

# CSCI 140 PA 1 Submission

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

Name: Ean Zheng

## Exercise 1 – zyBook 1.13 zyLab training: Basics:

The screenshot displays the zyBook interface for the '1.13 C++ example: Salary Calculation' section. The sidebar on the left contains a table of contents with the following items:

- 1) Introduction to C++
  - 1.1 Programming (general)
  - 1.2 Programming basics
  - 1.3 Console input
  - 1.4 Comments and whitespace
  - 1.5 Errors and warnings
  - 1.6 Computers and programs (general)
  - 1.7 IDE
  - 1.8 Computer tour
  - 1.9 Language history
  - 1.10 Problem solving
  - 1.11 Why programming
  - 1.12 Why whitespace matters
- 1.13 C++ example: Salary Calculation
- 1.14 C++ example: Married-couple names
- 1.15 zyLab training: Basics
- 1.16 zyLab training: Interleaved input / output
- 1.17 LAB: Formatted output: Hello World
- 1.18 LAB: Formatted output: No parking sign
- 1.19 LAB: Input and formatted output: House real estate summary
- 1.20 LAB: Input and formatted output: Right-facing arrow
- 1.21 LAB: Warm up: Hello world
- 1.22 LAB: Warm up: Basic outputs with variables

The main content area shows the title '1.13 C++ example: Salary Calculation' and a brief introduction. It includes a note about the material's purpose and a specific instruction for the exercise:

zyDE 1.13.1: Modify salary calculation.

The following program calculates yearly and monthly salary given an hourly wage. The program assumes a work-hours-per-week of 40 and work-weeks-per-year of 50.

1. Insert the correct number in the code below to print a monthly salary. Then run the program.

The code editor shows the following C++ code:

```
1 //Modified by: Ean Zheng
2 #include <iostream>
3 using namespace std;
4
5 int main () {
6     int hourlyWage;
7
8     hourlyWage = 20;
9
10    cout << "Annual salary is: ";
11    cout << hourlyWage * 40 * 50;
12    cout << endl;
13
14    cout << "Monthly salary is: ";
15    cout << ((hourlyWage * 40 * 50) / 12);
16    cout << endl;
17    // FIXME: The above is wrong. Change the 1 so the statement outputs monthly salary.
18
19    return 0;
20 }
```

The output section shows the results of the program execution:

```
Annual salary is: 40000
Monthly salary is: 3333
```

## Exercise 1 – zyBook 1.15 zyLab training: Basics:

The image displays two screenshots of the zyBook 1.15 zyLab training interface, showing the C++ code and the test results for the exercise.

**Top Screenshot:** The interface shows the C++ code in the `main.cpp` file. The code is as follows:

```
1 //Modified by: Ean Zheng
2 #include <iostream>
3 using namespace std;
4
5 int main() {
6     int userNum;
7     int userNumSquared;
8
9     cin >> userNum;
10
11     userNumSquared = userNum * userNum; // Bug here; fix it when instructed
12
13     cout << userNumSquared<<endl; // Output formatting issue here; fix it when instructed
14
15     return 0;
16 }
17
```

Below the code, the "Submit for grading" button is visible. The test results show a "Latest submission - 3:50 PM PST on 02/25/25" with a "Submission passed all tests" status and a "Total score: 3 / 3".

**Bottom Screenshot:** This screenshot shows the test results for the submission. The "Latest submission - 3:50 PM PST on 02/25/25" is shown with a "Submission passed all tests" status and a "Total score: 3 / 3". The test results are as follows:

- 1. Compare output: 1 / 1. Input: 2, Your output: 4.
- 2. Compare output: 1 / 1. Input: 12, Your output: 144.
- 3. Compare output: 1 / 1. Input: -5, Your output: 25.

## Exercise 2 – zyBook 1.21 LAB\*: Program: ASCII art:

1.21 LAB: Warm up: Hello world

**LAB ACTIVITY** 1.21.1: LAB: Warm up: Hello world Full screen 3/3

This zyLab activity prepares a student for a full programming assignment. Warm up exercises are typically simpler and worth fewer points than a full programming assignment, and are well-suited for an in-person scheduled lab meeting or as self practice.

For each of the following steps, end the program's output with a newline.

**Step 1 (1 pt): Write a program that outputs the following:**

```
Hello world!
```

**Step 2 (1 pt): Update to output the following:**

```
Hello world!
How are you?
```

**Step 3 (1 pt): Finally, update to output the following:**

```
Hello world!
How are you?
(I'm fine).
```

[Open new tab](#) [Dock](#)

[Run](#) [History](#) [Tutorial](#)

```
main.cpp
1  /* Program: Hello World for 1.2.1
2  Author: Ean Zheng
3  Class: CSCI 140
4  Date: 2/26/2025
5  Description:
```

**Step 3 (1 pt): Finally, update to output the following:**

```
Hello world!
How are you?
(I'm fine).
```

[Open new tab](#) [Dock](#)

[Run](#) [History](#) [Tutorial](#)

```
main.cpp
1  /* Program: Hello World for 1.2.1
2  Author: Ean Zheng
3  Class: CSCI 140
4  Date: 2/26/2025
5  Description:
6  I certify that the code below is my own work.
7  Exception(s): N/A
8  */
9
10 #include <iostream>
11 using namespace std;
12
13 int main()
14 {
15     cout<<"Hello world!"<<endl;
16     cout<<"How are you?"<<endl;
17     cout<<" (I'm fine)."<<endl;
18     return 0;
19 }
```

DESKTOP CONSOLE

PA 1 Submission (due end of ...)

Section 1.21 - CSCI 140 C++ L...

ReadSpeaker® docReader™

zyBooks Assignment 1

Section 1.8 - CSCI 140 C++ L...

learn.zybooks.com/zybook/MTSACCSCI140VoSpring2025/chapter/1/section/21

zyBooks My library > CSCI 140 C++ Language and Object Development home > 1.21: LAB: Warm up: Hello world

zyBooks catalog Help/FAQ Ean Zheng

```
15 cout<<"Hello world!"<<endl;
16 cout<<"How are you?"<<endl;
17 cout<<" (I'm fine)."<<endl;
18 return 0;
19 }
```

DESKTOP CONSOLE

Submit for grading

Coding trail of your work [What is this?](#)

2/26 W - 0, 1 - 3 min: 8

Latest submission - 11:27 AM PST on 02/26/25 Submission passed all tests ✓ Total score: 3 / 3

☐ Only show failing tests [Open submission's code](#)

1: Compare output ^ 1 / 1

Compare output

Your Output

Expected output starts with

1. Hello world!	1. Hello world!
2. How are you?	2.
3. (I'm fine).	
4.	

## Exercise 3 – zyBook 1.24 zyLab training: One large program:

**1.24 zyLab training\*: One large program**

**LAB ACTIVITY** 1.24.1: zyLab training\* One large program Full screen 10 / 10

Most zyLabs are designed to be completed in 20 - 25 minutes and emphasize a single concept. However, some zyLabs (such as the one large program or OLP) are more comprehensive and may take longer to complete.

**Incremental development** is a good programming practice and is the process of writing, compiling, and testing a small amount of code, then repeating the process with a small amount more (an incremental amount), and so on.

Suggested process to complete longer zyLabs:

- Implement Step 1 and submit for grading. Only one test will pass.
- Continue to implement one step at a time and resubmit for grading. At least one additional test should pass after each step.
- Continue until all steps are completed and all tests pass.

**Program Specifications** For practice with incremental development, write a program to output three statements as specified.

**Step 1 (4 pts).** Use cout to output "Step 1 complete". Submit for grading to confirm one of three tests passes.

Output should be:

```
Step 1 complete
```

**Step 2 (3 pts).** Use cout to output "Step 2 as well". Submit for grading to confirm two of three tests pass.

Output should be:

```
Step 1 complete
Step 2 as well
```

**Step 3 (3 pts).** Use cout to output "All steps now complete". Submit for grading to confirm all tests pass.

Output should be:

```
Step 1 complete
Step 2 as well
All steps now complete
```

[Open new tab](#) [Dock](#)

**Step 3 (3 pts).** Use cout to output "All steps now complete". Submit for grading to confirm all tests pass.

Output should be:

```
Step 1 complete
Step 2 as well
All steps now complete
```

[Open new tab](#) [Dock](#)

**main.cpp**

```
1  // Program: Large Program for 1.24
2  Author: Ean Zheng
3  Class: CSCI 140
4  Date:
5  Description:
6  I certify that the code below is my own work.
7  Exception(s): N/A
8  */
9
10 #include <iostream>
11 using namespace std;
12
13 int main()
14 {
15     cout << "Step 1 complete" << endl;
16     cout << "Step 2 as well" << endl;
17     cout << "All steps now complete" << endl;
18     return 0;
19 }
20
```

**DESKTOP CONSOLE**

```
Author: Ean Zheng
Step 1 complete
Step 2 as well
All steps now complete
→
```

all PA 1 Submission (due end of ...)

My subscription - CSCI 140 C++

Section 1.24 - CSCI 140 C++ L...

ReadSpeaker® docReader™

all zyBook Assignment 1

Section 1.6 - CSCI 140 C++ L...

learn.zybooks.com/zybook/MTSACCSCI140/NoSpring2025/chapter/1/section/24

zyBooks catalog

Help/FAQ

Ean Zheng

zyBooks My library > CSCI 140 C++ Language and Object Development home > 1.24: zyLab training\*. One large program

```
13 int main()
14 {
15     cout << "Step 1 complete" << endl;
16     cout << "Step 2 as well" << endl;
17     cout << "All steps now complete" << endl;
18     return 0;
19 }
20
```

DESKTOP CONSOLE

Author: Ean Zheng  
Step 1 complete  
Step 2 as well  
All steps now complete  
→

Submit for grading

Coding trail of your work [What is this?](#)

3/1 S = 0.10 min:1

Latest submission - 10:18 PM PST on 03/01/25

Submission passed all tests ✓ Total score: 10 / 10

☐ Only show failing tests

[Open submission's code](#)

1. Compare output 4 / 4

Compare output

Your Output

Expected output starts with

1. Step 1 complete	1. Step 1 complete
2. Step 2 as well	
3. All steps now complete	
4. ↵	

## Exercise 4 – zyBook 2.27 LAB: Driving costs:

zyBooks My library > CSCI 140: C++ Language and Object Development home > 2.27: LAB: Driving costs

zyBooks catalog Help/FAQ Ean Zheng

LAB ACTIVITY 2.27.1: LAB: Driving costs Full screen 10 / 10

Driving is expensive. Write a program with a car's gas mileage (miles/gallon) and the cost of gas (dollars/gallon) as floating-point input, and output the gas cost for 20 miles, 75 miles, and 500 miles.

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. Note: End with a newline.

Ex: If the input is:

```
25.0 3.1599
```

where the gas mileage is 25.0 miles/gallon and the cost of gas is \$3.1599/gallon, the output is:

```
2.53 9.48 63.20
```

Note: Real per-mile cost would also include maintenance and depreciation.

Run Open new tab Dock

main.cpp

```
1 //Modified by: Ean Zheng
2 #include <iostream>
3 #include <iomanip> //For setprecision
4 using namespace std;
5
6 int main() {
7     float gasmileage;
8     float gascost;
9     cin >> gasmileage;
10    cout << fixed << setprecision(2); // All later cout's will print floating-point values to exactly 2 dec
11                                     // Ex: 3.60
12    // Type your cout code here
13    cout << gascost * (20/gasmileage) << " " << gascost * (75/gasmileage) << " " << gascost * (500/gasmileage) << endl;
14    return 0;
15 }
16
17
```

DESKTOP CONSOLE

zyBooks My library > CSCI 140: C++ Language and Object Development home > 2.27: LAB: Driving costs

zyBooks catalog Help/FAQ Ean Zheng

main.cpp

```
1 //Zheng
2 #include <iostream>
3 #include <iomanip> //For setprecision
4 using namespace std;
5
6 int main() {
7     float gasmileage;
8     float gascost;
9     cin >> gasmileage;
10    cout << fixed << setprecision(2); // All later cout's will print floating-point values to exactly 2 decimal places.
11                                     // Ex: 3.60
12    // Type your cout code here
13    cout << gascost * (20/gasmileage) << " " << gascost * (75/gasmileage) << " " << gascost * (500/gasmileage) << endl;
14    return 0;
15 }
16
17
```

DESKTOP CONSOLE

```
25.0 3.1599
2.53 9.48 63.20
→
```

Submit for grading

Coding trail of your work [What is this?](#)

3/1 0 - 0 -- 10 min:9

Latest submission - 10:43 PM PST on 03/01/25 Submission passed all tests ✓ Total score: 10 / 10

☐ Only show failing tests [Open submission's code](#)

## Exercise 5 – zyBook 2.29 LAB: Using math functions:

LAB ACTIVITY 2.29.1: LAB: Using math functions 10 / 10

Given three floating-point numbers  $x$ ,  $y$ , and  $z$ , output  $x$  to the power of  $z$ ,  $x$  to the power of  $(y$  to the power of  $z)$ , the absolute value of  $y$ , and the square root of  $(xy$  to the power of  $z)$ .

Ex: If the input is:

```
5.0 6.5 3.2
```

the output is:

```
172.466 1.29951e+279 6.5 262.43
```

[Open new tab](#) [Dock](#)

[Run](#) [History](#) [Tutorial](#)

```
main.cpp
1 //Modified by: Ean Zheng
2 #include <iostream>
3 #include <cmath>
4 using namespace std;
5
6 int main() {
7     double x;
8     double y;
9     double z;
10
11     cin >> x;
12     cin >> y;
13     cin >> z;
14     /* Type your code here. Note: Include the math library above first. */
15     double a = pow(y, z);
16     cout << pow(x, z) << " " << pow(x, a) << " " << abs(y) << " " << sqrt(pow(x*y, z)) << endl;
17     return 0;
18 }
19
```

DESKTOP CONSOLE

```
5.0 6.5 3.2
172.466 1.29951e+279 6.5 262.43
```

[Submit for grading](#)

Coding trail of your work [What is this?](#)

3/1 [-----](#) 0,10 min:13

Latest submission - 11:12 PM PST on 03/01/25 Submission passed all tests ✓ Total score: 10 / 10

☐ Only show failing tests [Open submissions code](#)

1: Compare output ^ 5 / 5

Compare output

Input 5.0 6.5 3.2



## Exercise 6 – zyBook 2.36 LAB\*: Program: Painting a wall:

The screenshot shows the zyBooks IDE interface. The top navigation bar includes the zyBooks logo and the path: My library > CSCI 140 C++ Language and Object Development home > 2.36 LAB\*: Program: Painting a wall. The main editor displays the C++ code for the program. The code includes headers for `iostream`, `iomanip`, and `cmath`, and uses the `std` namespace. The `main` function takes height `h` and width `w` as input, calculates the wall area, paint needed, and total cost, and outputs the results with appropriate formatting. The output window shows the program's execution results for input values 8.0 and 64.0.

```
1 //Modified by: Ean Zheng
2 #include <iostream>
3 #include <iomanip> // needed for setprecision() and fixed
4 #include <cmath> // needed for ceil()
5 using namespace std;
6
7 int main() {
8     float h;
9     float w;
10    double cancost;
11    cin >> h;
12    cin >> w;
13    cin >> cancost;
14    float a = h*w;
15    cout << fixed << setprecision(1) << "Wall area: " << a << " sq ft" << endl;
16    cout << fixed << setprecision(3) << "Paint needed: " << a/350 << " gallons" << endl;
17    int cans = ceil(a/350);
18    cout << fixed << setprecision(0) << "Cans needed: " << cans << " can(s)" << endl;
19    float paintcost = cancost * cans;
20    cout << fixed << setprecision(2) << "Paint cost: $" << paintcost << endl;
21    cout << "Sales tax: $" << paintcost * 0.07 << endl;
22    cout << "Total cost: $" << paintcost * 1.07 << endl;
23    return 0;
24 }
```

DESKTOP CONSOLE

```
8.0 64.0 49.20
Wall area: 64.0 sq ft
Paint needed: 0.183 gallons
Cans needed: 1 can(s)
Paint cost: $49.20
Sales tax: $3.44
Total cost: $52.64
```

This screenshot shows the bottom portion of the zyBooks IDE interface. It includes a 'Submit for grading' button, a coding trail showing the user's progress (3/2), and submission details. The latest submission was made on 03/02/25 at 4:29 PM PST and passed all tests, resulting in a total score of 10/10. There is an option to 'Open submission's code'.

**Submit for grading**

Coding trail of your work [What is this?](#)

3/2 U-----10 min:15

Latest submission - 4:29 PM PST on 03/02/25      Submission passed all tests ✓      Total score: 10 / 10

☐ Only show failing tests      [Open submission's code](#)

### Exercise 7 – Writing Your First Program from Scratch

Step 1: Develop a design that leads to an algorithm that will read in a number that represents the number of total seconds. The output will be the equivalent number of hours, number of minutes, and leftover seconds. For examples, an input of 10000 seconds would be equivalent to 2 hours, 46 minutes, and 40 seconds. Provide pseudocode for this step.

Pseudocode below:

Variables: input, seconds, minutes, and hours, all int variables. Cin input variable to get input.

Divide input by 60 and floor it, store this as minutes variable, store mod of input by 60 as seconds variable

Divide minutes by 60 and floor it, store this as hours variable, update minutes variable as mod of minutes by 60

Print hours, minutes, and seconds with labels on them.

Step 2: Use your own IDE such as MS Visual Studio to develop your program. Call this program timeConversion.cpp. Translate pseudocode to C+ code. Compile the program. If you get compile errors, try to fix them, and recompile until your program is free of syntax errors.

Source code below:

```
/* Program: First Scratch Program for Exercise 7, PA Submission 1
```

```
Author: Ean Zheng
```

```
Class: CSCI 140
```

```
Date: 3/2/2025
```

```
Description:
```

```
I certify that the code below is my own work.
```

```
Exception(s): N/A
```

```
*/
```

```
#include <iostream>
```

```
#include <cmath>
```

```
using namespace std;
```

```
int main()
```

```
{
```

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

```
    int input;
```

```
    int seconds;
```

```
    int minutes;
```

```
    int hours;
```

```
    cin >> input;
```

```
    minutes = floor(input/60);
```

```
    seconds = input%60;
```

```
    hours = floor(minutes/60);
```

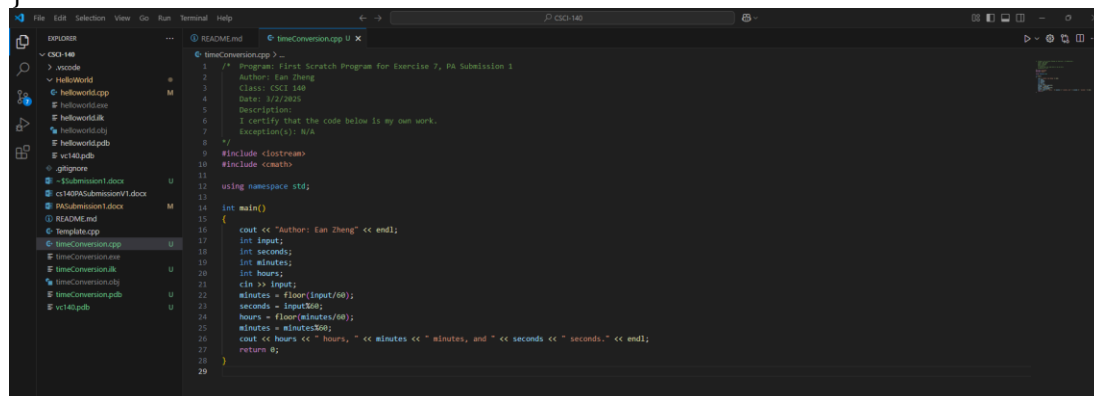
```
    minutes = minutes%60;
```

```
    cout << hours << " hours, " << minutes << " minutes, and " << seconds << " seconds."
```

```
<< endl;
```

```
    return 0;
```

```
}
```



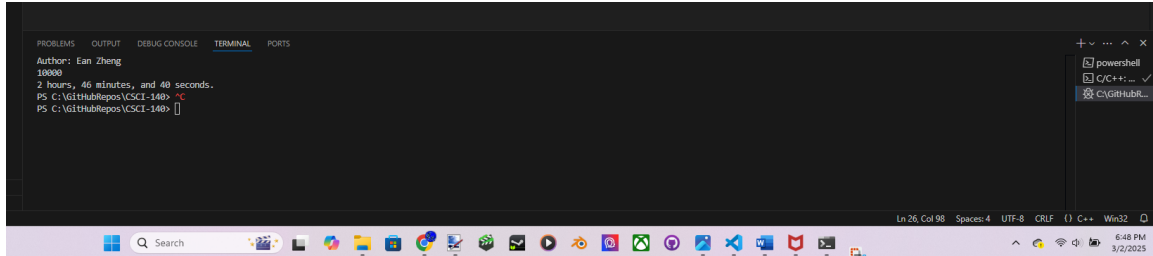
Step 3: Run the program with 10000 seconds. Is your output what you expect from the input you gave? If not, try to find and correct the logic error and run the program again. Continue this process until you have a program that produces the correct result. Try another value, 601 seconds, and confirm it produces the correct result as well.

Input/output below:

Author: Ean Zheng

10000

2 hours, 46 minutes, and 40 seconds.



The screenshot shows a Visual Studio Code terminal window with the following text:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
Author: Ean Zheng
10000
2 hours, 46 minutes, and 40 seconds.
PS C:\GithubRepos\CSCI-140>
PS C:\GithubRepos\CSCI-140> 
```

The terminal window is titled "TERMINAL" and shows the output of a program. The input is 10000, and the output is "2 hours, 46 minutes, and 40 seconds." The terminal is running in a PowerShell session. The status bar at the bottom indicates "Ln 26, Col 98", "Spaces: 4", "UTF-8", "CRLF", "C++", and "Win32". The system tray at the bottom right shows the time as "6:48 PM" and the date as "3/2/2025".

Question 1: Explain why source code is much more important than executable code for a SW developer.

Because the source code is the written down code that hasn't been compiled into executable code yet, and can still be edited or changed. Executable code is code compiled into 0 and 1s and can't be easily edited in the same way as source code, which is before it was compiled.

Question 2: How do you know for sure that your program is working correctly?

If there are no syntax, runtime, or compilation errors, and the output is correct for every single intended scenario.

Extra Credit (2 points): Develop a design that leads to an algorithm and a program that will read in a positive integer that represents a potential value of 1 less than power of 2. The output will be “yes” for a 1 less than power of 2 and “no” when it is not. Some examples of yes: 0, 7, 4095. Some examples of no: 5, 10, 5000. You must start with pseudocode and provide pseudocode for this exercise in addition to the program and its output. Note: do not use bit shift operators and you may need selection, repetition, and possibly a function for this exercise

Pseudocode below if applicable:

One int input variable that is later put in cin.

Input variable is added 1.

While loop for while input is not equal 1.:

    If  $\text{input} \bmod 2 = 0$ , divide input by 2

    Else if  $\text{input} \bmod 2 \neq 0$ , print no and return 0.

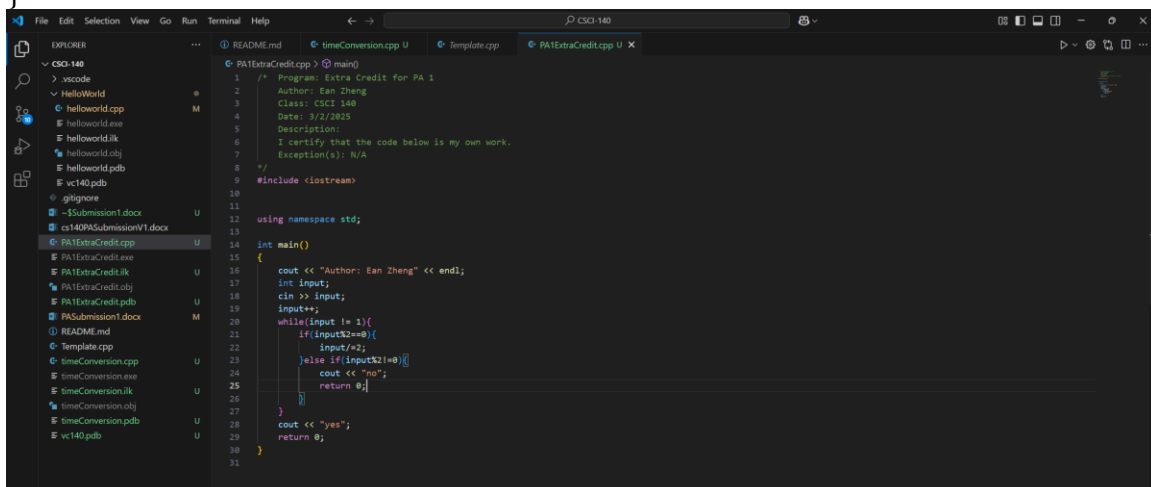
If while loop exits, print yes and return.

Source code below:

```
/* Program: Extra Credit for PA 1
   Author: Ean Zheng
   Class: CSCI 140
   Date: 3/2/2025
   Description:
   I certify that the code below is my own work.
   Exception(s): N/A
*/
#include <iostream>
```

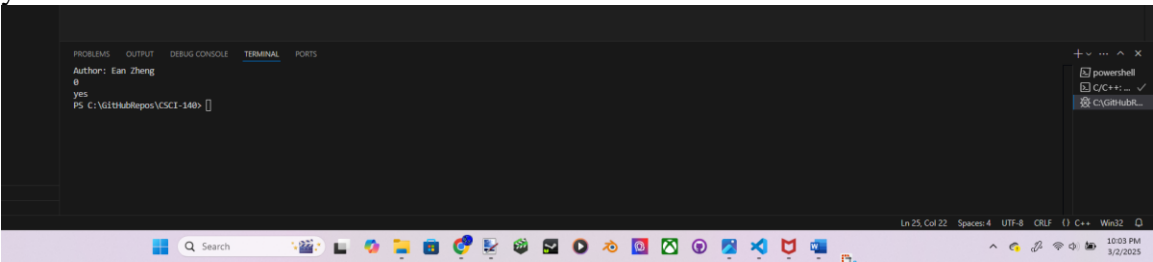
```
using namespace std;
```

```
int main()
{
    cout << "Author: Ean Zheng" << endl;
    int input;
    cin >> input;
    input++;
    while(input != 1){
        if(input%2==0){
            input/=2;
        }else if(input%2!=0){
            cout << "no";
            return 0;
        }
    }
    cout << "yes";
    return 0;
}
```

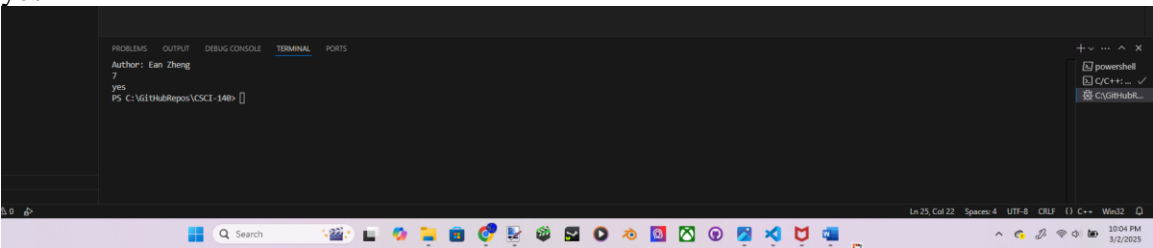


Input/output below:

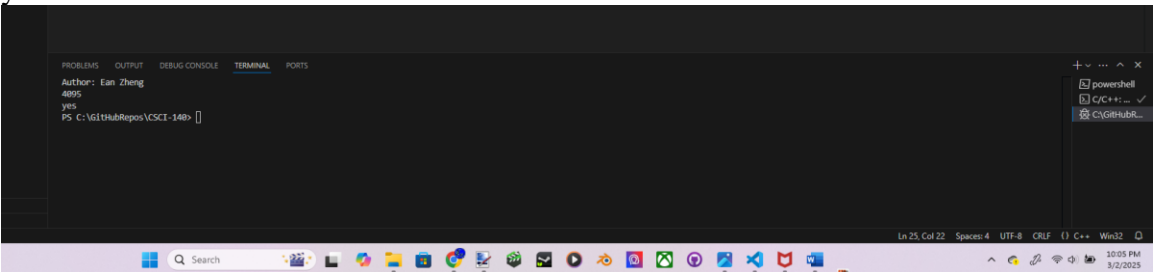
0  
yes



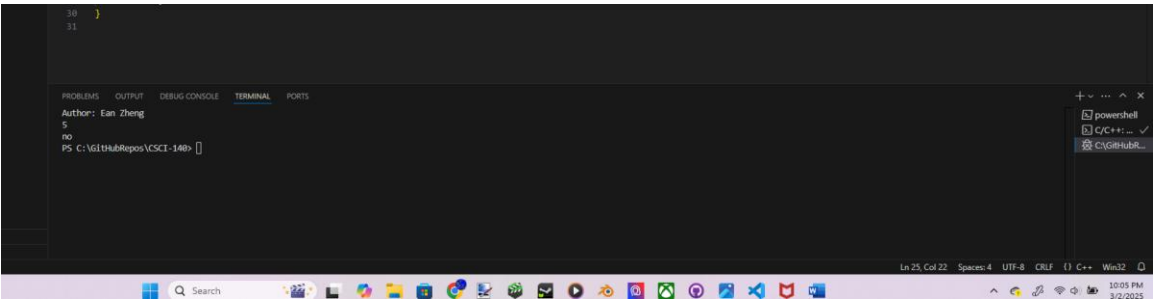
7  
yes



4095  
yes



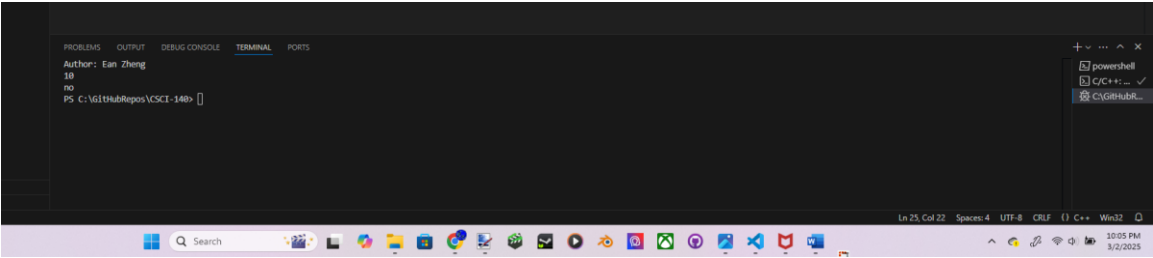
5  
no





10

no



5000

no

