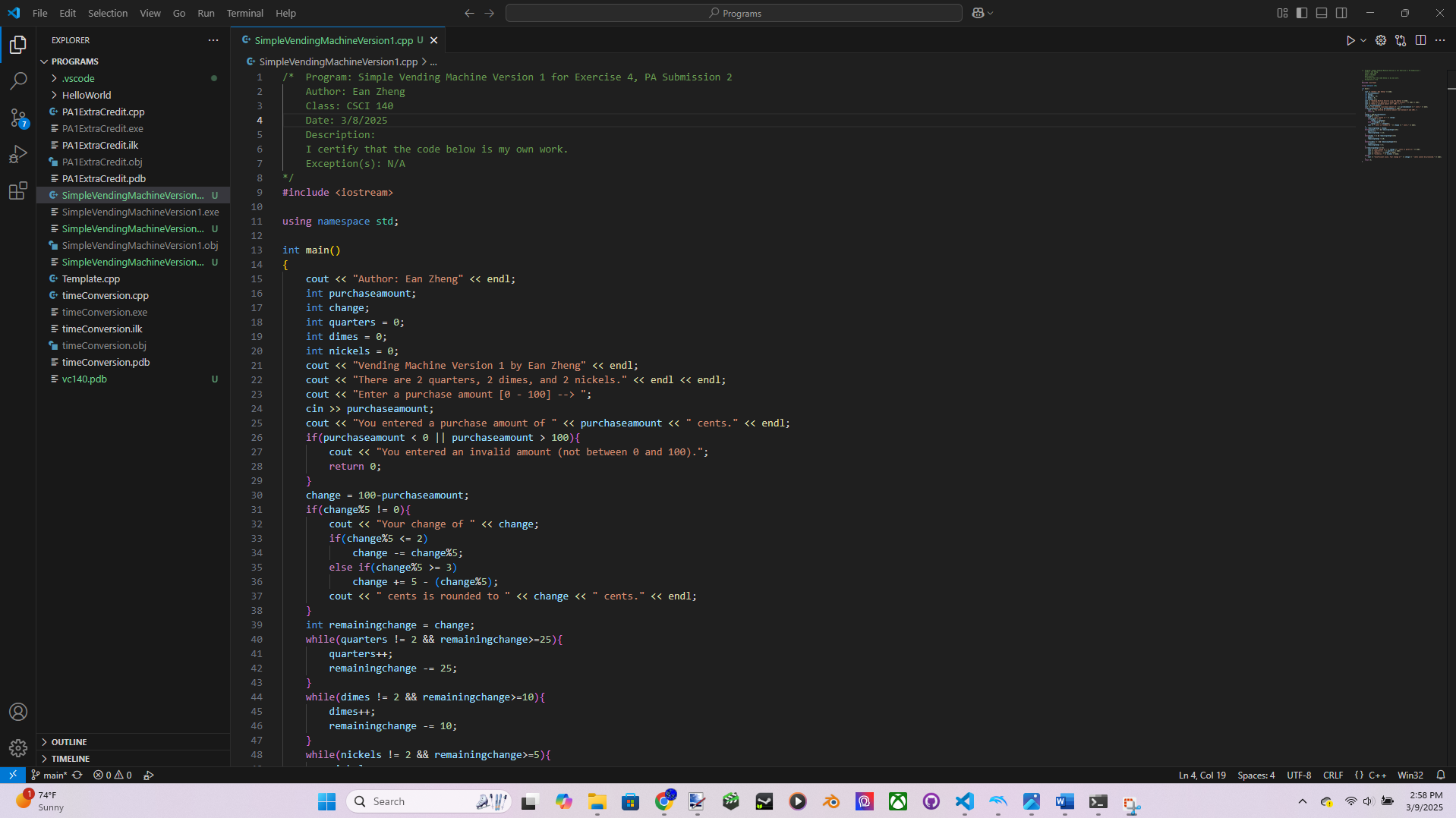
}else{

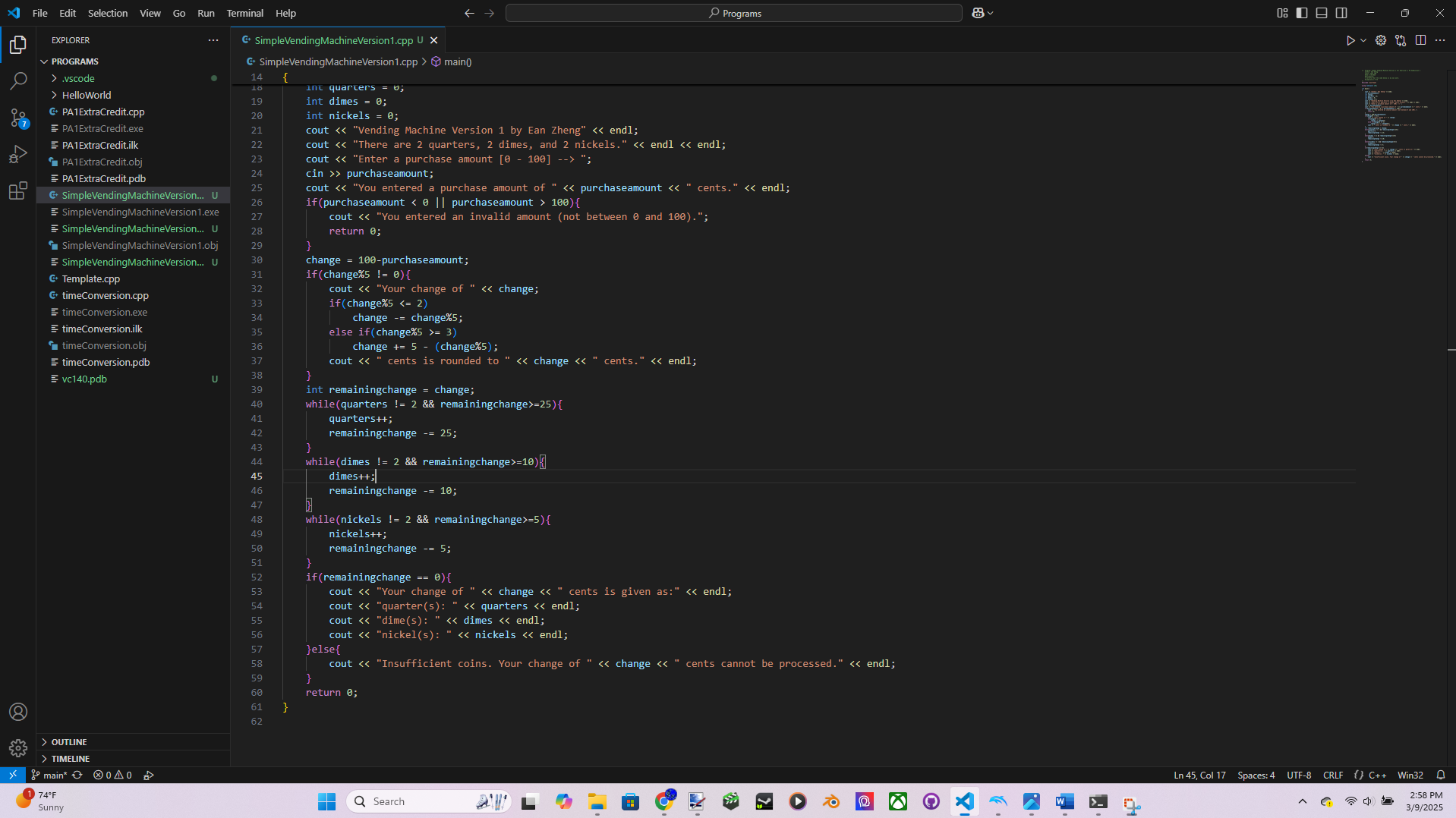
cout << "Insufficient coins. Your change of " << change << " cents cannot be processed." << endl;

}

return 0;

}





Input/output below:

Author: Ean Zheng

Vending Machine Version 1 by Ean Zheng

There are 2 quarters, 2 dimes, and 2 nickels.

Enter a purchase amount [0 - 100] --> 37

You entered a purchase amount of 37 cents.

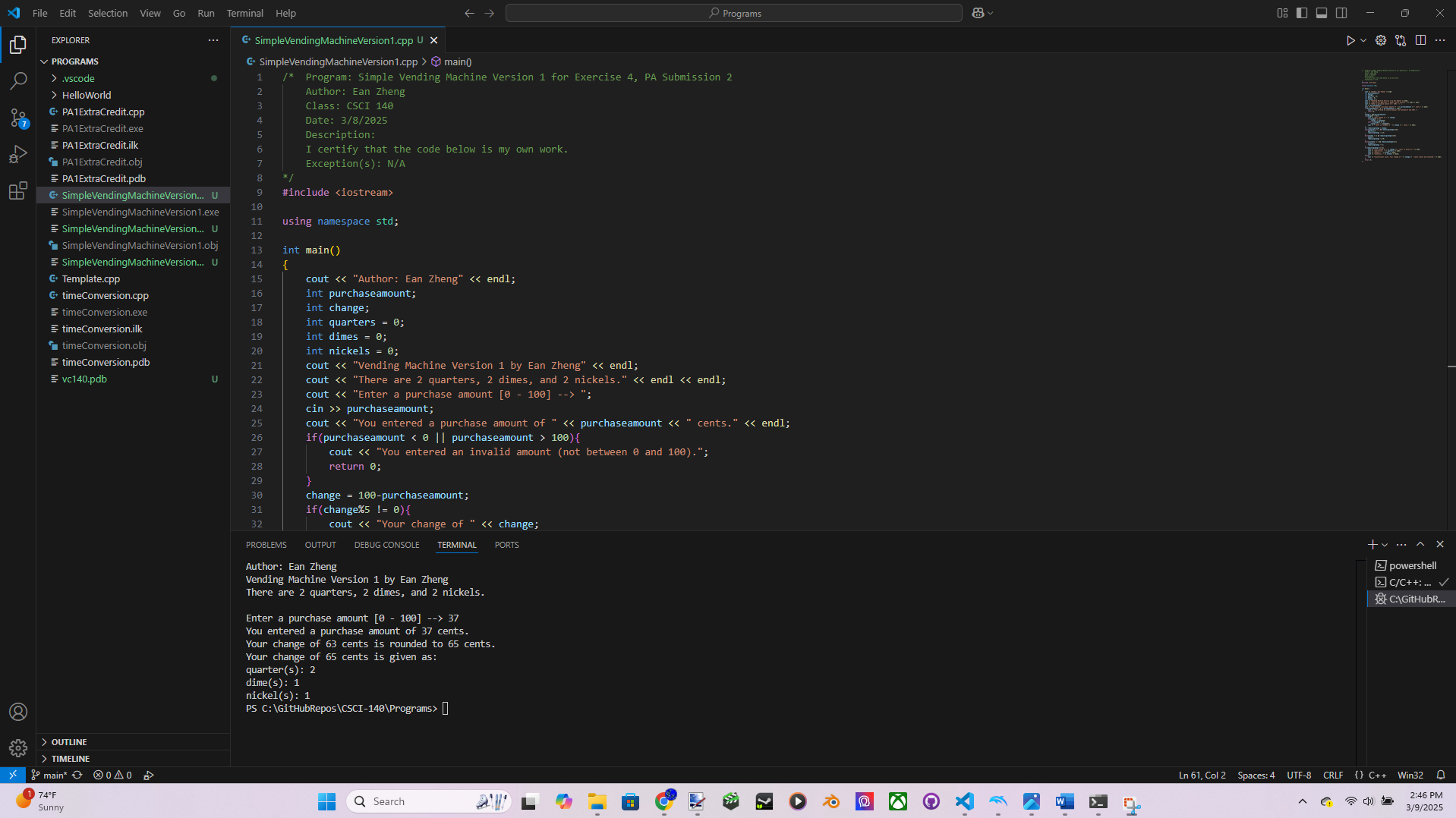
Your change of 63 cents is rounded to 65 cents.

Your change of 65 cents is given as:

quarter(s): 2

dime(s): 1

nickel(s): 1



Author: Ean Zheng

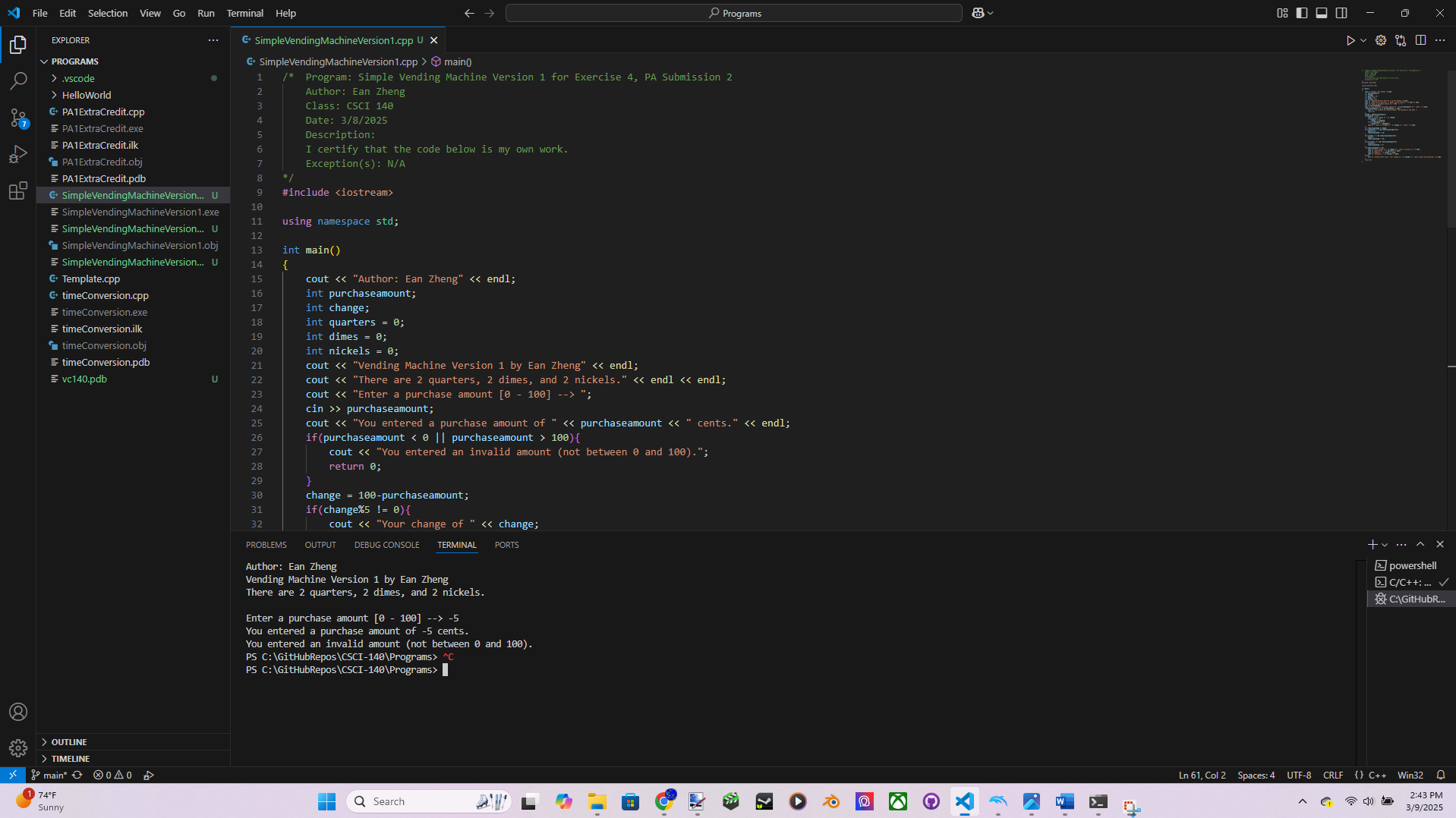
Vending Machine Version 1 by Ean Zheng

There are 2 quarters, 2 dimes, and 2 nickels.

Enter a purchase amount [0 - 100] --> -5

You entered a purchase amount of -5 cents.

You entered an invalid amount (not between 0 and 100).



Author: Ean Zheng

Vending Machine Version 1 by Ean Zheng

There are 2 quarters, 2 dimes, and 2 nickels.

Enter a purchase amount [0 - 100] --> 39

You entered a purchase amount of 39 cents.

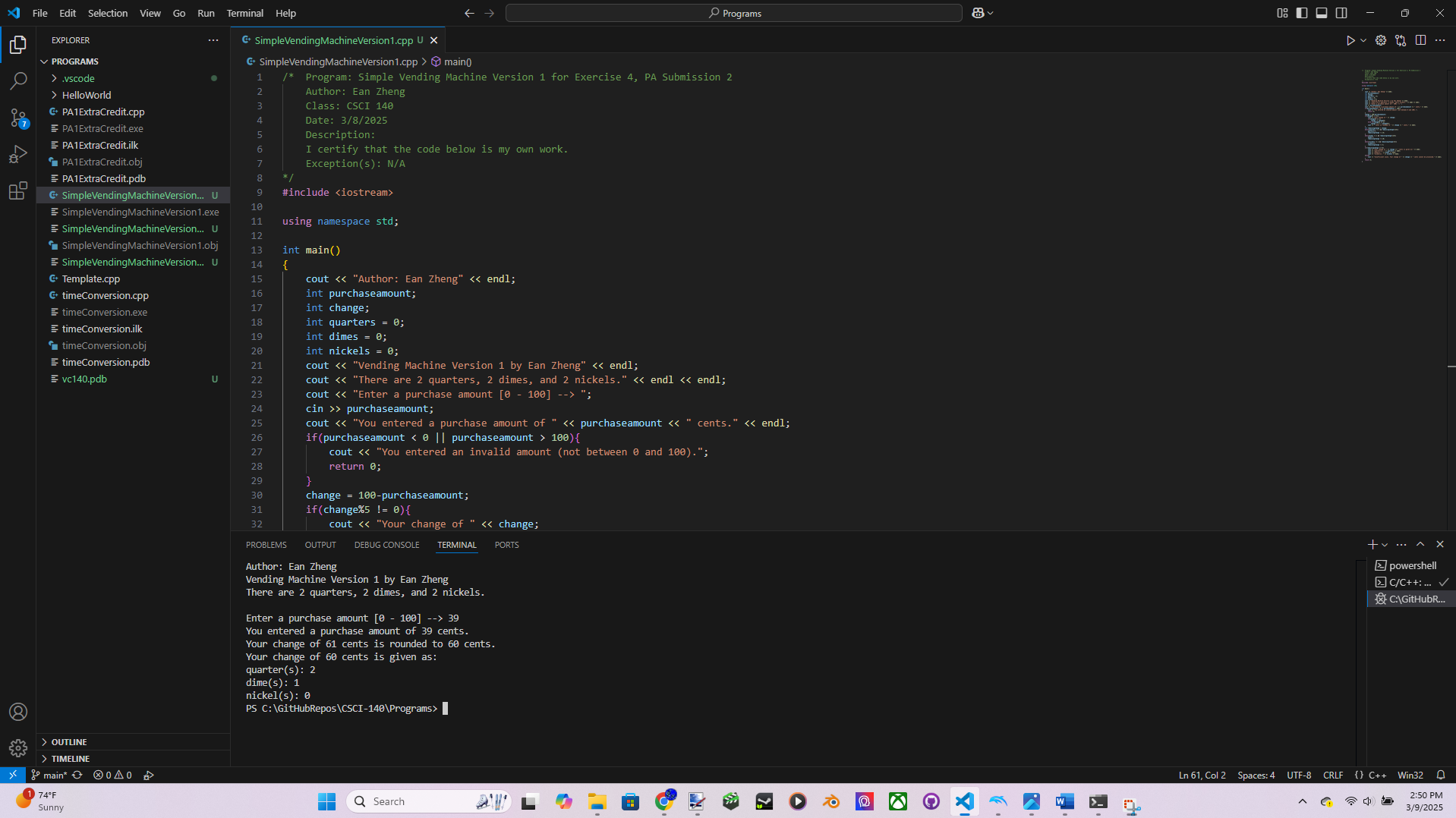
Your change of 61 cents is rounded to 60 cents.

Your change of 60 cents is given as:

quarter(s): 2

dime(s): 1

nickel(s): 0



Author: Ean Zheng

Vending Machine Version 1 by Ean Zheng

There are 2 quarters, 2 dimes, and 2 nickels.

Enter a purchase amount [0 - 100] --> 25

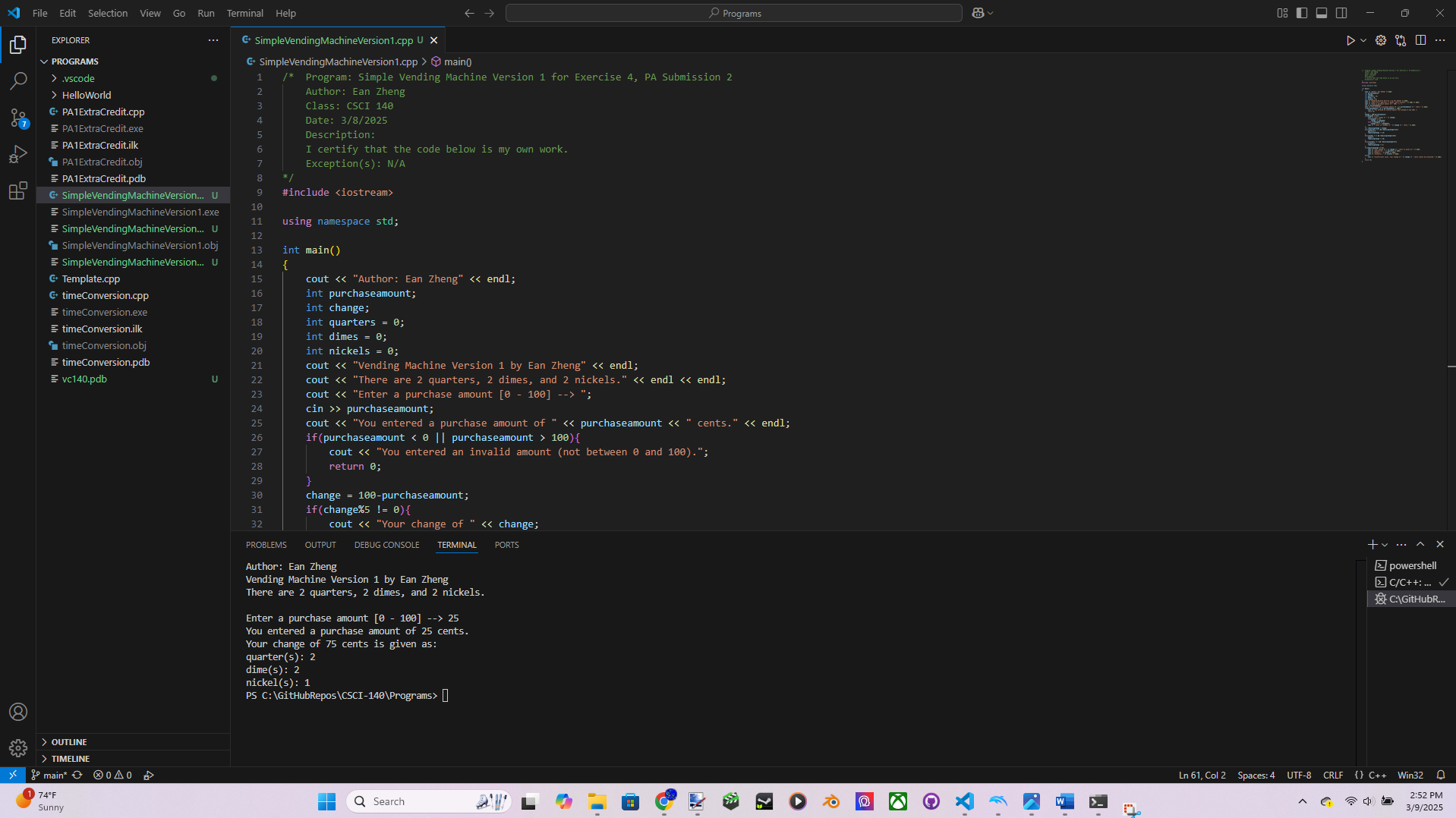
You entered a purchase amount of 25 cents.

Your change of 75 cents is given as:

quarter(s): 2

dime(s): 2

nickel(s): 1



Author: Ean Zheng

Vending Machine Version 1 by Ean Zheng

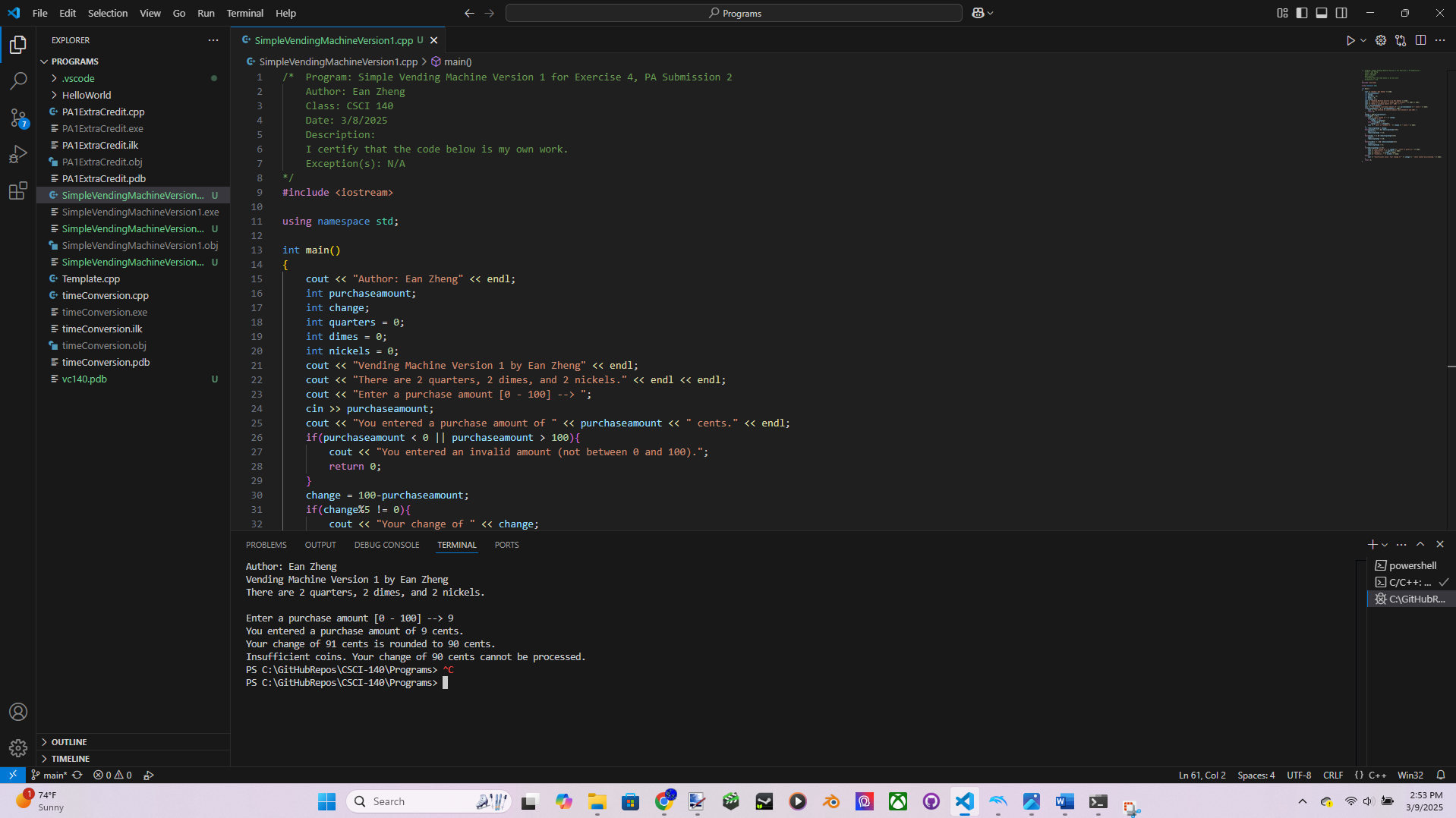
There are 2 quarters, 2 dimes, and 2 nickels.

Enter a purchase amount [0 - 100] --> 9

You entered a purchase amount of 9 cents.

Your change of 91 cents is rounded to 90 cents.

Insufficient coins. Your change of 90 cents cannot be processed.



Question 1: What is the main difference between an expression and a condition? Can  
you use an expression when a condition is required in C++ (e.g., use a + b as a  
condition)? Why or why not?

An expression is a combination of values/literals with operators/function calls that is also a process these values go through. The expression takes all the values, subjects them to its process, delivers a final result value, and assigns it to wherever it’s at. A condition, on the other hand, is a comparison statement that evaluates to either true or false. You can use an expression that evaluates to true or false, because where a condition is required only takes true or false values, and anything that evaluates to true or false is good there. Although most expressions don’t evaluate to true or false and can’t be used where a condition is, there are some that do. As such, they can be used where conditions are.

Question 2: It is easy to convert a switch statement to a nested if statement and it can be  
quite difficult to convert a nested if statement to a switch statement. Provide an example  
or situation to show that it can be quite difficult to convert a nested if statement to a  
switch statement.

A nested if statement can have two conditions that must both be fulfilled. For example, a nested if statement is composed of an if statement outside another, and both of them have two different conditions that must be fulfilled. If someone tries to convert it into a switch statement, they can’t include both conditions in one switch case. If they try to put a switch case below another, the case below can be executed skipping the one above, as long as it’s true. Therefore, this will cause only one of two conditions to be required for the code in the nested if statement to execute.

Extra Credit (2 points): Text message expander V2; copy your working code from  
exercise 3 to your IDE and modify it so it would expand all abbreviations. For example:  
Input: IDK how that happened, TTYL. IDK and TTYL.  
Output: I don't know how that happened, talk to you later. I don't know  
and talk to you later.

Source Code:

//Modified by: Ean Zheng

#include <iostream>

#include <string>

using namespace std;

int main() {

string input;

cout << "Enter text:" << endl;

getline(cin, input);

cout << "You entered: " << input << endl;

while((input.find("BFF") != string::npos) || (input.find("IDK") != string::npos) ||

(input.find("JK") != string::npos) || (input.find("TMI") != string::npos) ||

(input.find("TTYL") != string::npos)){

if(input.find("BFF") != string::npos){

input.replace(input.find("BFF"), 3, "best friend forever");

}

if(input.find("IDK") != string::npos){

input.replace(input.find("IDK"), 3, "I don't know");

}

if(input.find("JK") != string::npos){

input.replace(input.find("JK"), 2, "just kidding");

}

if(input.find("TMI") != string::npos){

input.replace(input.find("TMI"), 3, "too much information");

}

if(input.find("TTYL") != string::npos){

input.replace(input.find("TTYL"), 4, "talk to you later");

}

}

cout << "Expanded: " << input << endl;

return 0;

}

Input/Output:

Enter text:

IDK how that happened, TTYL. IDK and TTYL.

You entered: IDK how that happened, TTYL. IDK and TTYL.

Expanded: I don't know how that happened, talk to you later. I don't know and talk to you later.

