

# CSCI 140 PA 3 Submission

Due Date: 3/17/2025 Late (date and time): \_\_\_\_\_

Name: Ean Zheng

## Exercise 1 – 4.18 LAB: Varied amount of input data

The screenshot displays the zyBooks lab environment for '4.18.1: LAB: Varied amount of input data'. The lab instructions state: 'Statistics are often calculated with varying amounts of input data. Write a program that takes any number of non-negative integers as input, and outputs the max and average. A negative integer ends the input and is not included in the statistics. Assume the input contains at least one non-negative integer. Output each floating-point value with two digits after the decimal point, which can be achieved by executing `cout << fixed << setprecision(2);` once before all other cout statements. Ex: When the input is: 15 20 0 3 -1 the output is: 20 9.50'.

The code editor shows the following C++ code in `main.cpp`:

```
1 //Modified by: Ean Zheng
2 #include <iostream>
3 #include <iomanip>
4 using namespace std;
5
6 int main() {
7
8     int input;
9     int max = 0;
10    int sum = 0;
11    int count = 0;
12    while(input >= 0){
13        cin >> input;
14        if(input >= 0){
15            if(input > max) max = input;
16            sum = sum + input;
17            count = count + 1;
18        }
19    }
20    cout << fixed << setprecision(2) << max << " " << static_cast<double>(sum)/count << endl;
21
22    return 0;
23 }
```

The console output shows: 'Your program produced no output.'

Below the console, the submission status is shown: 'Submit for grading', 'Coding trail of your work', '3/15 8:0 - 0,0,10 min:5', 'Latest submission - 9:30 PM PDT on 03/15/25', 'Submission passed all tests ✓', 'Total score: 10 / 10', and 'Open submission's code'.

## Exercise 2 – 4.22 LAB: Password modifier (only available in Classic mode)

The screenshot displays the zyBooks online IDE interface for the '4.22.1: LAB: Password modifier' exercise. The code in `main.cpp` is as follows:

```
1 //Modified by: Ean Zheng
2 #include <iostream>
3 #include <string>
4 using namespace std;
5
6 int main() {
7
8     string input;
9     getline(cin, input);
10    int index = 0;
11    while(index < input.length()){
12        if(input.at(index) == ' '){
13            input.at(index) = '1';
14        }else if(input.at(index) == 'a'){
15            input.at(index) = '8';
16        }else if(input.at(index) == 'm'){
17            input.at(index) = '9';
18        }else if(input.at(index) == '8'){
19            input.at(index) = '5';
20        }else if(input.at(index) == 's'){
21            input.at(index) = '4';
22        }else if(input.at(index) == 'L'){
23            input.at(index) = '7';
24        }
25        index = index + 1;
26    }
27    input.push_back('!');
28    cout << input << endl;
29    return 0;
30 }
```

The submission history shows a successful submission on 03/15/25 with a score of 10/10. The compiler warnings are as follows:

```
main.cpp: In function 'int main()':
main.cpp:11:16: warning: comparison of integer expressions of different signedness: 'int' and
11 | while(index < input.length()){
   |             ~~~~~^~~~~~
```

The input field shows the password: `password`.

## Exercise 3 – 4.29 LAB: Brute force equation solver

LAB ACTIVITY 4.29.1: LAB: Brute force equation solver 10/10

Numerous engineering and scientific applications require finding solutions to a set of equations. Ex:  $8x + 7y = 38$  and  $3x - 5y = -1$  have a solution  $x = 3, y = 2$ . Given integer coefficients ( $a, b, c, d, e$ , and  $f$ ) of two linear equations with variables  $x$  and  $y$  listed below, use brute force to find an integer solution for  $x$  and  $y$  in the range  $-10$  to  $10$ .

$$ax + by = c$$
$$dx + ey = f$$

Ex: If the input is:

```
8 7 38
3 -5 -1
```

the output is:

```
x = 3, y = 2
```

Use this brute force approach:

```
For every value of x from -10 to 10
  For every value of y from -10 to 10
    Check if the current x and y satisfy both equations. If so, output the solution, and
    finish.
Ex: If no solution is found, output:
There is no solution
```

Assume the two input equations have no more than one solution.

Note: Elegant mathematical techniques exist to solve such linear equations. However, for other kinds of equations or situations, brute force can be handy.

Open new tab | Dock

main.cpp

```
1 //Modified by: Ean Zheng
2 #include <iostream>
3 using namespace std;
4
5 int main() {
6     int a;
7     int b;
8     int c;
9     int d;
10    int e;
11    int f;
12    cin >> a >> b >> c;
13    cin >> d >> e >> f;
14    for(int x = -10; x < 11; x++){
15        for(int y = -10; y < 11; y++){
16            if((a*x)+(b*y) == c && (d*x)+(e*y) == f){
17                cout << "x = " << x << ", " << "y = " << y << endl;
18                return 0;
19            }
20        }
21    }
22    cout << "There is no solution" << endl;
23    return 0;
24 }
25
```

DESKTOP CONSOLE

```
8 7 38
3 -5 -1
```

Submit for grading

Coding trail of your work What is this?

PA 3 SubmissionzyBook Assignment 1Section 2.15 - CSCI 140 C++ LabMy activity - CSCI 140 C++ LabSection 4.29 - CSCI 140 C++ LabSection 4.3 - CSCI 140 C++ Lab

learn.zybooks.com/zybook/MTSACCSCI140/Spring2025/chapter/4/section/29

zyBooksMy library > CSCI 140: C++ Language and Object Development home > 4.29: LAB: Brute force equation solverzyBooks catalogHelp/FAQEan Zheng

```
9 int d;
10 int e;
11 int f;
12 cin >> a >> b >> c;
13 cin >> d >> e >> f;
14 for(int x = -10; x < 11; x++){
15     for(int y = -10; y < 11; y++){
16         if((a*x)+(b*y) == c && (d*x)+(e*y) == f){
17             cout << "x = " << x << ", " << "y = " << y << endl;
18             return 0;
19         }
20     }
21 }
22 cout << "There is no solution" << endl;
23 return 0;
24 }
25
```

DESKTOPCONSOLE

8 7 38
3 -5 -1

Submit for grading

Coding trail of your workWhat is this?

3/150,0 - 10 min:9

Latest submission - 10:08 PM PDT on 03/15/25Submission passed all tests ✓Total score: 10 / 10

☐ Only show failing testsOpen submission's code

1: Compare output ^3 / 3

Compare output

Input8 7 38
3 -5 -1

100%Clear

Search

9:30 PM3/15/2025

Exercise 4 – Simple Vending Machine Version 2 – more points for this exercise  
Modify “Simple Vending Machine Version 1” from previous PA to support the following new requirements:

- Fill machine with coins via keyboard input before any purchase (quarters, dimes, and nickels).
- Must allow purchases to be repeated until a sentinel value of 0 is entered.
- Keep track the number of dollar bills, coins in the machine and print them at the end.

Note: this new version might not work for all test cases, but that is not a problem (see one test under extra credit option). Pseudocode is not required since we modify it from the previous PA, but feel free to use it. Follow the interface below and you must try the following test case:

Vending Machine Version 2 by [Your Name]

Enter number of quarters, dimes, and nickels --> 2 4 4<Enter>

Number of quarters: 2

Number of dimes : 4

Number of nickels : 4

Machine balance : \$1.10

Only one-dollar bill will be accepted.

Only amount between 0 to 100 is accepted.

Enter 0 to stop the program.

Enter a purchase amount [0 - 100] --> 36<Enter>

You entered a purchase amount of 36 cents.

Inserting one-dollar bill.

Processing your purchase ...

Your change of 64 cents is rounded to 65 cents.

Your change of 65 cents is given as:

quarter(s): 2

dime(s): 1

nickel(s): 1

Enter a purchase amount [0 - 100] --> -1<Enter>

You entered a purchase amount of -1 cents.

Invalid amount (outside valid range). Try again.

Enter a purchase amount [0 - 100] --> 35<Enter>

You entered a purchase amount of 35 cents.

Please insert one-dollar bill.

Processing your purchase ...

Insufficient changes!

Your transaction cannot be processed.

Please take back your dollar bill.

Enter a purchase amount [0 - 100] --> 73<Enter>

You entered a purchase amount of 73 cents.

Inserting one-dollar bill.

Processing your purchase ...

Your change of 23 cents is rounded to 25 cents.

Your change of 25 cents is given as:

quarter(s): 0

dime(s): 2

nickel(s): 1

Enter a purchase amount [0 - 100] --> 105<Enter>

You entered a purchase amount of 105 cents.

Invalid amount (outside valid range). Try again.

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

Number of dollars: 2

Number of quarters: 0

Number of dimes : 1

Number of nickels : 2

Machine balance : \$2.20

Pseudocode below if applicable:

Input quarters, dimes, and nickels as variables with prompt.

Output coin values, then calculate machine balance and output it with this equation:  
 $(\text{quarters} * 0.25) + (\text{dimes} * 0.1) + (\text{nickels} * 0.05)$ . Set precision output to 2 decimal places

Add new prompts of specification

Add a do while loop around the enter purchase amount process with a condition of the input not equal to 0.

Add an else if statement for the if statement checking invalid range, with condition of input isn't 0, and put code for valid range there.

Redo process of detection: Add new variables in do while loop for given quarters, dimes, and nickels. Increment given coins for every time a coin is needed. Conditions for coins rewrite if given coin variables are not equal to normal coin variables. If processing successful, subtract given coin variables from normal coin variables, and output the given coin variables.

Update invalid amount message and add new messages for processing. Update change process fail message.

Add dollars variable initialized as 0. Increment it if change process is successful.

After loop is exited, output money left in machine, dollars, quarters, dimes, and nickels. Output machine balance to with same equation as before except with dollars added.

Source code below:

```
/* Program: Simple Vending Machine Version 2 for Exercise 4, PA Submission 3
   Author: Ean Zheng
   Class: CSCI 140
   Date: 3/16/2025
   Description:
   I certify that the code below is my own work.
   Exception(s): N/A
*/
#include <iostream>
#include <iomanip>

using namespace std;

int main()
{
    cout << "Author: Ean Zheng" << endl;
    int purchaseamount;
    int change;
    int quarters;
    int dimes;
    int nickels;
    int dollars = 0;
    cout << "Vending Machine Version 2 by Ean Zheng" << endl << endl;
    cout << "Enter number of quarters, dimes, and nickels -->";
    cin >> quarters >> dimes >> nickels;
    cout << "Number of quarters: " << quarters << endl;
    cout << "Number of dimes: " << dimes << endl;
    cout << "Number of nickels: " << nickels << endl;
    cout << "Machine balance: $" << fixed << setprecision(2)<<
    (quarters*0.25)+(dimes*0.1)+(nickels*0.05) << endl << endl;
    cout << "Only one-dollar bill will be accepted." << endl;
    cout << "Only amount between 0 to 100 is accepted." << endl;
    cout << "Enter 0 to stop the program." << endl;

    do{
        int givenquarters = 0;
        int givendimes = 0;
        int givennickels = 0;
        cout << "Enter a purchase amount [0 - 100] --> ";
        cin >> purchaseamount;
        if(purchaseamount < 0 || purchaseamount > 100){
            cout << "You entered a purchase amount of " << purchaseamount << " cents." <<
endl;
            cout << "Invalid amount (outside valid range). Try again." << endl << endl;
        }
    } while (purchaseamount < 0 || purchaseamount > 100);
}
```

```

    }
    else if (purchaseamount != 0){
        cout << "You entered a purchase amount of " << purchaseamount << " cents." <<
endl;
        cout << "Inserting one-dollar bill." << endl;
        cout << "Processing your purchase..." << endl;
        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(givenquarters != quarters && remainingchange>=25){
            givenquarters++;
            remainingchange -= 25;
        }
        while(givendimes != dimes && remainingchange>=10){
            givendimes++;
            remainingchange -= 10;
        }
        while(givennickels != nickels && remainingchange>=5){
            givennickels++;
            remainingchange -= 5;
        }
        if(remainingchange == 0){
            dollars++;
            quarters = quarters - givenquarters;
            dimes = dimes - givendimes;
            nickels = nickels - givennickels;
            cout << "Your change of " << change << " cents is given as:" << endl;
            cout << "\tquarter(s): " << givenquarters << endl;
            cout << "\tdime(s): " << givendimes << endl;
            cout << "\tnickel(s): " << givennickels << endl;
        }else{
            cout << "Insufficient changes!" << endl;
            cout << "Your transaction cannot be processed." << endl;
            cout << "Please take back your dollar bill." << endl;
        }
        cout << endl;
    }
}while(purchaseamount != 0);
cout << "Number of dollars: " << dollars << endl;

```



```

cout << "Number of quarters: " << quarters << endl;
cout << "Number of dimes: " << dimes << endl;
cout << "Number of nickels: " << nickels << endl;
cout << "Machine balance: $" << fixed << setprecision(2)<<
dollars+(quarters*0.25)+(dimes*0.1)+(nickels*0.05) << endl << endl;
return 0;
}

```

```

// Program: Simple Vending Machine Version 2 for Exercise 4, PA Submission 1
// Author: Fan Zheng
// Class: CSCI 140
// Date: 3/16/2025
// Description:
// I certify that the code below is my own work.
// Exception(s): N/A

#include <iostream>
#include <iomanip>

using namespace std;

int main()
{
    cout << "Author: Fan Zheng" << endl;
    int purchaseamount;
    int change;
    int quarters;
    int dimes;
    int nickels;
    int dollars = 0;
    cout << "Vending Machine Version 2 by Fan Zheng" << endl << endl;
    cout << "Enter number of quarters, dimes, and nickels --> ";
    cin >> quarters >> dimes >> nickels;
    cout << "Number of quarters: " << quarters << endl;
    cout << "Number of dimes: " << dimes << endl;
    cout << "Number of nickels: " << nickels << endl;
    cout << "Machine balance: $" << fixed << setprecision(2) << (quarters*0.25)+(dimes*0.1)+(nickels*0.05) << endl << endl;
    cout << "Only one-dollar bill will be accepted." << endl;
    cout << "Only amount between 0 to 100 is accepted." << endl;
    cout << "Enter 0 to stop the program." << endl;

    do{
        int givenquarters = 0;
        int givendimes = 0;
        int givennickels = 0;
        cout << "Enter a purchase amount [0 - 100] --> ";
        cin >> purchaseamount;
        if(purchaseamount < 0 || purchaseamount > 100){
            cout << "You entered a purchase amount of " << purchaseamount << " cents." << endl;
            cout << "Invalid amount (outside valid range). Try again." << endl << endl;
        }
        else if (purchaseamount != 0){
            cout << "You entered a purchase amount of " << purchaseamount << " cents." << endl;
            cout << "Inserting one-dollar bill." << endl;
            cout << "Processing your purchase..." << endl;
            change = 100 - purchaseamount;
        }
    } while (purchaseamount != 0);
    cout << "Number of dollars: " << dollars << endl;
}

```

```

int main()
{
    // ... (previous code) ...

    if(purchaseamount < 0 || purchaseamount > 100){
        cout << "Invalid amount (outside valid range). Try again." << endl << endl;
    }
    else if (purchaseamount != 0){
        cout << "You entered a purchase amount of " << purchaseamount << " cents." << endl;
        cout << "Inserting one-dollar bill." << endl;
        cout << "Processing your purchase..." << endl;
        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(givenquarters != quarters && remainingchange >= 25){
            givenquarters++;
            remainingchange -= 25;
        }
        while(givennickels != nickels && remainingchange >= 5){
            givennickels++;
            remainingchange -= 5;
        }
        if(remainingchange == 0){
            dollars += quarters;
            quarters = 0;
            dimes = dimes - givenquarters;
            nickels = nickels - givennickels;
            cout << "Your change of " << change << " cents is given as:" << endl;
            cout << "  Quarters: " << givenquarters << endl;
            cout << "  Dimes: " << givendimes << endl;
            cout << "  Nickels: " << givennickels << endl;
        }
        else{
            cout << "Insufficient change!" << endl;
            cout << "Your transaction cannot be processed." << endl;
            cout << "Please take back your dollar bill." << endl;
        }
    }
    cout << endl;
}

```

```
File Edit Selection View Go Run Terminal Help
SimpleVendingMachineVersion2.cpp U X PA1ExtraCredit.cpp
Programs > SimpleVendingMachineVersion2.cpp > main()
14 int main()
15 {
16     do{
17         else if (purchaseAmount != 0){
18             if(changeK5 != 0){
19                 cout << "Your change of " << change;
20                 if(changeK5 <= 3)
21                     change = changeK5;
22                 else if(changeK5 >= 3)
23                     change = 5 - (changeK5);
24                 cout << " cents is rounded to " << change << " cents." << endl;
25             }
26             int remainingChange = change;
27             while(givenQuarters != quarters && remainingChange >= 25){
28                 givenQuarters++;
29                 remainingChange -= 25;
30             }
31             while(givenDimes != dimes && remainingChange >= 10){
32                 givenDimes++;
33                 remainingChange -= 10;
34             }
35             while(givenNickels != nickels && remainingChange >= 5){
36                 givenNickels++;
37                 remainingChange -= 5;
38             }
39             if(remainingChange == 0){
40                 dollars++;
41                 quarters = quarters - givenQuarters;
42                 dimes = dimes - givenDimes;
43                 nickels = nickels - givenNickels;
44                 cout << "Your change of " << change << " cents is given as: " << endl;
45                 cout << "\tquarters: " << givenQuarters << endl;
46                 cout << "\tdimes: " << givenDimes << endl;
47                 cout << "\tnickels: " << givenNickels << endl;
48             }
49             else{
50                 cout << "Insufficient change!" << endl;
51                 cout << "Your transaction cannot be processed." << endl;
52                 cout << "Please take back your dollar bill." << endl;
53             }
54             cout << endl;
55         }
56         while(purchaseAmount != 0);
57         cout << "Number of dollars: " << dollars << endl;
58         cout << "Number of quarters: " << quarters << endl;
59         cout << "Number of dimes: " << dimes << endl;
60         cout << "Number of nickels: " << nickels << endl;
61         cout << "Machine balance: $" << fixed << setprecision(2) << dollars*(quarters*0.25)+(dimes*0.1)+(nickels*0.05) << endl << endl;
62         return 0;
63     }
64 }
```

Input/output below:

Author: Ean Zheng

Vending Machine Version 2 by Ean Zheng

Enter number of quarters, dimes, and nickels --> 2 4 4

Number of quarters: 2

Number of dimes: 4

Number of nickels: 4

Machine balance: \$1.10

Only one-dollar bill will be accepted.

Only amount between 0 to 100 is accepted.

Enter 0 to stop the program.

Enter a purchase amount [0 - 100] --> 36

You entered a purchase amount of 36 cents.

Inserting one-dollar bill.

Processing your purchase...

Your change of 64 cents is rounded to 65 cents.

Your change of 65 cents is given as:

    quarter(s): 2

    dime(s): 1

    nickel(s): 1

Enter a purchase amount [0 - 100] --> -1

You entered a purchase amount of -1 cents.

Invalid amount (outside valid range). Try again.

Enter a purchase amount [0 - 100] --> 35

You entered a purchase amount of 35 cents.

Inserting one-dollar bill.

Processing your purchase...

Insufficient changes!

Your transaction cannot be processed.

Please take back your dollar bill.

Enter a purchase amount [0 - 100] --> 73

You entered a purchase amount of 73 cents.

Inserting one-dollar bill.

Processing your purchase...

Your change of 27 cents is rounded to 25 cents.

Your change of 25 cents is given as:

    quarter(s): 0

    dime(s): 2

    nickel(s): 1

Enter a purchase amount [0 - 100] --> 105  
Enter a purchase amount [0 - 100] --> 105  
You entered a purchase amount of 105 cents.  
Invalid amount (outside valid range). Try again.

Enter a purchase amount [0 - 100] --> 0  
Number of dollars: 2  
Number of quarters: 0  
Number of dimes: 1  
Number of nickels: 2  
Machine balance: \$2.20

```
Author: Ean Zheng
Vending Machine Version 2 by Ean Zheng

Enter number of quarters, dimes, and nickels --> 2 4 4
Number of quarters: 2
Number of dimes: 4
Number of nickels: 4
Machine balance: $1.18

Only one-dollar bill will be accepted.
Only amount between 0 to 100 is accepted.
Enter 4 to stop the program.
Enter a purchase amount [0 - 100] --> 36
You entered a purchase amount of 36 cents.
Inserting one-dollar bill.
Processing your purchase...
Your change of 64 cents is rounded to 65 cents.
Your change of 65 cents is given as:
    quarter(s): 2
    dime(s): 1
    nickel(s): 1

Enter a purchase amount [0 - 100] --> -1
You entered a purchase amount of -1 cents.
Invalid amount (outside valid range). Try again.

Enter a purchase amount [0 - 100] --> 35
You entered a purchase amount of 35 cents.
Inserting one-dollar bill.
Processing your purchase...
Insufficient change!
Your transaction cannot be processed.
Please take back your dollar bill.

Enter a purchase amount [0 - 100] --> 73
You entered a purchase amount of 73 cents.
Inserting one-dollar bill.
Processing your purchase...
Your change of 27 cents is rounded to 25 cents.
Your change of 25 cents is given as:
    quarter(s): 0
    dime(s): 2
    nickel(s): 1

Enter a purchase amount [0 - 100] --> 105
You entered a purchase amount of 105 cents.
Invalid amount (outside valid range). Try again.

Enter a purchase amount [0 - 100] --> 0
Number of dollars: 2
Number of quarters: 0
Number of dimes: 1
Number of nickels: 2
Machine balance: $2.20
```

Question 1: A while loop and a for loop are equivalent, but we may prefer to use one type of loop to another type of loop depending on the situation. Discuss when it is best to use a for loop over a while loop.

When we have a set number of repetitions for that loop. For example, if we know we have to repeat this 4 times, a for loop can easily have that condition in a much simpler, compact, and faster way.

Question 2: You are testing a program with a while loop. Which of the following test cases can be safe to ignore (select one best option that you might not have to try)? Pick one and briefly explain why it might not be needed.

- a. Skip the loop entirely
- b. Run the loop once
- c. Run the loop a few times
- d. Run the loop forever

C: run the loop a few times. The only things necessary to know about the loop is what makes it iterate and what prevents it. One time is good enough to know what makes it iterate. Therefore, a few times isn't necessary when it can be solved with a 1 iteration case.

Extra Credit (2 points): The strategy with maximizing the coin with highest value first will work for most cases, but there are a few cases where it would not work (such as  $Q = 2$ ,  $D = 4$ ,  $N = 0$ , and you need to give back 55 cents). You do not have to handle such situation with the regular version, and it is okay reject such transaction as insufficient coins (try it out with current version to confirm it). Modify your program to handle the above case and it must be a general solution so it would work with similar situation (think of another situation). Try your new solution with the original test cases above (should work like before), the special test case below, and another special situation that would not work with the original version.

Vending Machine Version 2 EC by [Your Name]

Enter number of quarters, dimes, and nickels --> 2 4 0<Enter>

Number of quarters: 2

Number of dimes : 4

Number of nickels : 0

Machine balance : \$0.90

Only one-dollar bill will be accepted.

Only amount between 0 to 100 is accepted.

Enter 0 to stop the program.

Enter a purchase amount [0 - 100] --> 45<Enter>

You entered a purchase amount of 45 cents.

Inserting one-dollar bill.

Processing your purchase ...

Your change of 55 cents is given as:

quarter(s): 1

dime(s): 3

nickel(s): 0

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

Number of dollars: 1

Number of quarters: 1

Number of dimes : 1

Number of nickels : 0

Machine balance : \$1.35

Fill out and turn in the PA submission file with this lab (save as PDF format).

Source Code:

```
/* Program: Simple Vending Machine Version 2 for Extra Credit, PA Submission 3
   Author: Ean Zheng
   Class: CSCI 140
   Date: 3/16/2025
   Description:
   I certify that the code below is my own work.
   Exception(s): N/A
*/
#include <iostream>
#include <iomanip>

using namespace std;

int main()
{
    cout << "Author: Ean Zheng" << endl;
    int purchaseamount;
    int change;
    int quarters;
    int dimes;
    int nickels;
    int dollars = 0;
    cout << "Vending Machine Version 2 by Ean Zheng" << endl << endl;
    cout << "Enter number of quarters, dimes, and nickels -->";
    cin >> quarters >> dimes >> nickels;
    cout << "Number of quarters: " << quarters << endl;
    cout << "Number of dimes: " << dimes << endl;
    cout << "Number of nickels: " << nickels << endl;
    cout << "Machine balance: $" << fixed << setprecision(2)<<
(quarters*0.25)+(dimes*0.1)+(nickels*0.05) << endl << endl;
    cout << "Only one-dollar bill will be accepted." << endl;
    cout << "Only amount between 0 to 100 is accepted." << endl;
    cout << "Enter 0 to stop the program." << endl;

    do{
        int givenquarters = 0;
        int givendimes = 0;
        int givennickels = 0;
        cout << "Enter a purchase amount [0 - 100] --> ";
        cin >> purchaseamount;
        if(purchaseamount < 0 || purchaseamount > 100){
            cout << "You entered a purchase amount of " << purchaseamount << " cents." <<
endl;
            cout << "Invalid amount (outside valid range). Try again." << endl << endl;
        }
    } while (purchaseamount < 0 || purchaseamount > 100);
}
```

```

    }
    else if (purchaseamount != 0){
        cout << "You entered a purchase amount of " << purchaseamount << " cents." <<
endl;
        cout << "Inserting one-dollar bill." << endl;
        cout << "Processing your purchase..." << endl;
        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;
        }
        for(givenquarters = 0; (givenquarters <= quarters && givenquarters <=
4)&&((givenquarters*25)+(givendimes*10)+(givennickels*5)!=change);
givenquarters++){
            for(givendimes = 0; (givendimes <= dimes && givendimes <=
10)&&((givenquarters*25)+(givendimes*10)+(givennickels*5)!=change);
givendimes++){
                for(givennickels = 0; (givennickels <= nickels && givennickels <=
20)&&((givenquarters*25)+(givendimes*10)+(givennickels*5)!=change);
givennickels++){
                    if((givenquarters*25)+(givendimes*10)+(givennickels*5)==change)break;
                }
                if(givennickels > nickels || givennickels > 20)givennickels = 0;
                if((givenquarters*25)+(givendimes*10)+(givennickels*5)==change)break;
            }
            if(givendimes > dimes || givendimes > 10)givendimes = 0;
            if((givenquarters*25)+(givendimes*10)+(givennickels*5)==change)break;
        }
        if((givenquarters*25)+(givendimes*10)+(givennickels*5)==change){
            dollars++;
            quarters = quarters - givenquarters;
            dimes = dimes - givendimes;
            nickels = nickels - givennickels;
            cout << "Your change of " << change << " cents is given as:" << endl;
            cout << "\tquarter(s): " << givenquarters << endl;
            cout << "\tdime(s): " << givendimes << endl;
            cout << "\tnickel(s): " << givennickels << endl;
        }else{
            cout << "Insufficient changes!" << endl;
            cout << "Your transaction cannot be processed." << endl;
            cout << "Please take back your dollar bill." << endl;
        }
    }
}

```



```

        cout << endl;
    }
    while(purchaseamount != 0);
    cout << "Number of dollars: " << dollars << endl;
    cout << "Number of quarters: " << quarters << endl;
    cout << "Number of dimes: " << dimes << endl;
    cout << "Number of nickels: " << nickels << endl;
    cout << "Machine balance: $" << fixed << setprecision(2)<<
dollars+(quarters*0.25)+(dimes*0.1)+(nickels*0.05) << endl << endl;
    return 0;
}

```

### Output/Input:

Author: Ean Zheng  
Vending Machine Version 2 by Ean Zheng

Enter number of quarters, dimes, and nickels --> 2 4 0  
Number of quarters: 2  
Number of dimes: 4  
Number of nickels: 0  
Machine balance: \$0.90

Only one-dollar bill will be accepted.  
Only amount between 0 to 100 is accepted.  
Enter 0 to stop the program.  
Enter a purchase amount [0 - 100] --> 45  
You entered a purchase amount of 45 cents.  
Inserting one-dollar bill.  
Processing your purchase...  
Your change of 55 cents is given as:  
    quarter(s): 1  
    dime(s): 3  
    nickel(s): 0

Enter a purchase amount [0 - 100] --> 0  
Number of dollars: 1  
Number of quarters: 1  
Number of dimes: 1  
Number of nickels: 0  
Machine balance: \$1.35