Lab Assignment/Task Report

Only for course Teacher

		Needs Improvement	Developing	Sufficient	Above Average	Total Mark
Allocate mark & Percentage		**25%**	**50%**	**75%**	**100%**	**20**
Clarity						
Content Quality						
Spelling & Grammar						
Organization and Formatting						
					Total obtained mark	
Comments						

Semester: July-Dec 25(2nd)

Student Name: Ridone Orion

Student ID: 0022520005101027

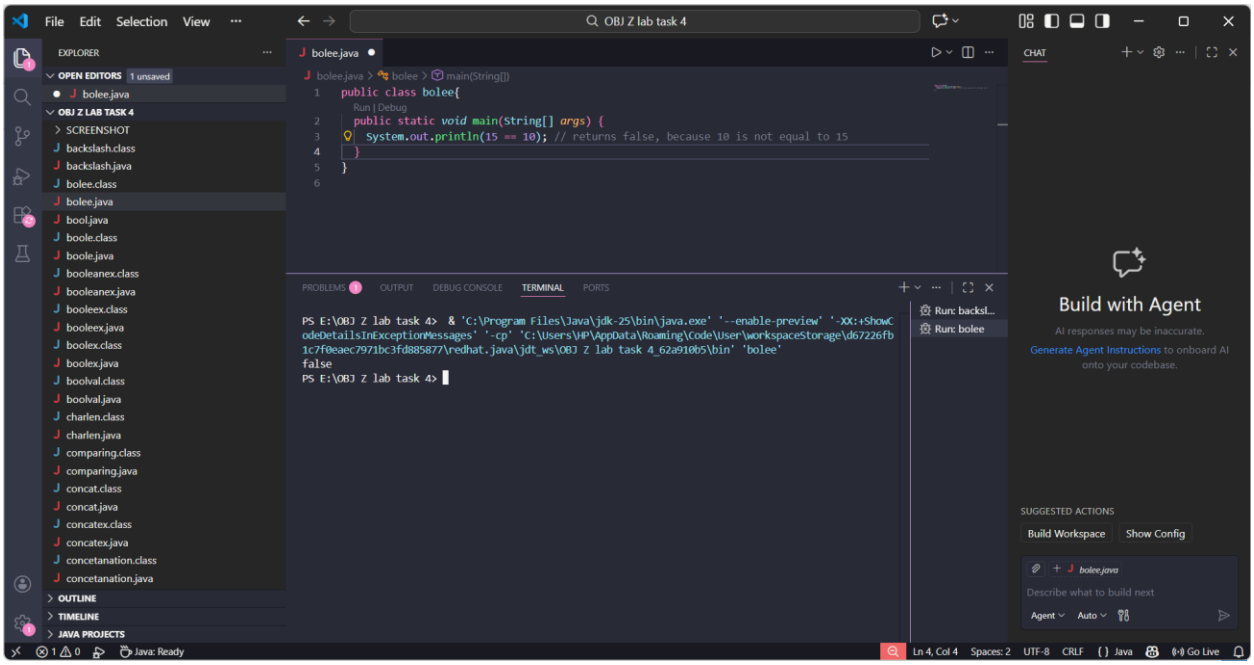
Batch: 45 **Section:** A

Course Code: CSE 124 **Course Name:** Object Oriented Programming Lab

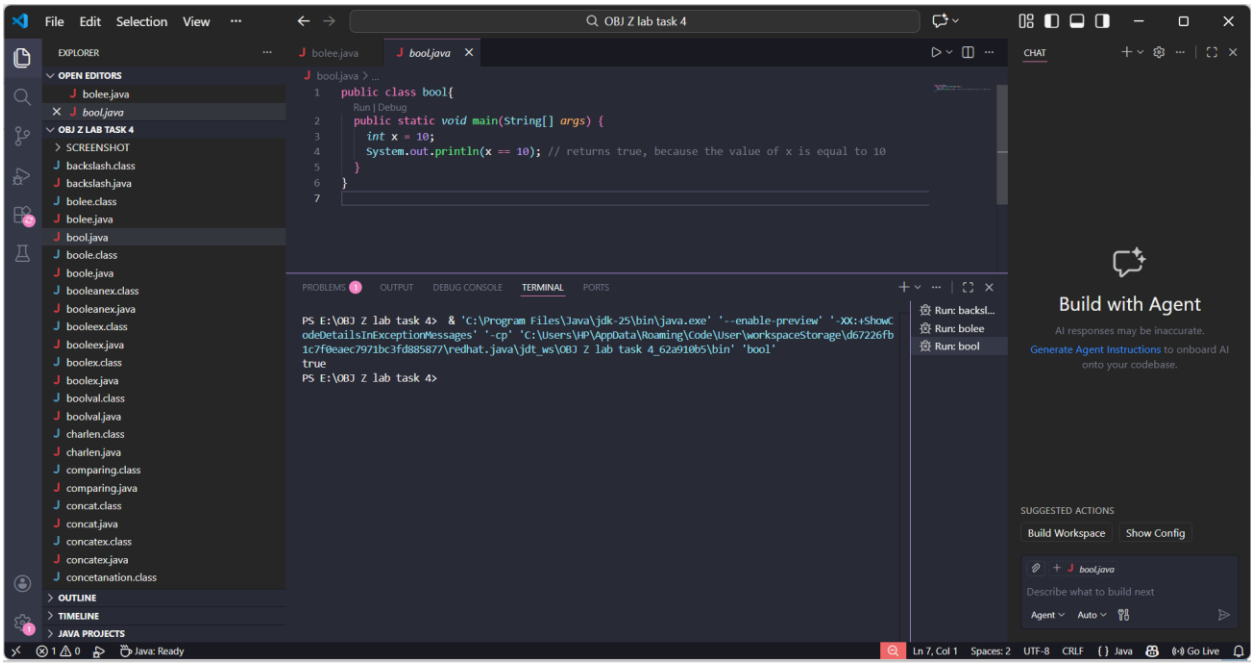
Course Teacher Name: Debabarata Mallick

Designation: Lecturer

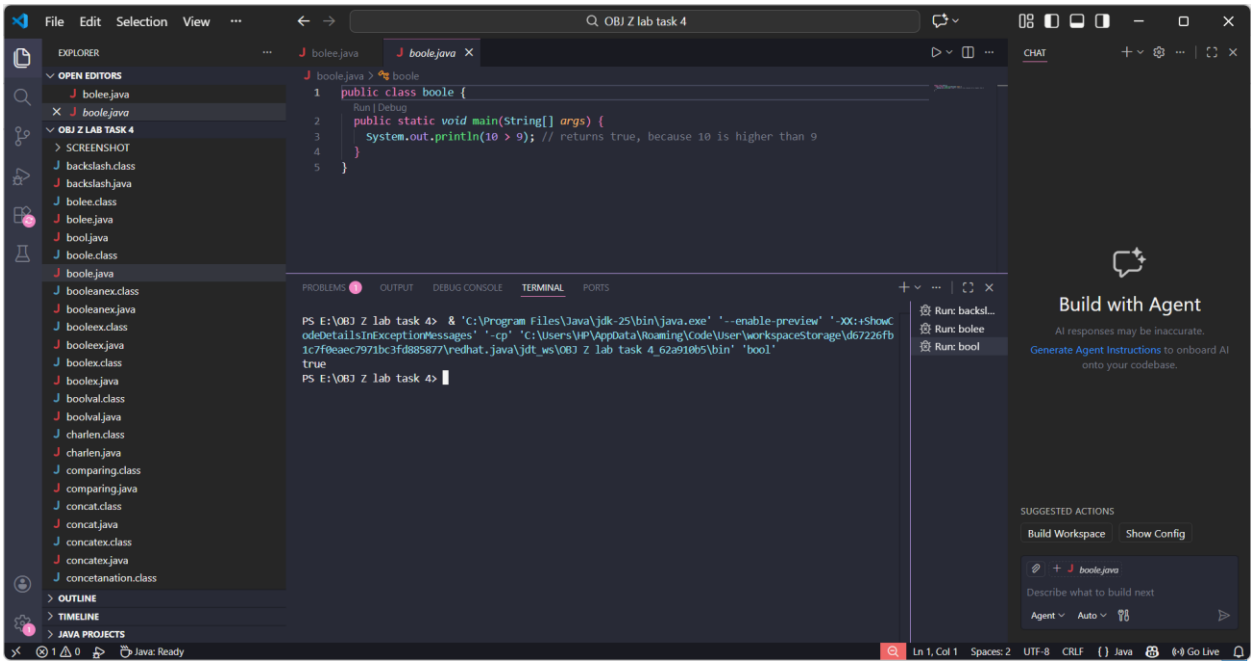
Submission Date: 30-01-2026



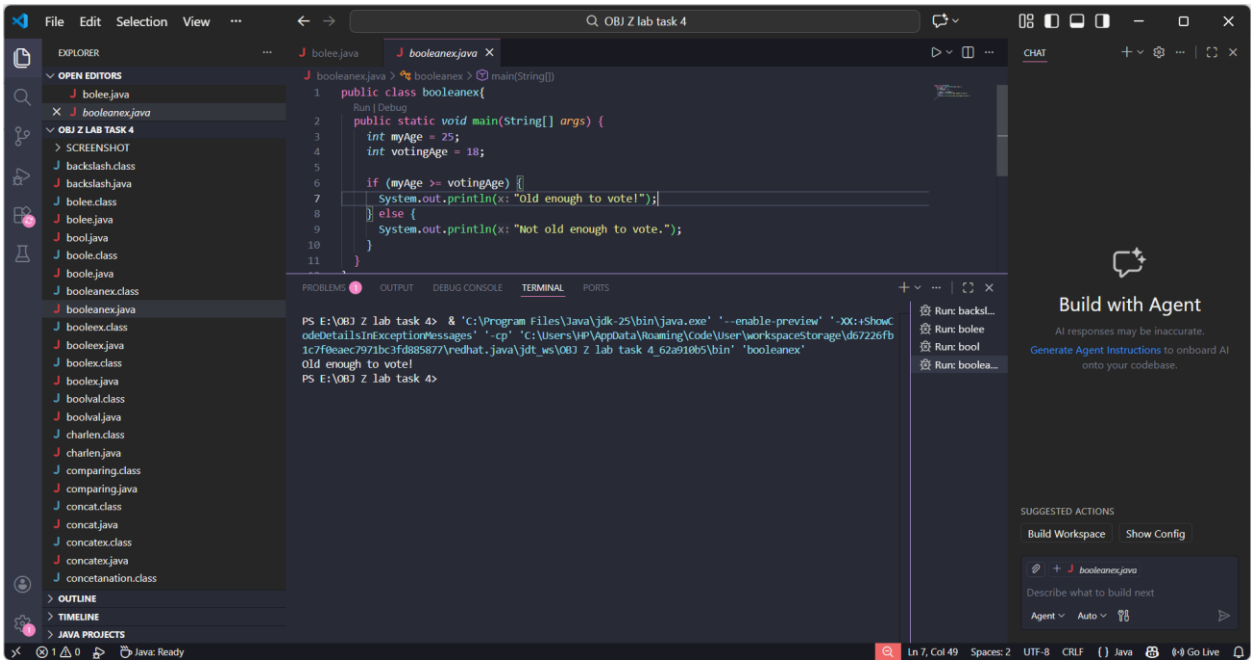
Two integers and false because 10 is equal to 15 . its equality operator



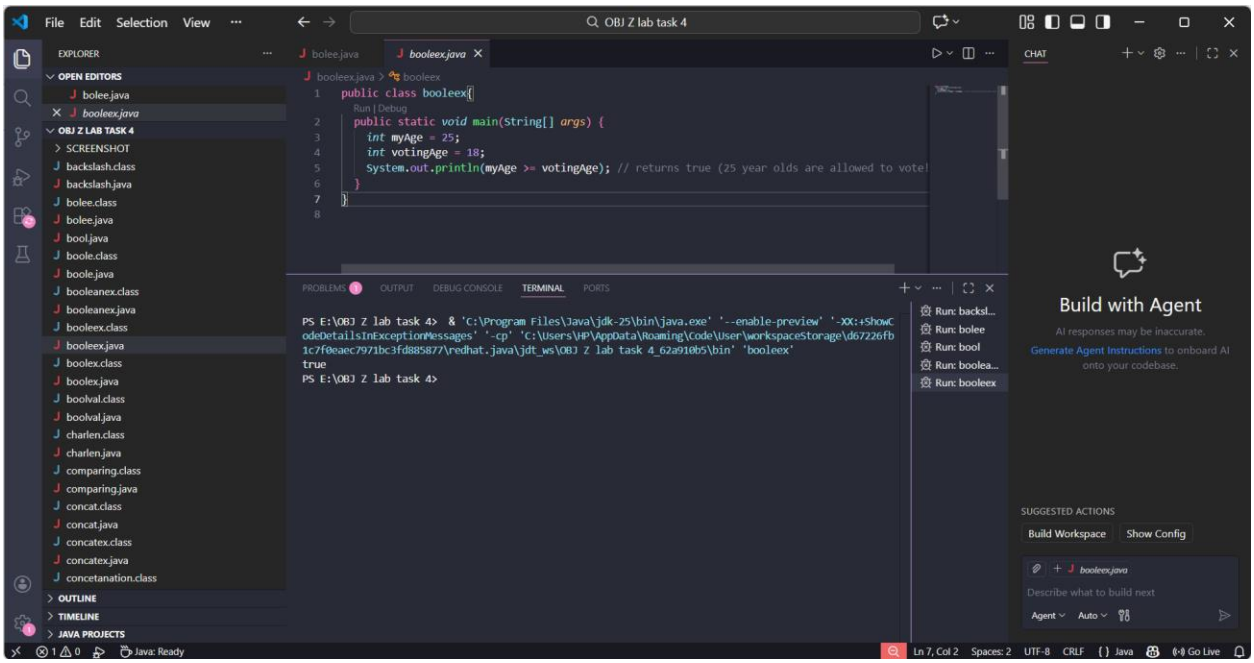
The code assigns the value 10 to a variable x and then checks if is equal to 10,which prints true.



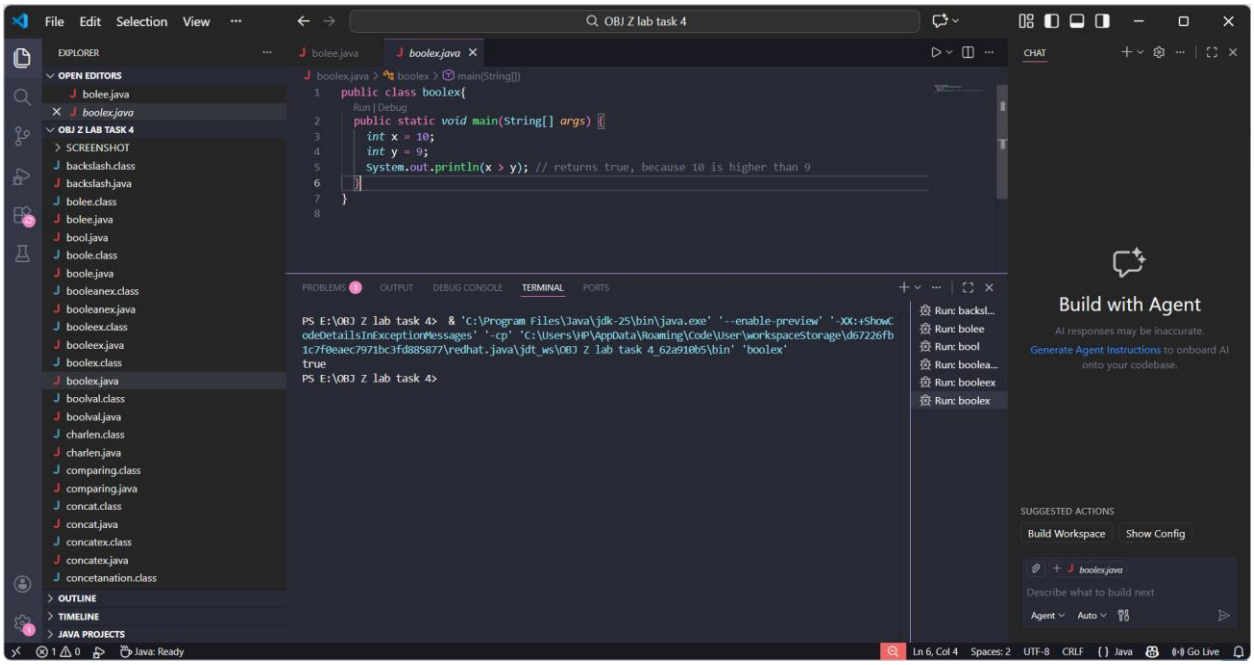
The code compares two integers and print true bcz 10 is greater then 9.



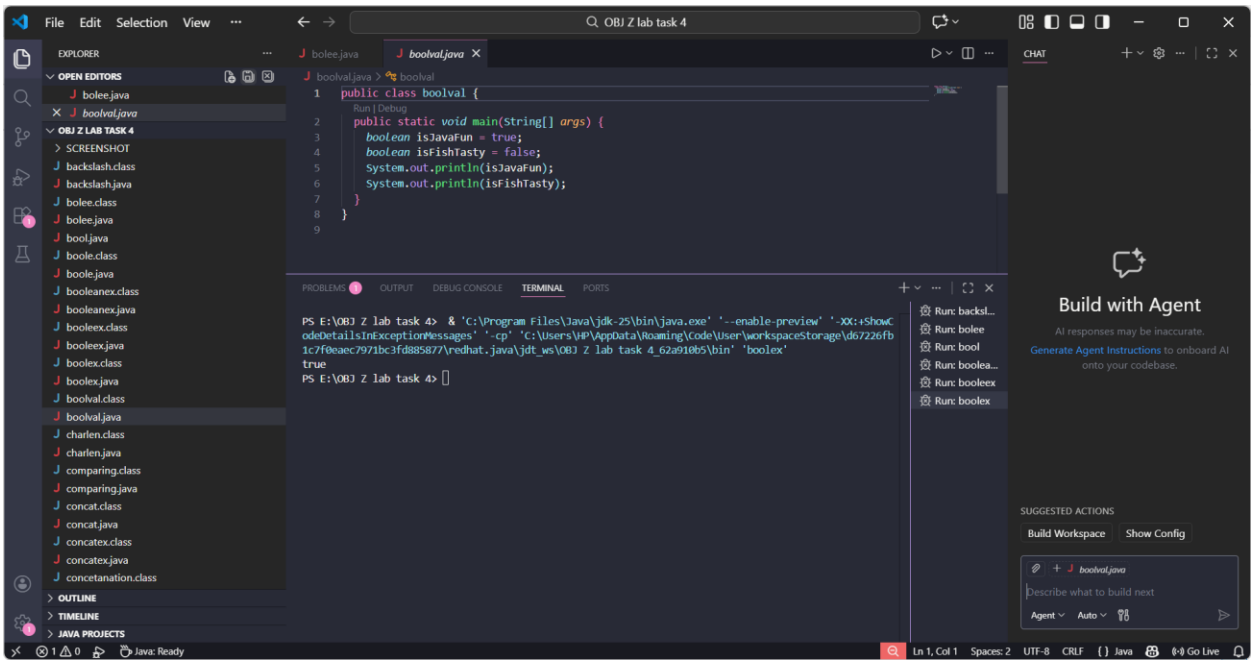
This code uses an if-else statement to compare my age .



This cod check veriables to compare my voatingage

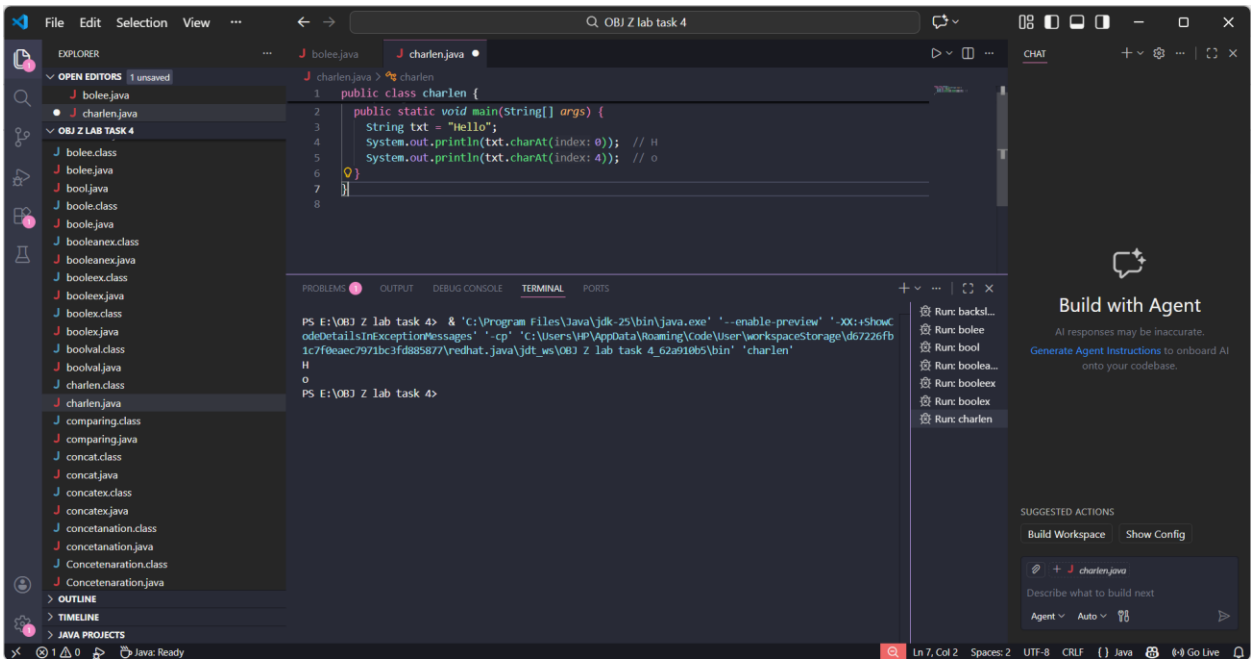


Checking variable using > this operator .

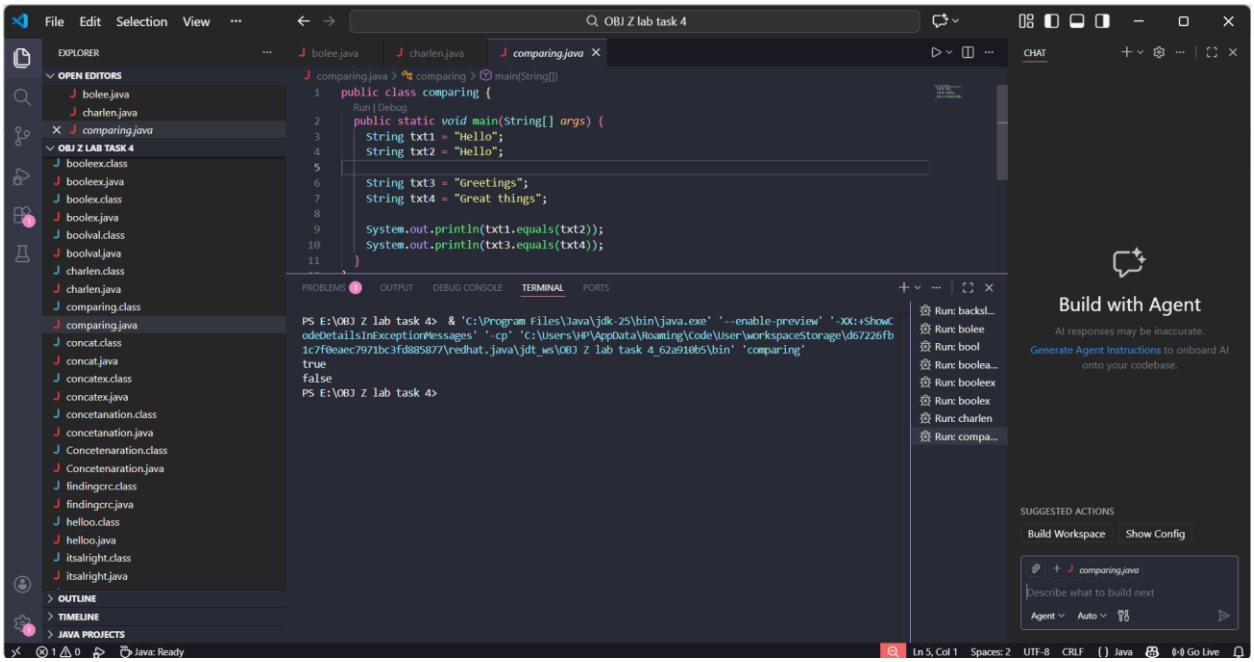


Demonstrates the basic declaration of true and false boolean types.

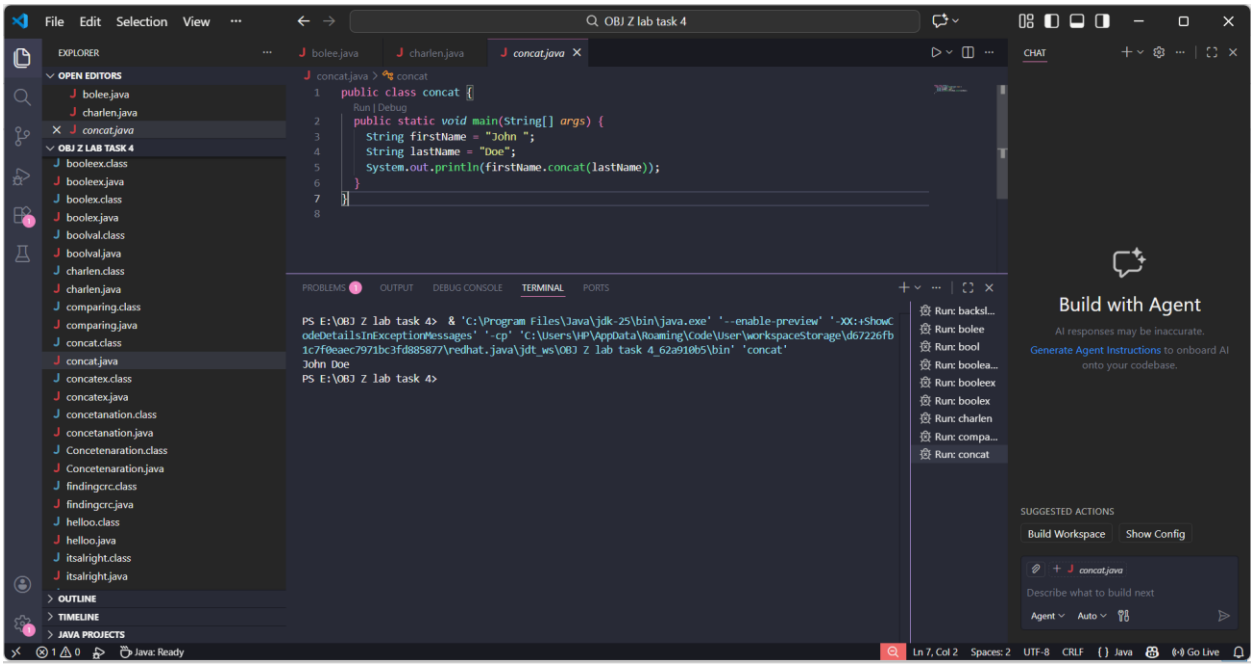
uses



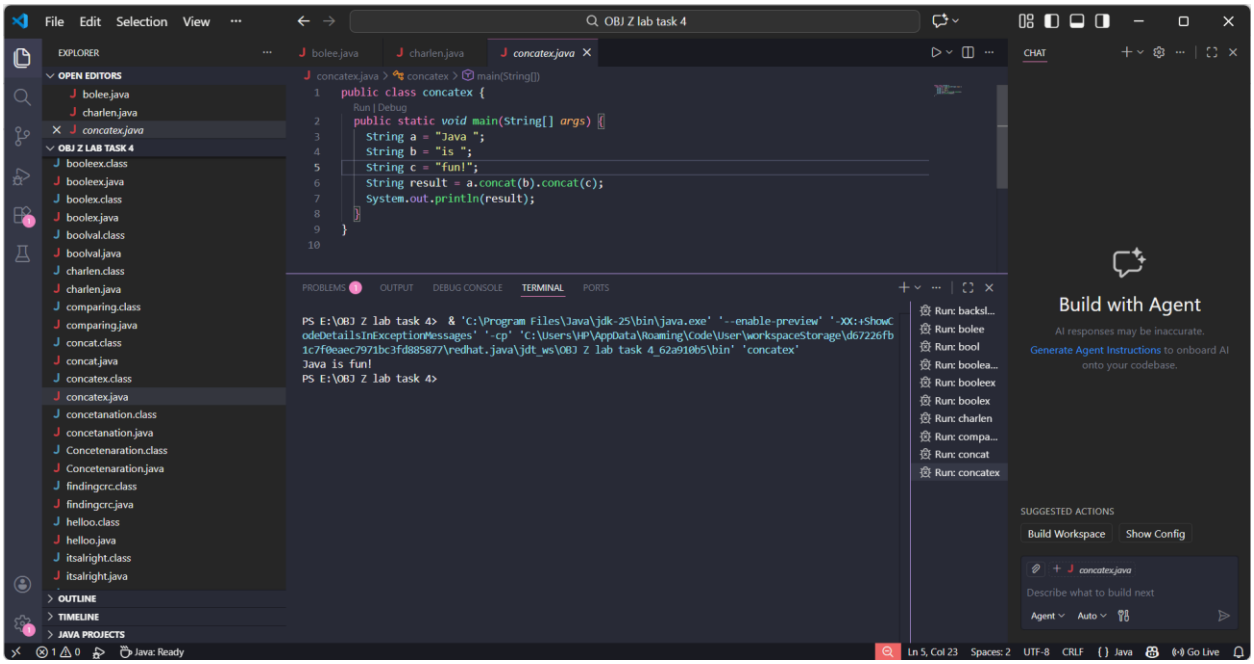
uses the charAt() method to extract and print the first character H and fifth character o.



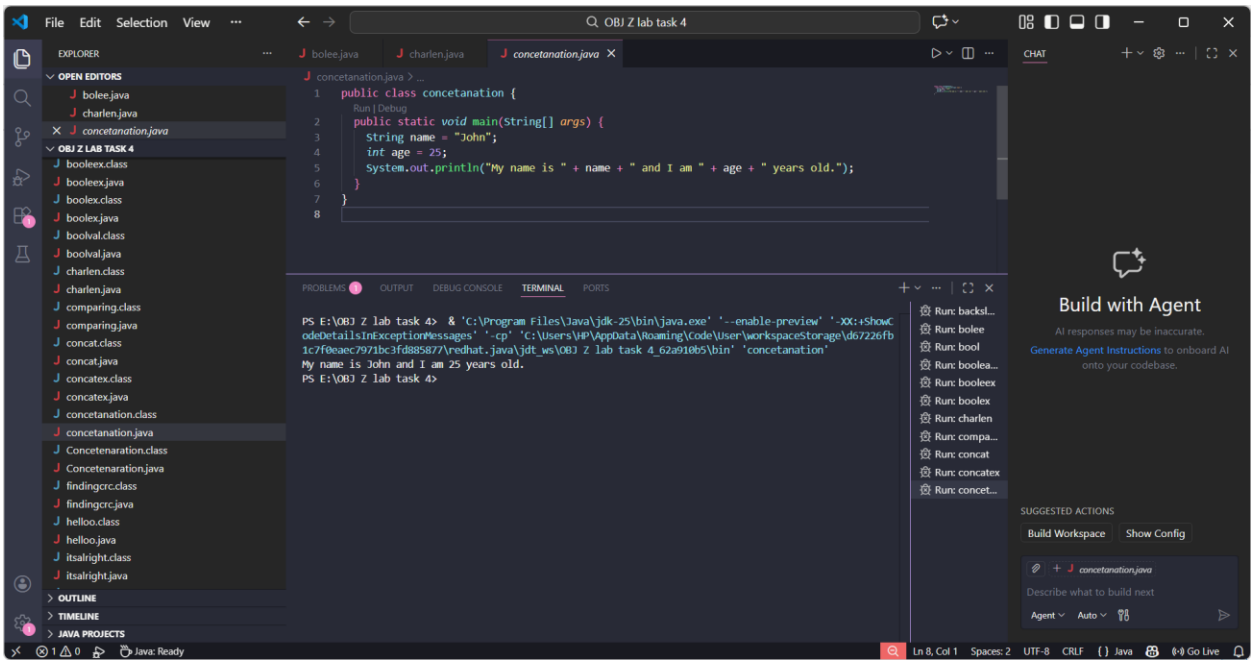
The code using equals () method to check if two strings match printing true and false .



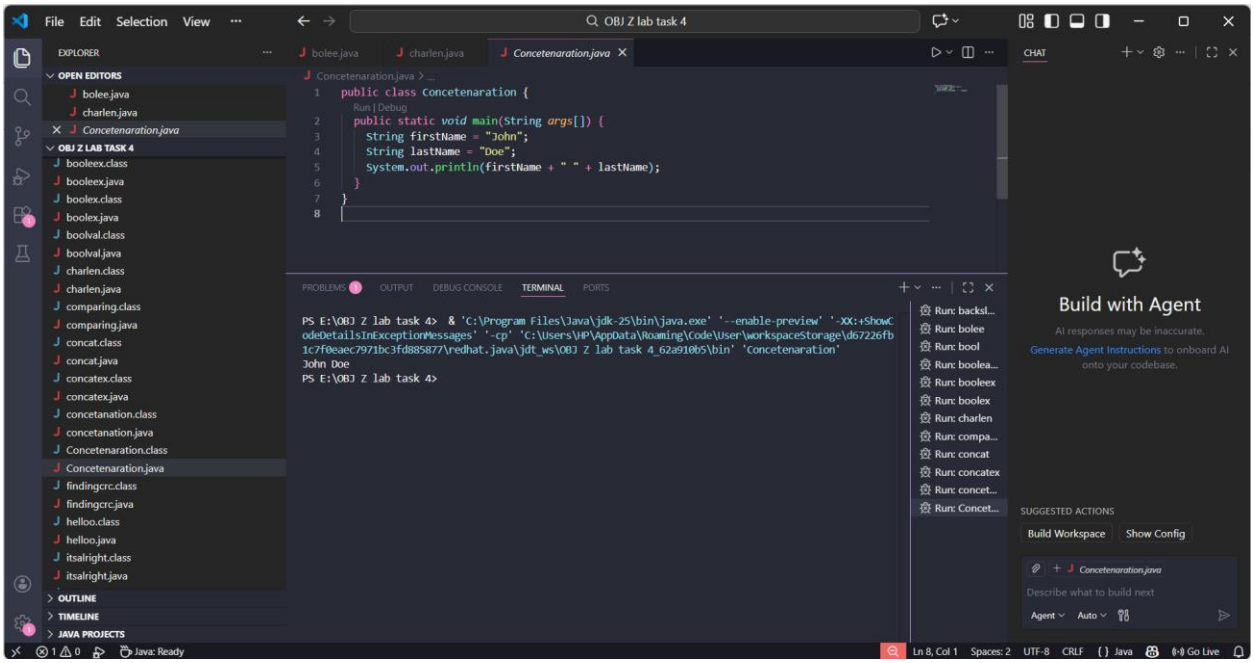
Concat() method to show output as full .



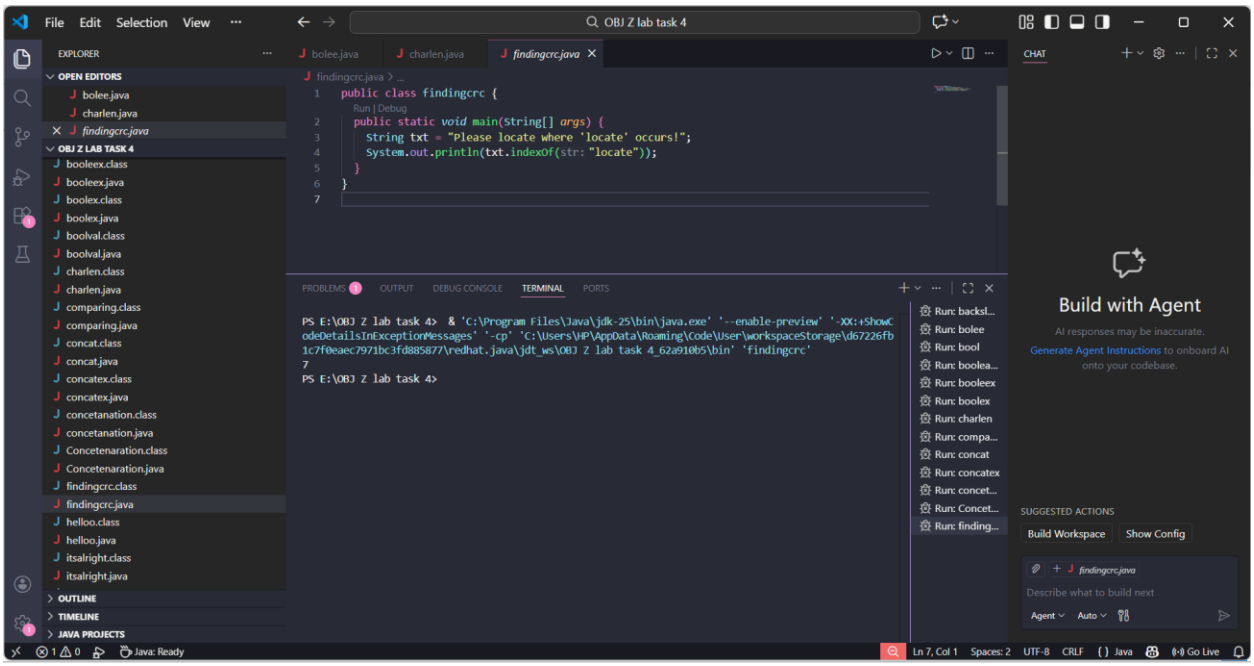
Adjust variable using concat() .



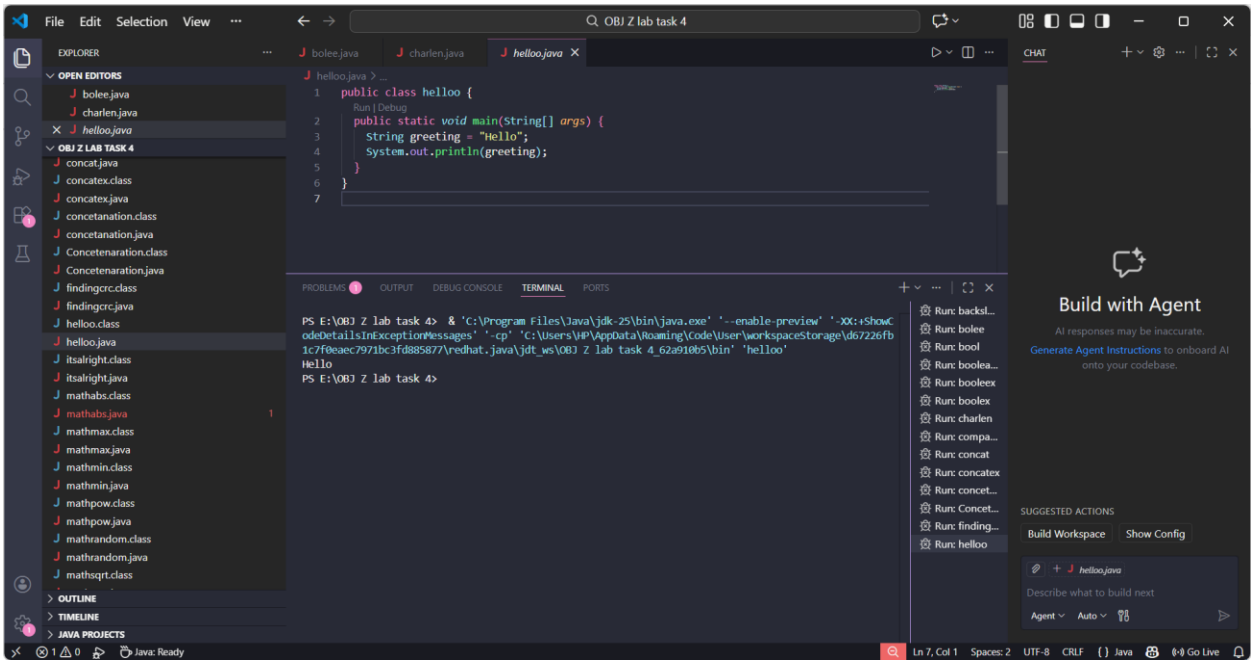
Basic java variable run to use name and age for output.



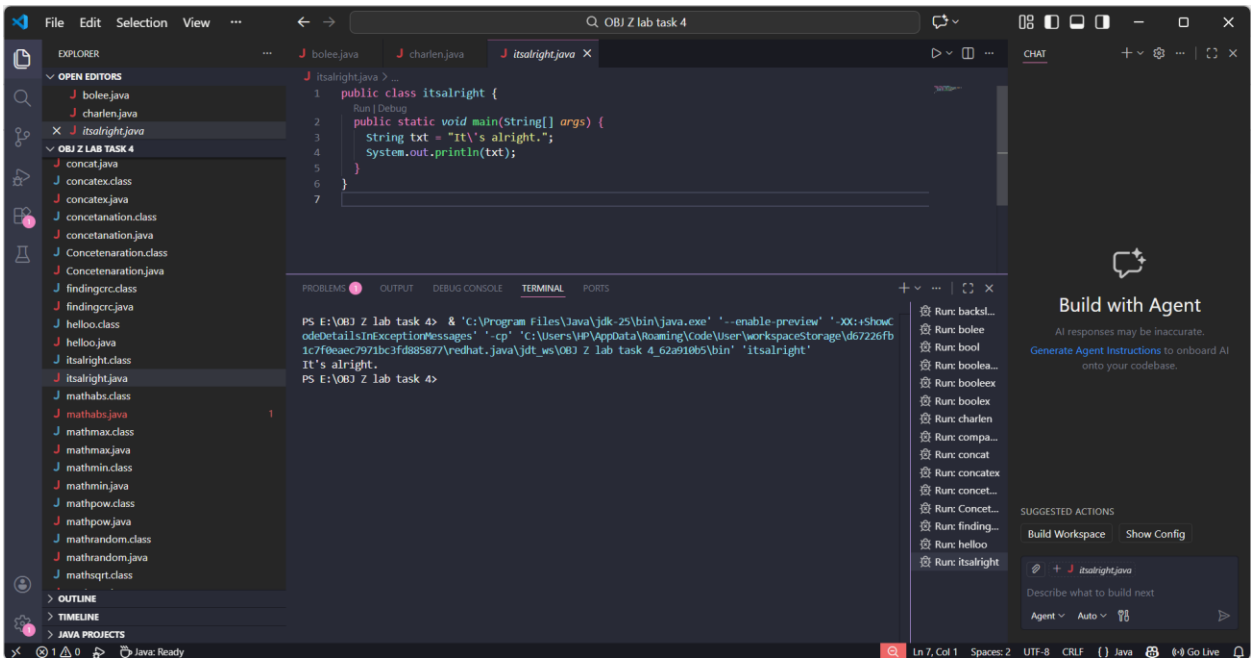
Joins strings and a space together using the **+ operator**, which is the most common way to concatenate in Java.



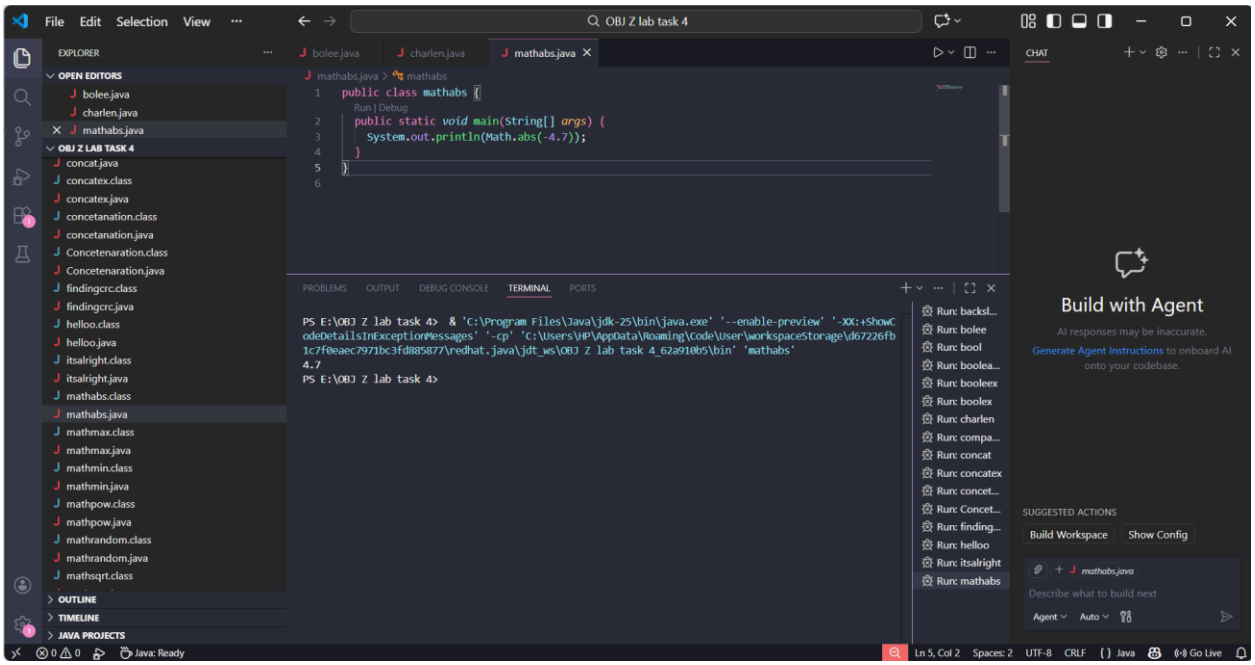
This Java code uses `indexOf("locate")` to find and print the index position of the first occurrence of the word **"locate"** in the string `txt`.



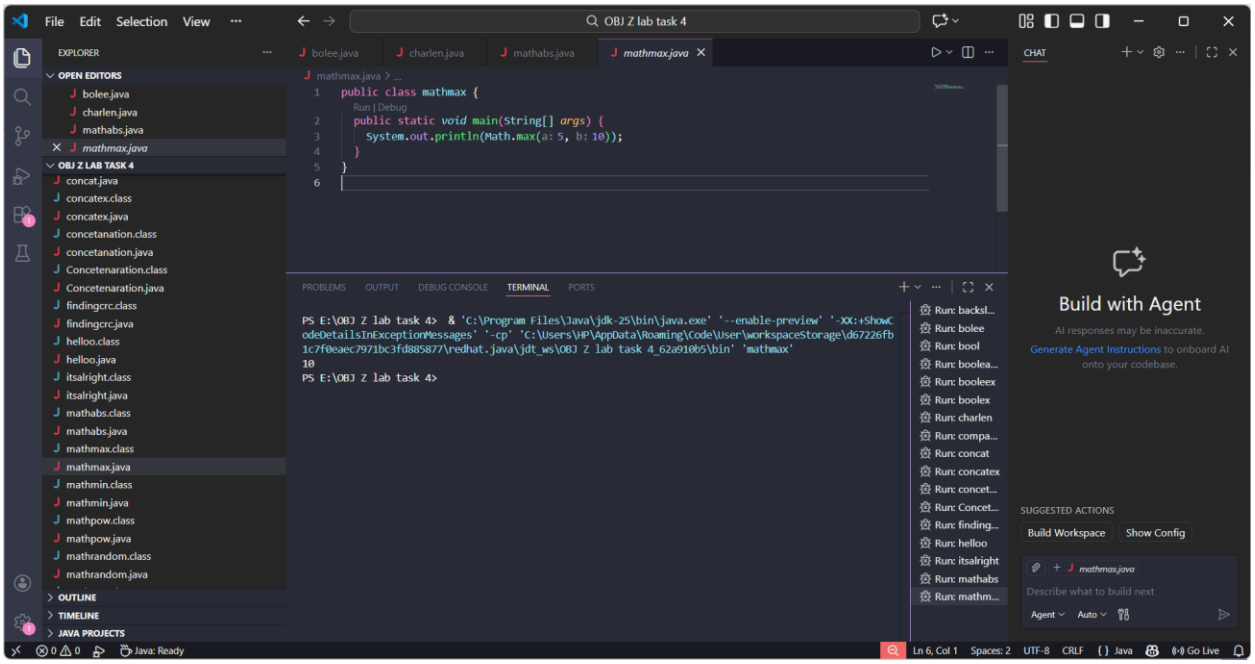
This Java program stores **"Hello"** in a string variable and prints it to the console.



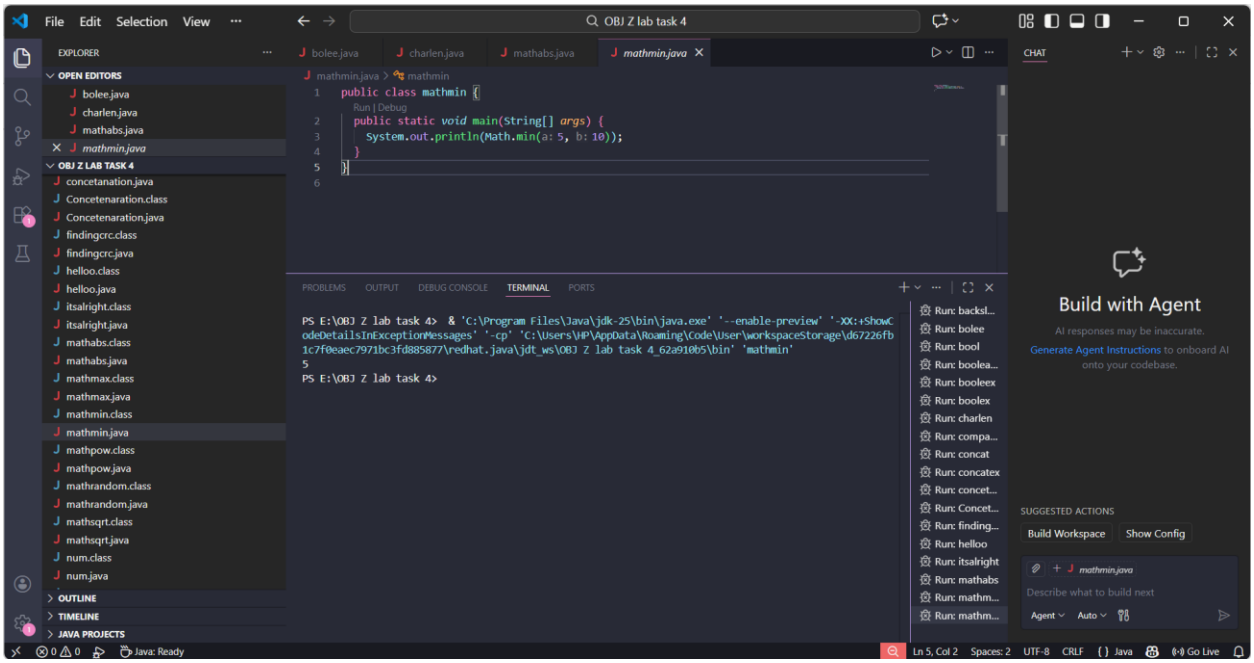
This screenshot shows VS Code running `itsalright.java` and printing **“It’s alright.”** in the terminal.



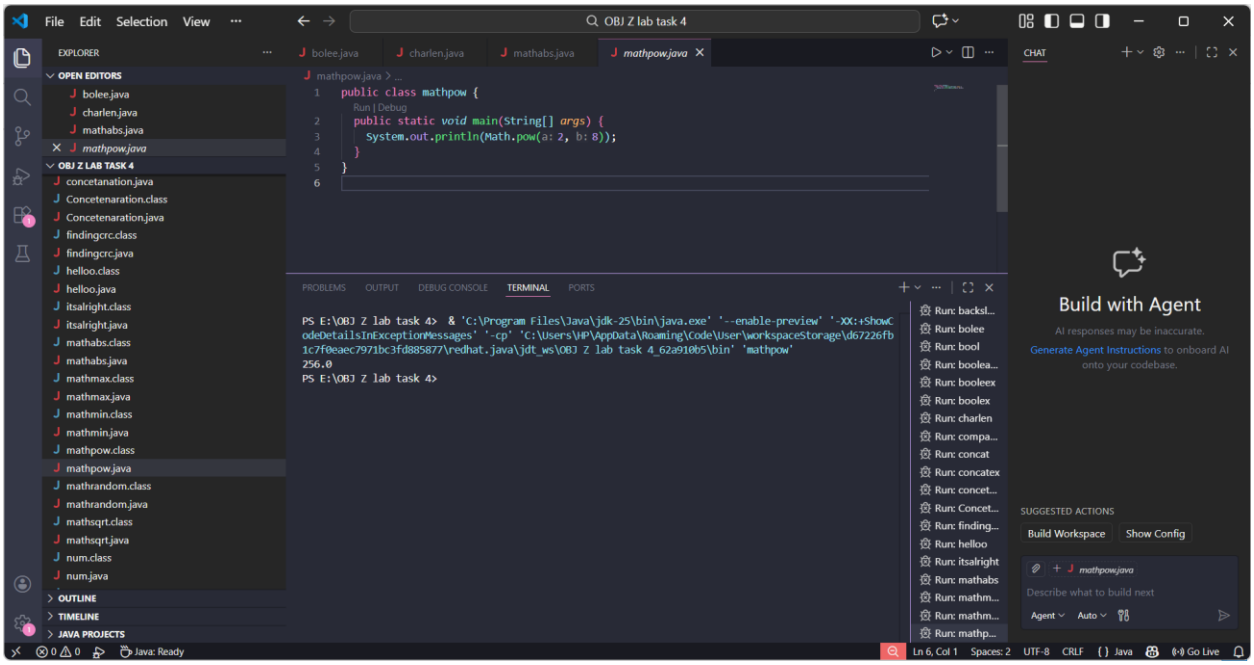
this program correctly prints `4.7` because `Math.abs(-4.7)` returns the absolute (positive) value of -4.7.



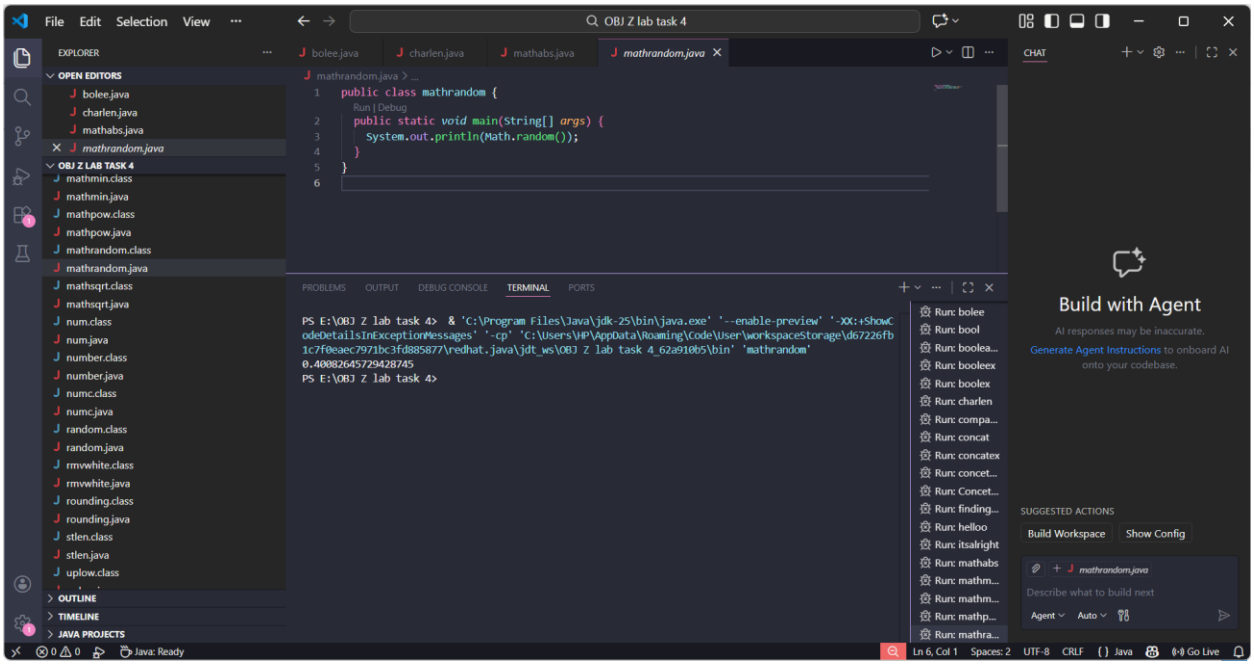
this program prints `10` because `Math.max(5, 10)` returns the larger of the two numbers (here 10).



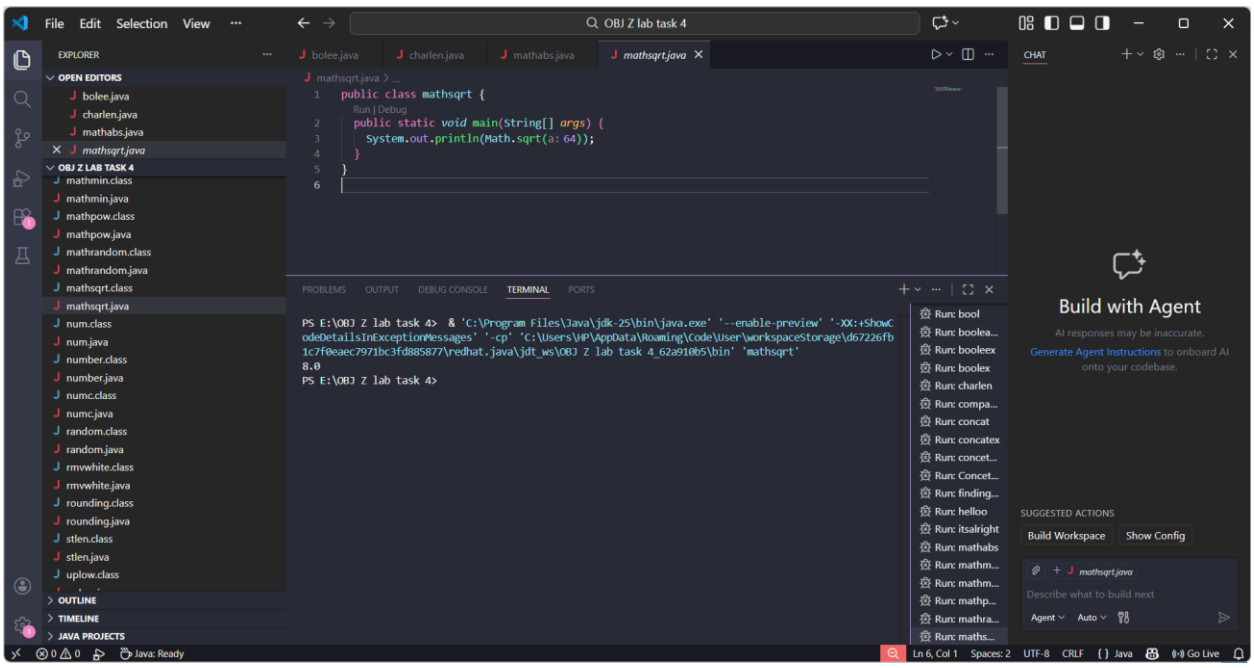
this program prints 5 because Math.min(5, 10) returns the smaller of the two numbers (here 5).



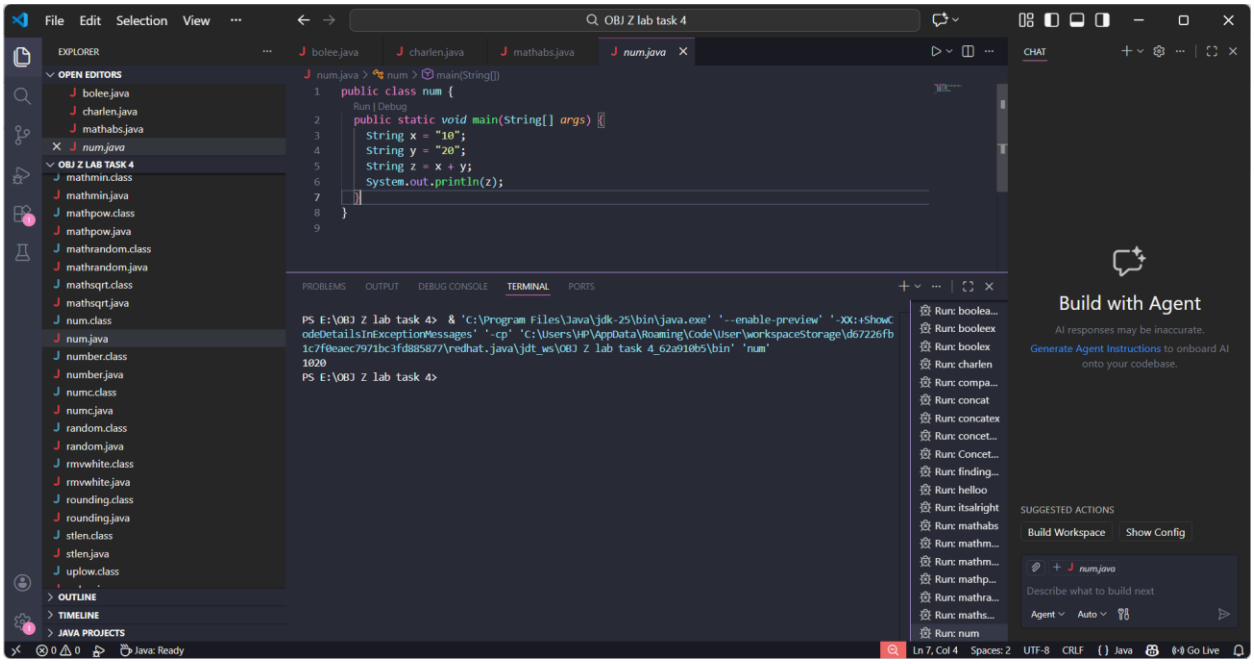
this program prints `256.0` because `Math.pow(2, 8)` calculates 2 raised to the power of 8, which equals 256.



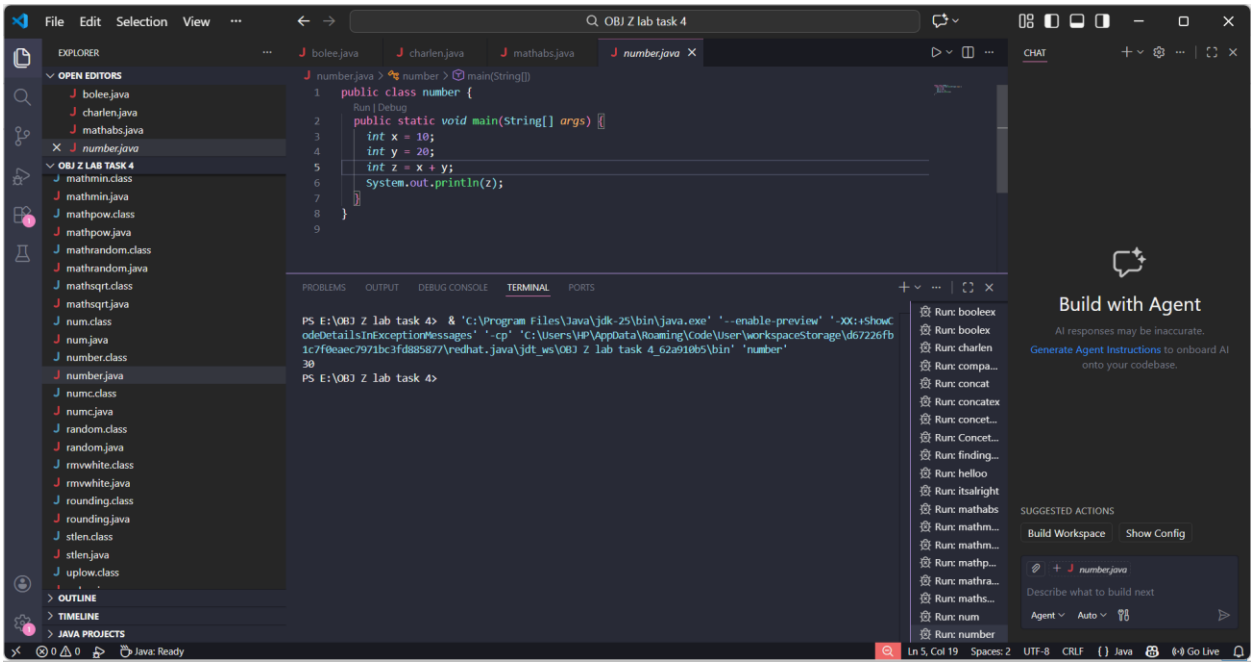
This program prints a random decimal number between 0.0 (inclusive) and 1.0 (exclusive) because Math.random() generates a pseudo-random double in that range — every run gives a different value (here ≈0.40088…).



this program prints a random decimal number between 0.0 (inclusive) and 1.0 (exclusive) because Math.random() generates a pseudo-random double in that range — every run gives a different value (here ≈0.40088…).

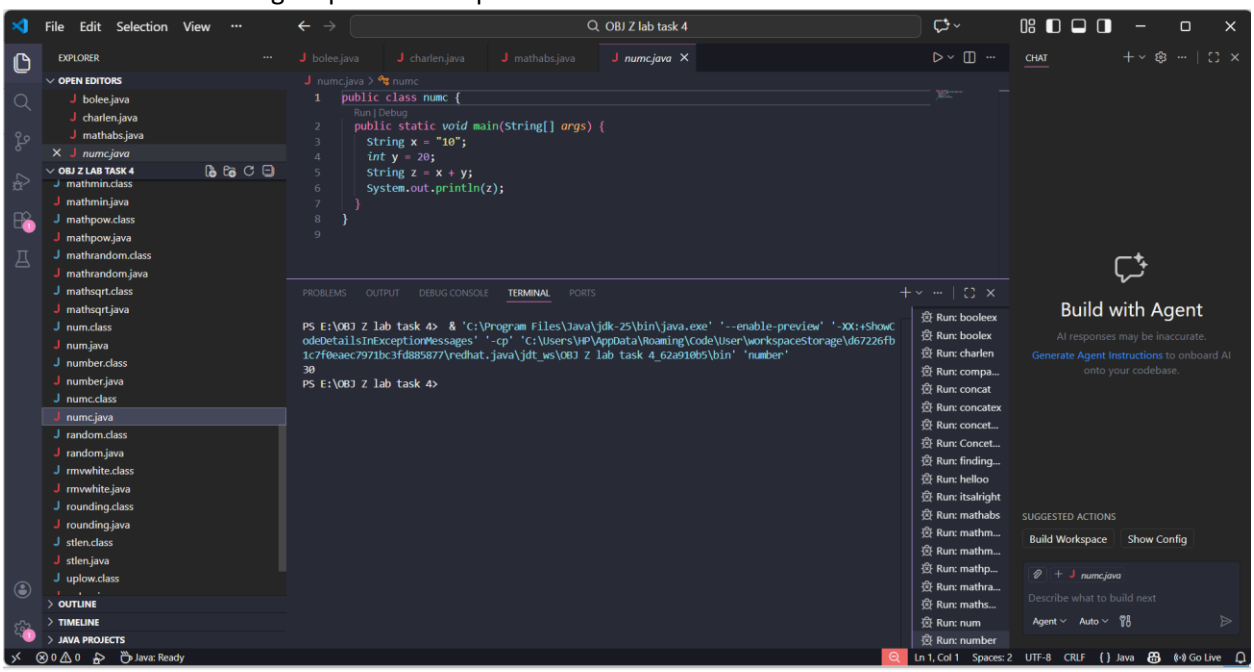


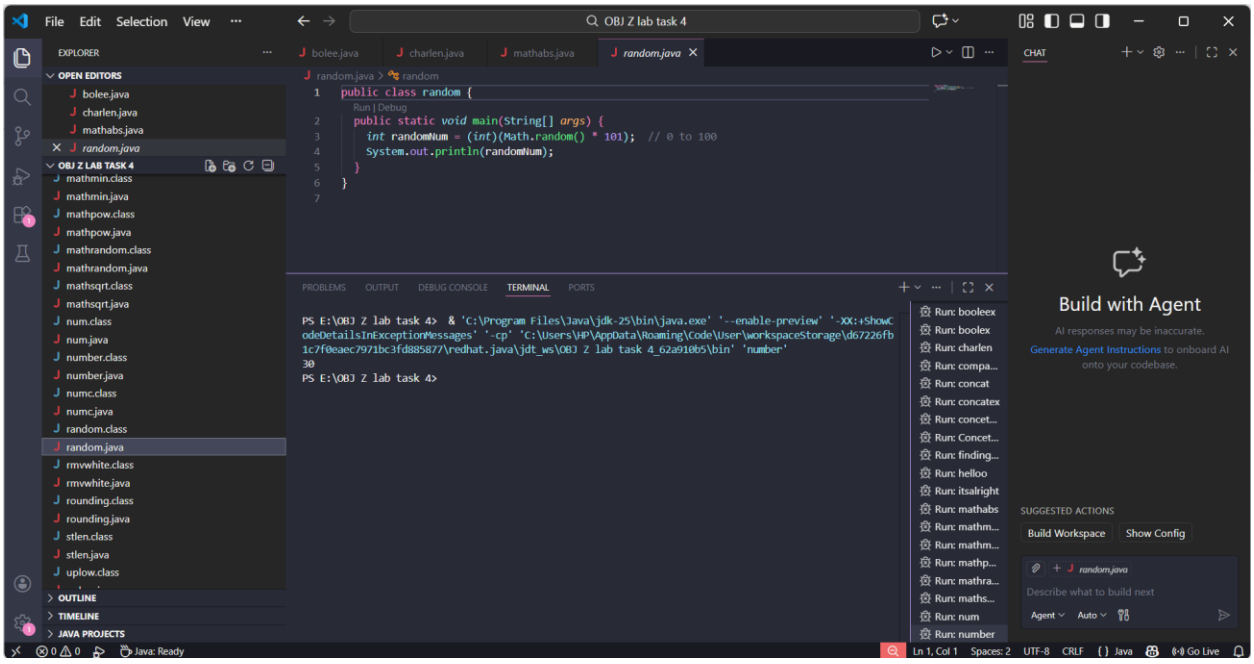
This program prints `1020` because `+` between Strings in Java does concatenation, so `"10" + "20"` joins them as the string `"1020"`.



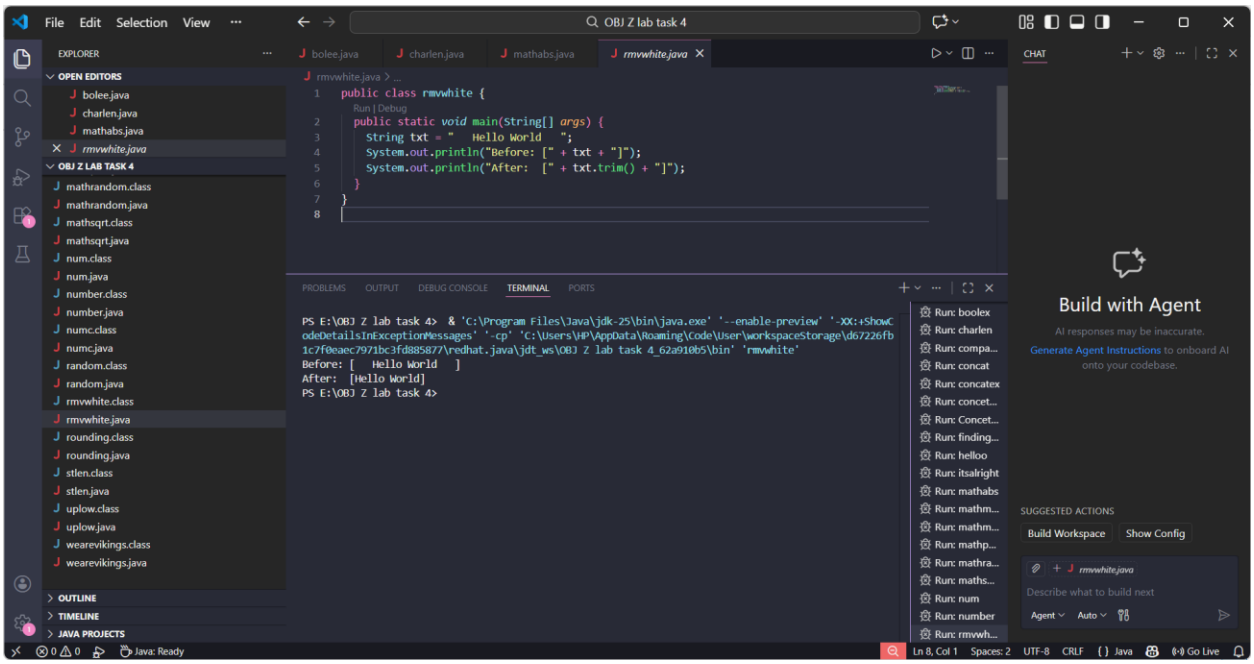
Adding two variables usig + operator .

Add two varriable using + operator and print as z.

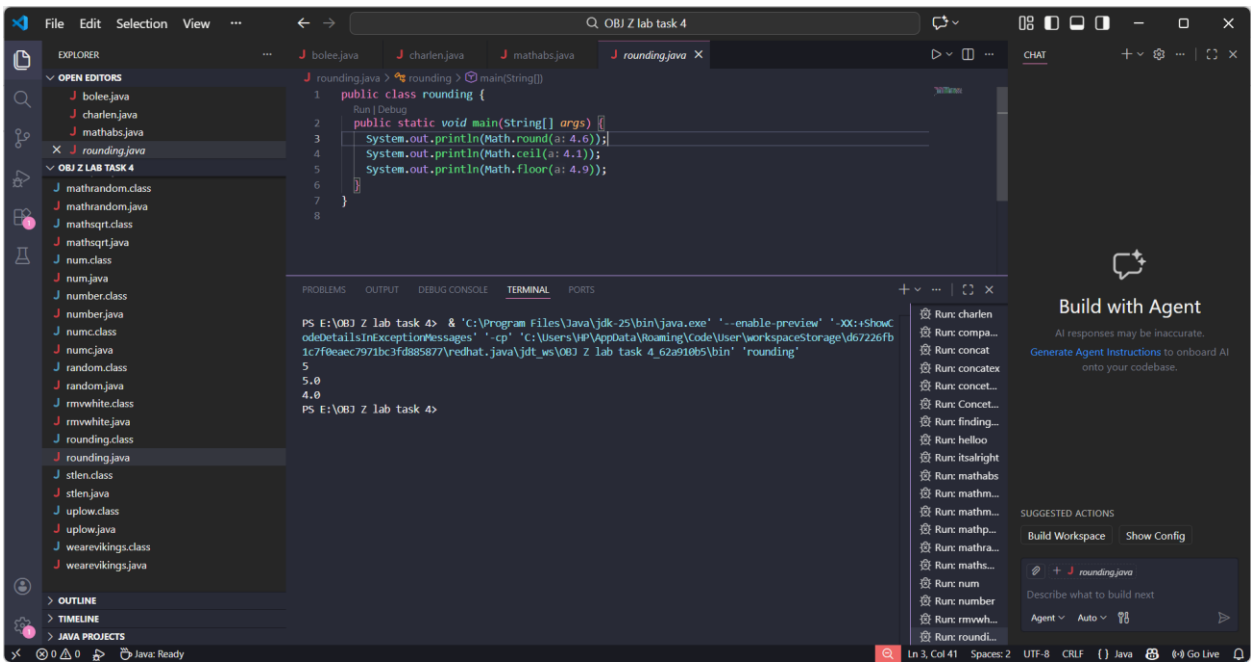




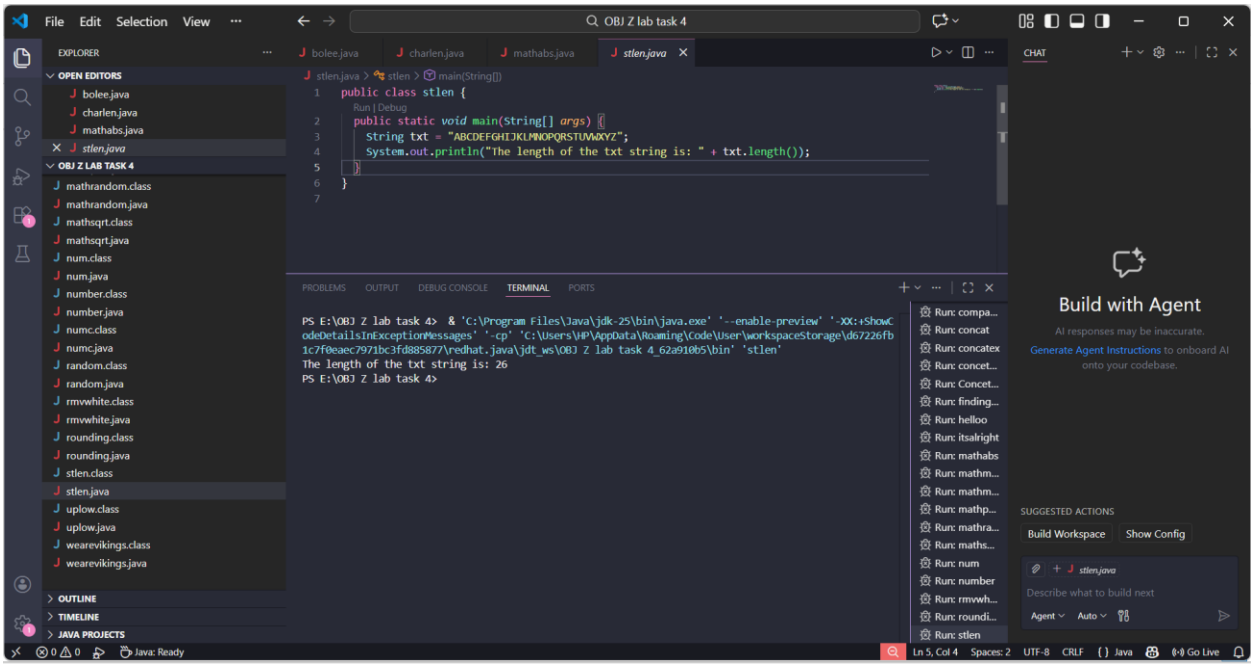
This program prints a random integer between 0 and 100 (inclusive) because `(int)(Math.random() * 101)` generates a random number from 0.0 to 100.999… and casting to `int` gives 0–100 (here



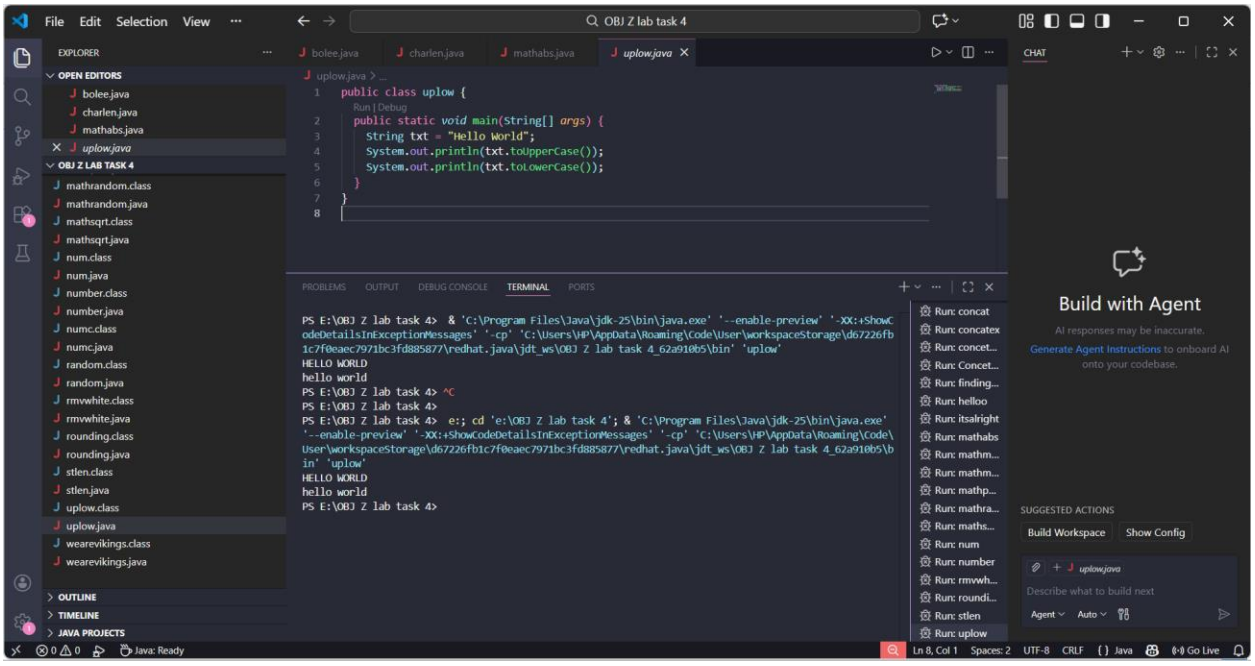
this program prints "Hello World " with a trailing space before .trim() and "Hello World" without trailing/leading spaces after .trim(), showing that String.trim() removes whitespace from both ends of the string.



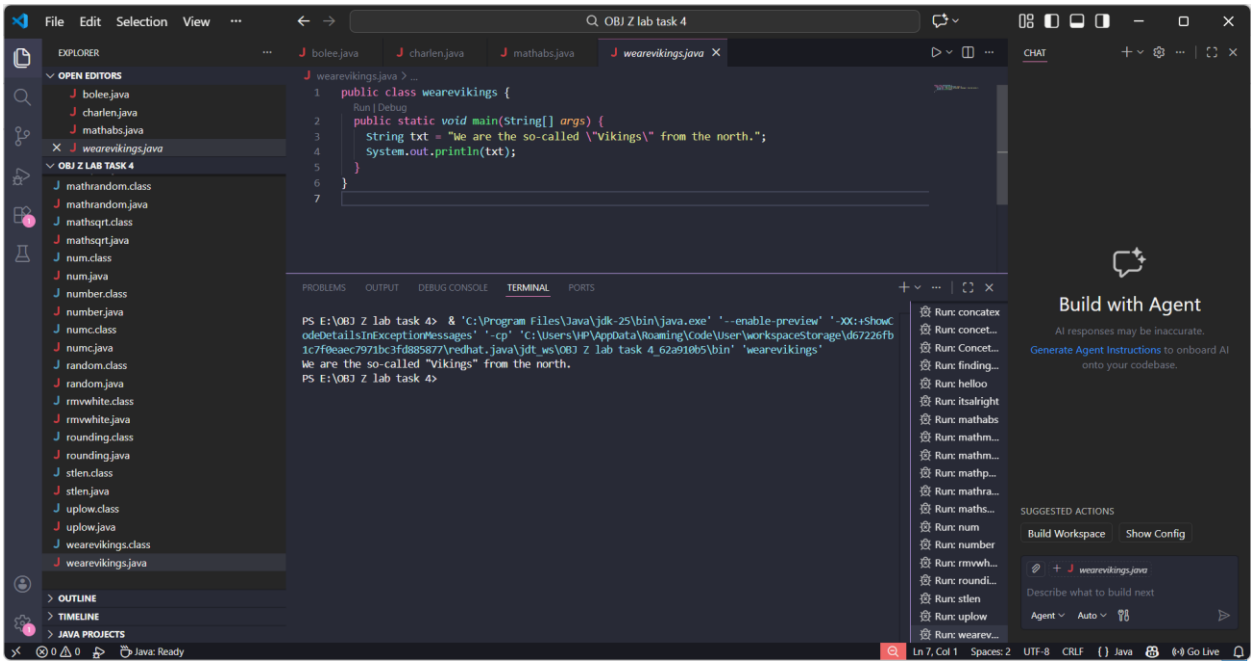
This program prints `5` `5.0` `4.0` because `Math.round(4.6)` rounds to nearest integer (5), `Math.ceil(4.1)` rounds up to 5.0, and `Math.floor(4.9)` rounds down to 4.0.



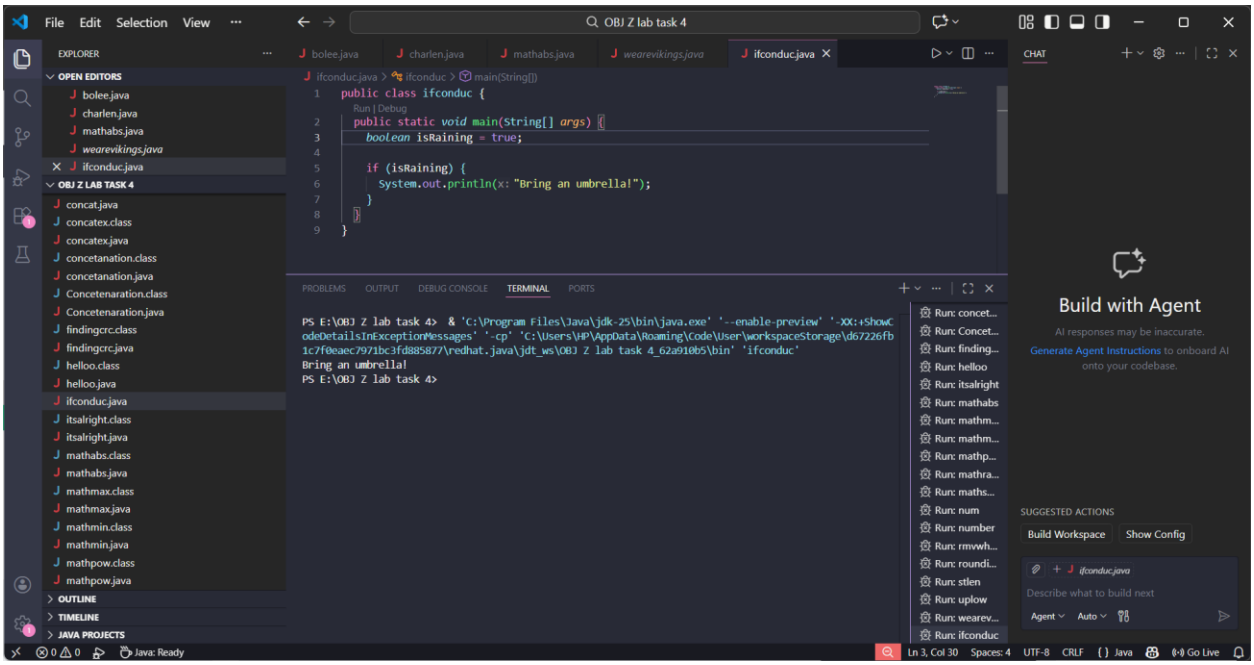
Using length() we know every sentence word length .



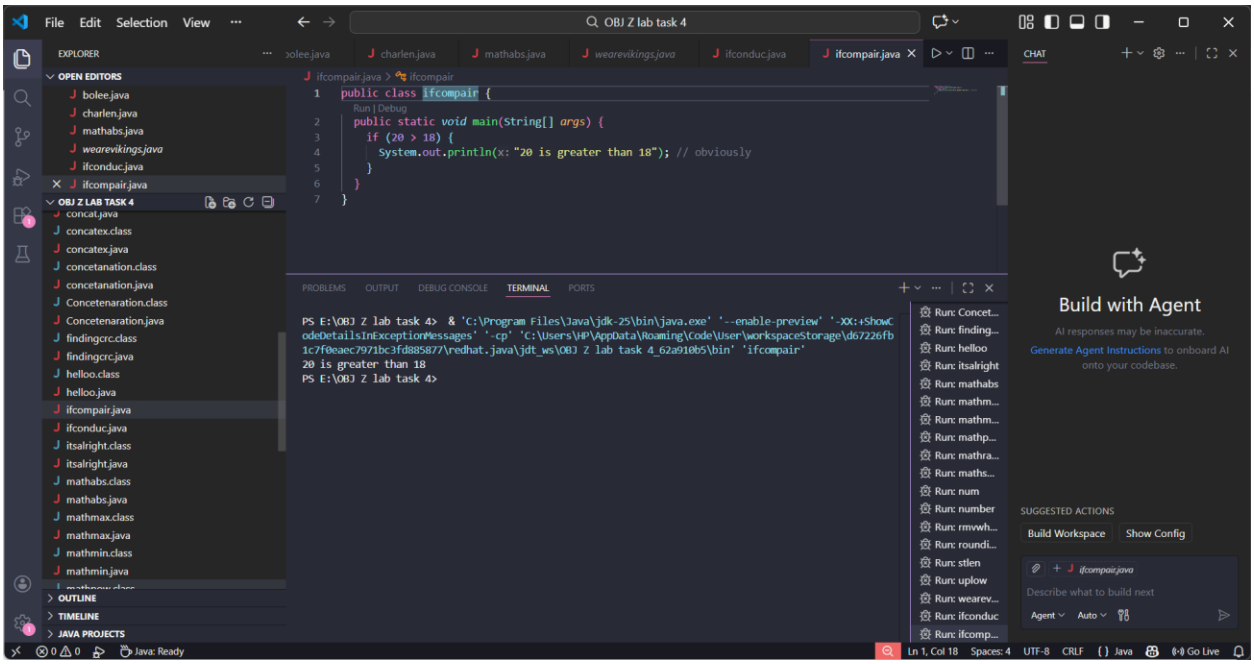
because toUpperCase() converts the string to all uppercase, toLowerCase() converts it to all lowercase, and the original string remains unchanged (strings are immutable in Java).



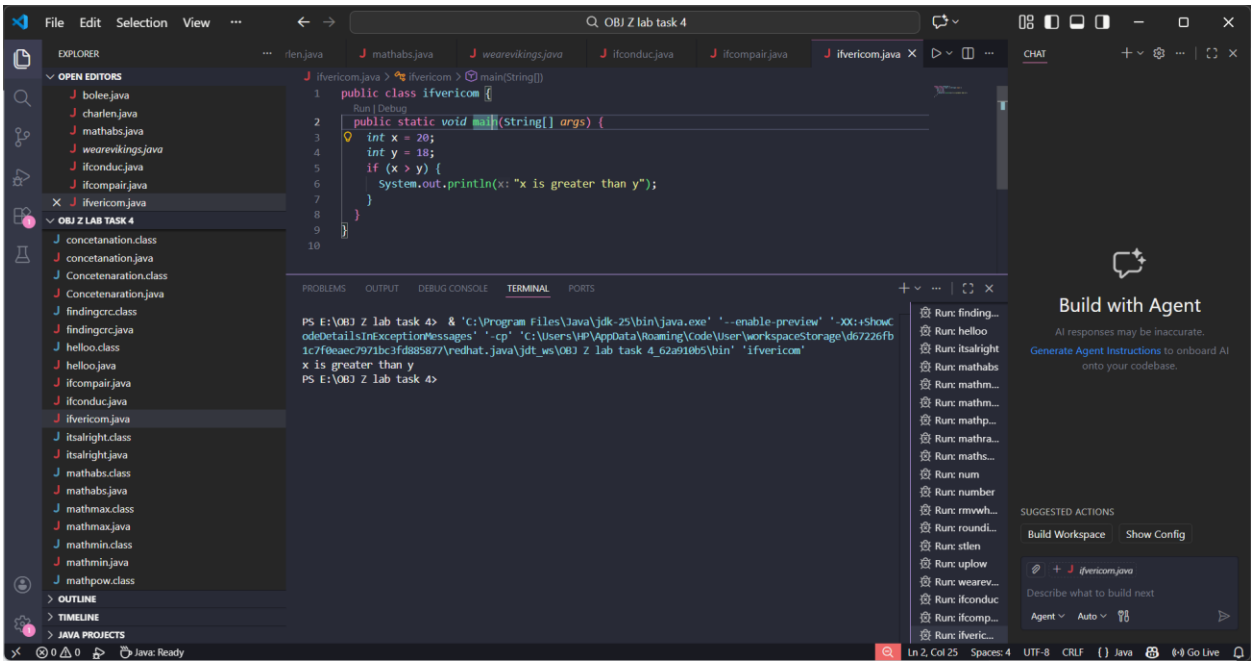
Print string txt variable sentence in print line .



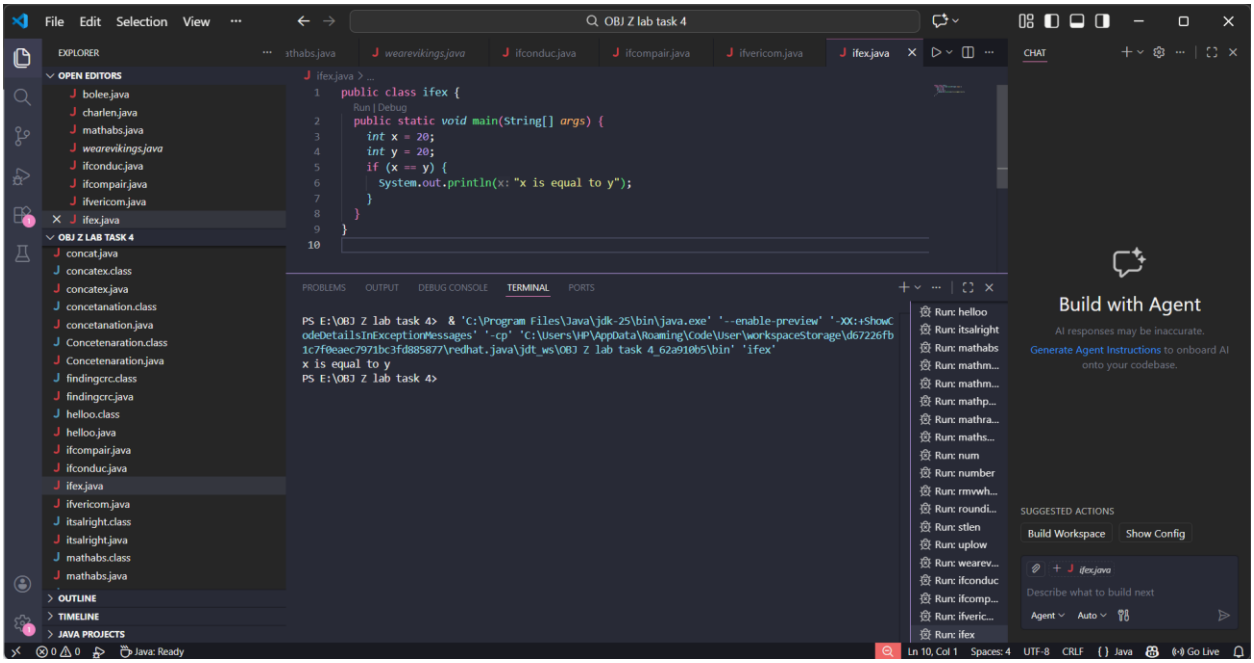
this program prints "Bring an umbrella!" because the if (isRaining) condition is true, so the message inside the if block executes.



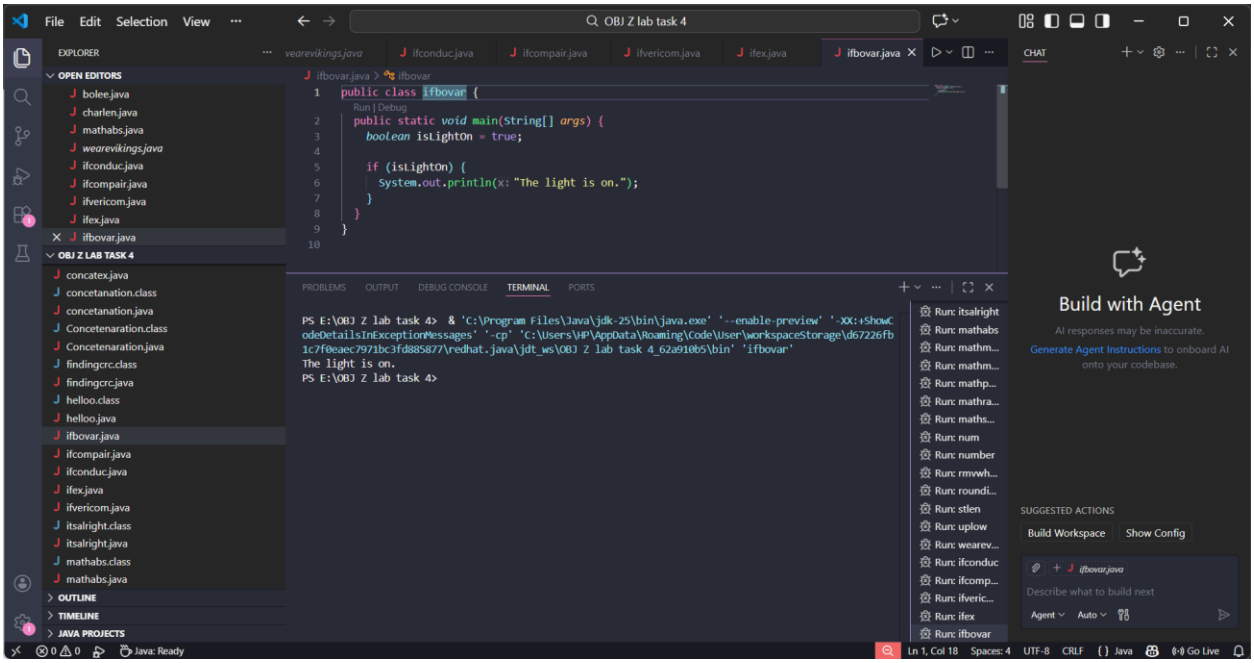
This program show us if else statement show us whats veriable is big.



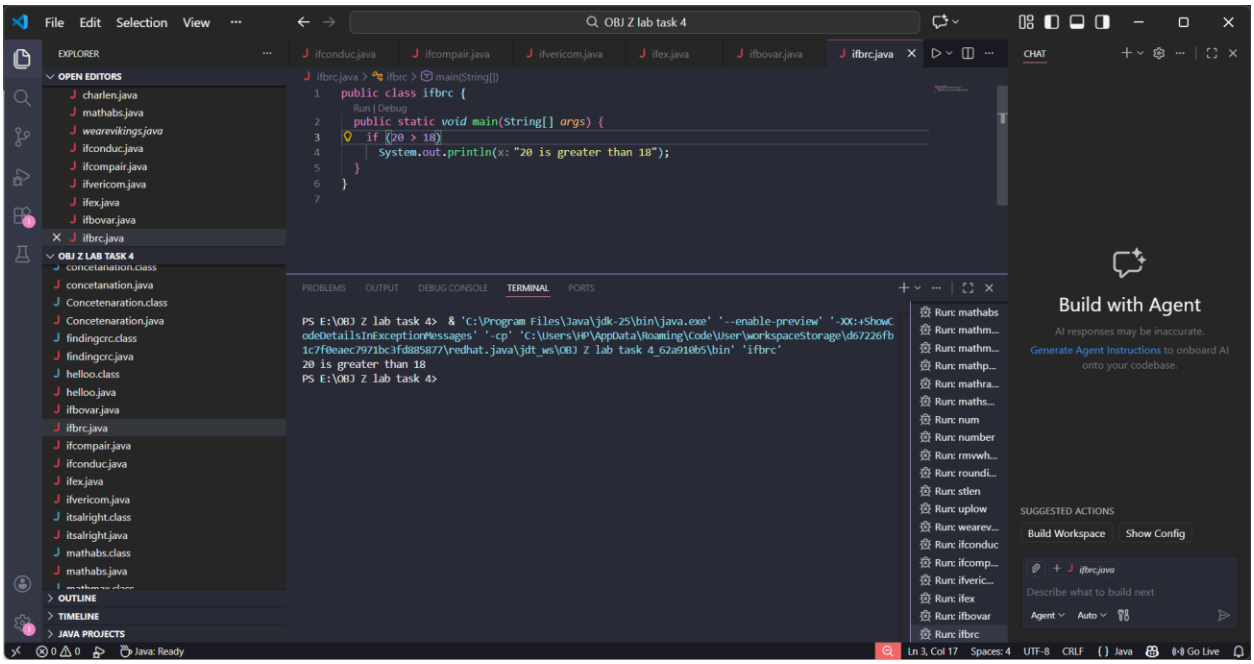
This program show us if else statement show us whats veriable is big.



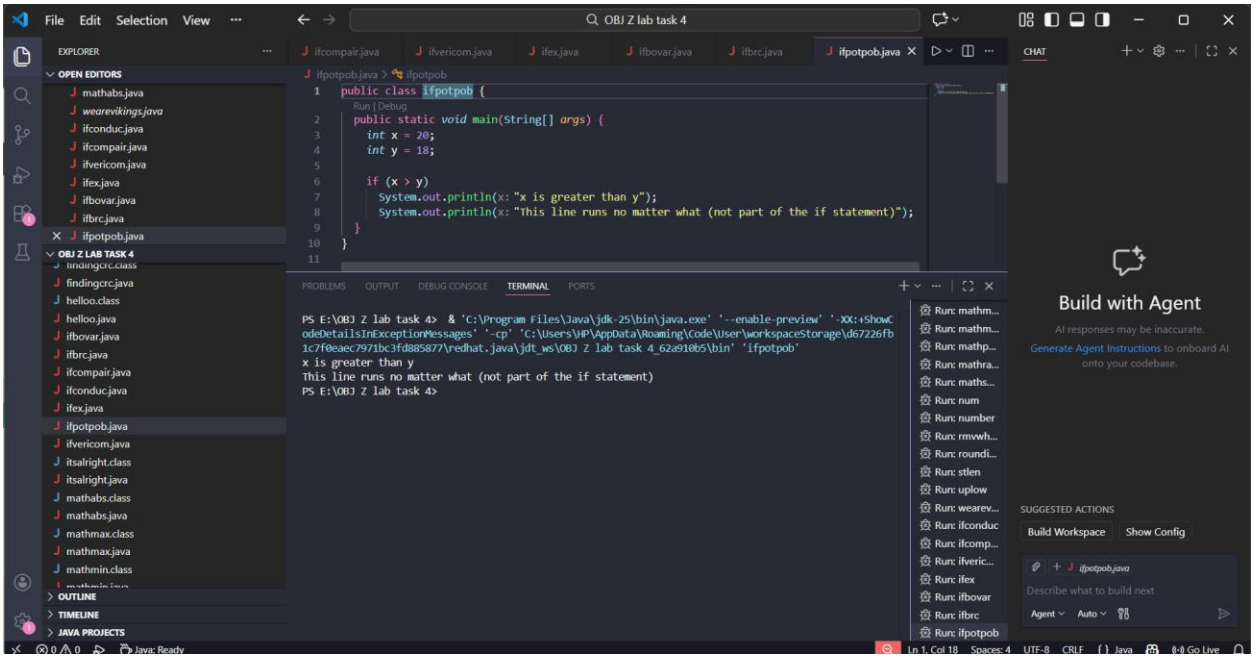
This program show us the equal statement show us whats veriable is equal or not.



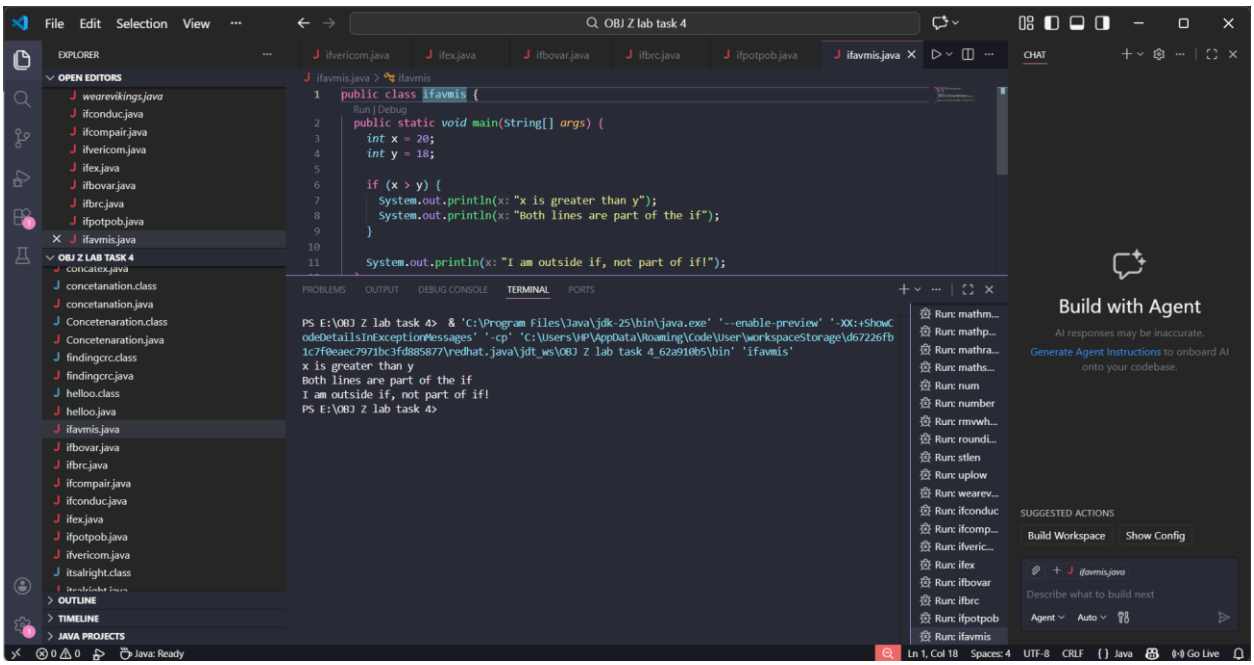
This program show us if else statement show us whats the veriable is enBLE OR NOT .



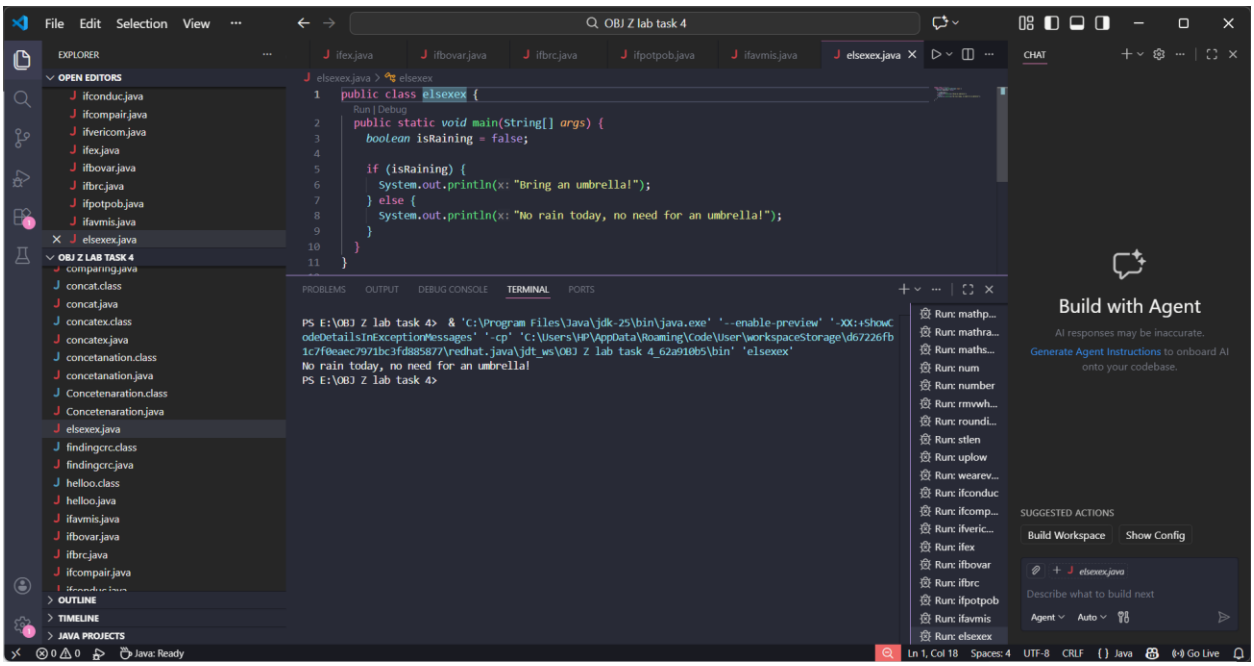
This program show us if else statement show us whats veriable is big.



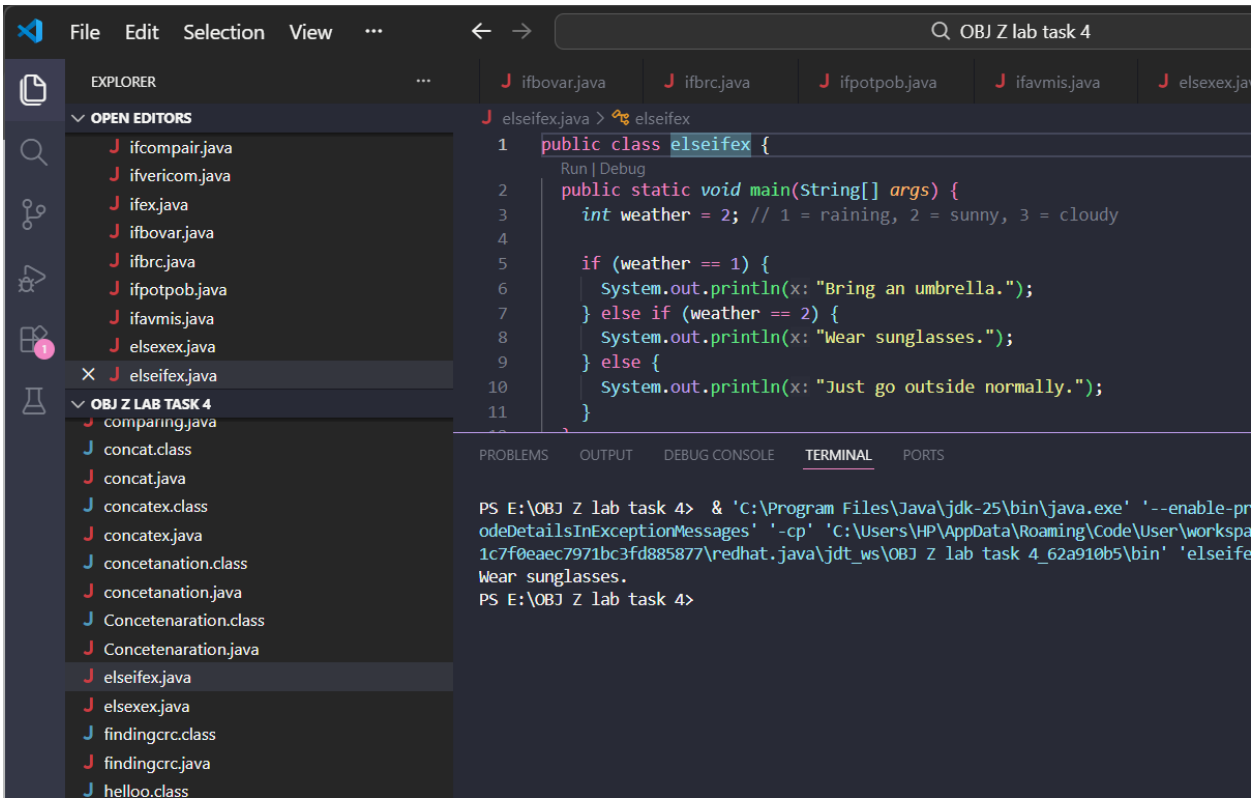
This program show us if else statement show us whats veriable is big. THIA LINE SHOW NO METTER WHAT EVERY LINE WE INPUT SHOW AS OUTPUT.



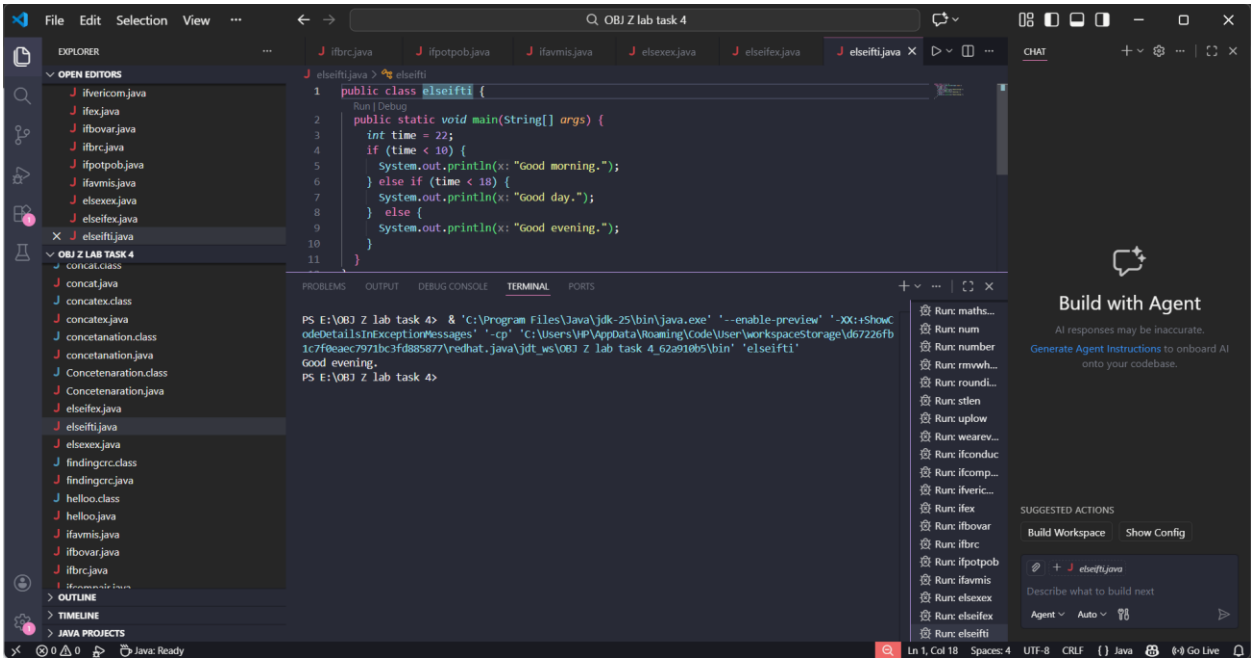
This code demonstrates **scope** issues. Because there are no curly braces {} after the if statement, only the first line is conditional. The second print statement runs no matter what.



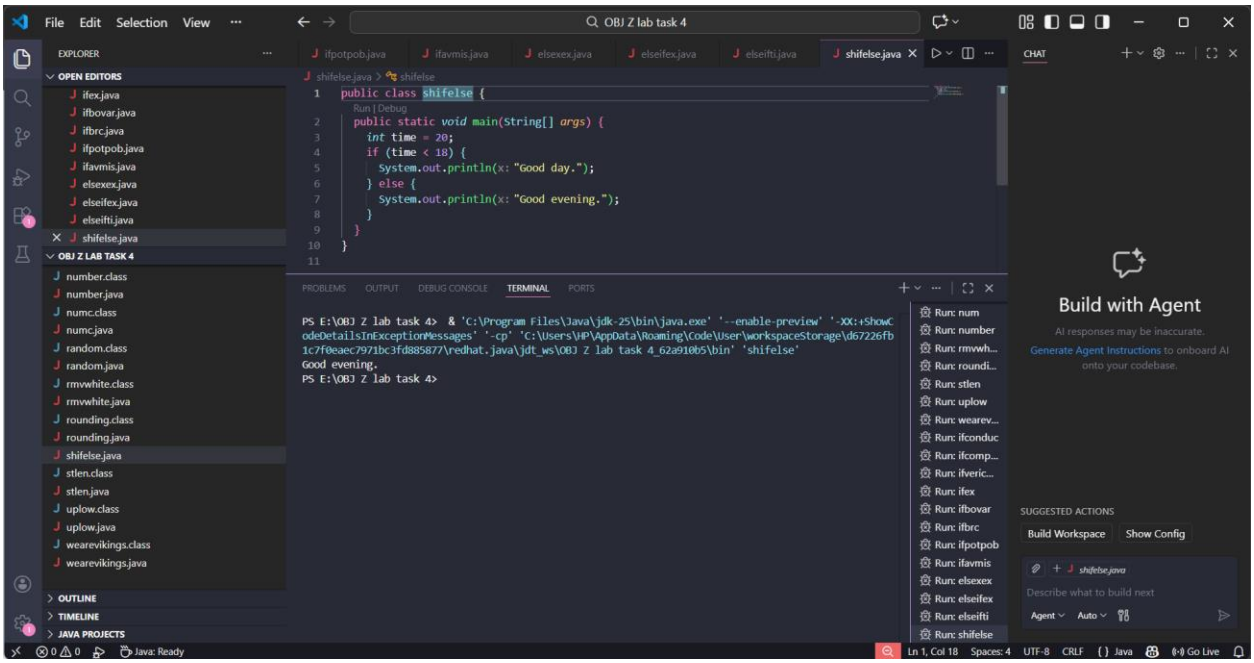
This uses a standard **if-else** statement. Since isRaining is false, it skips the if block and executes the else block, printing that no umbrella is needed.



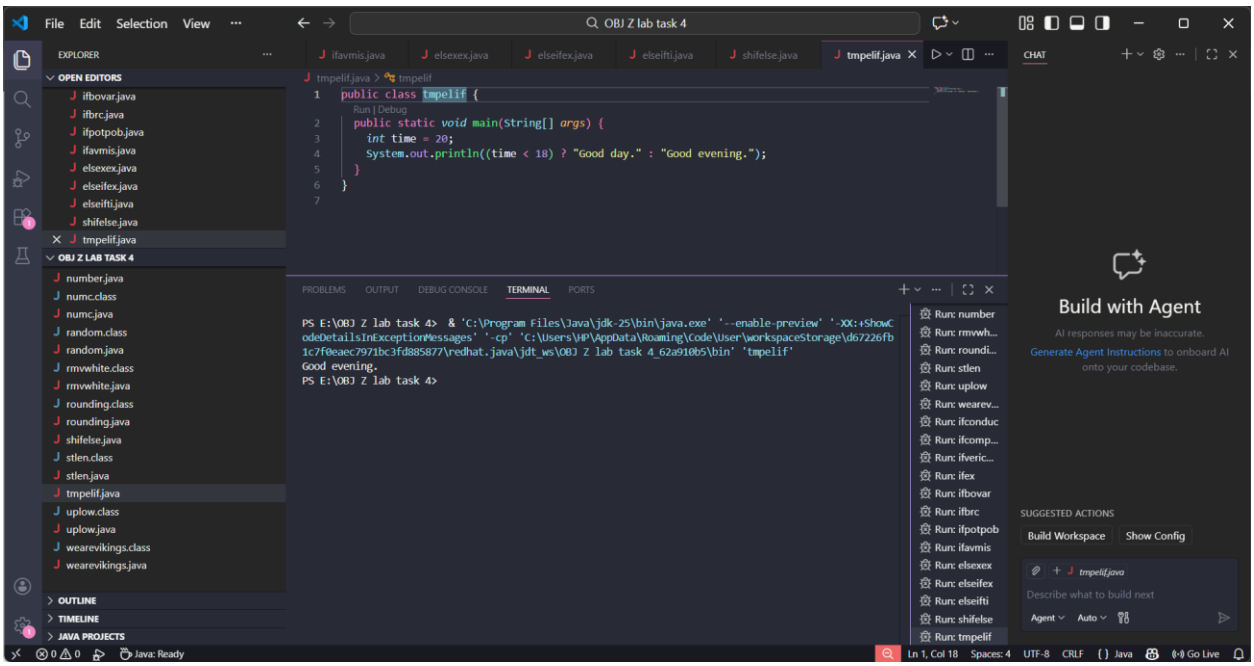
This is another example of an **if-else** structure checking if time (20) is less than 18. Since it is not, it prints "Good evening."



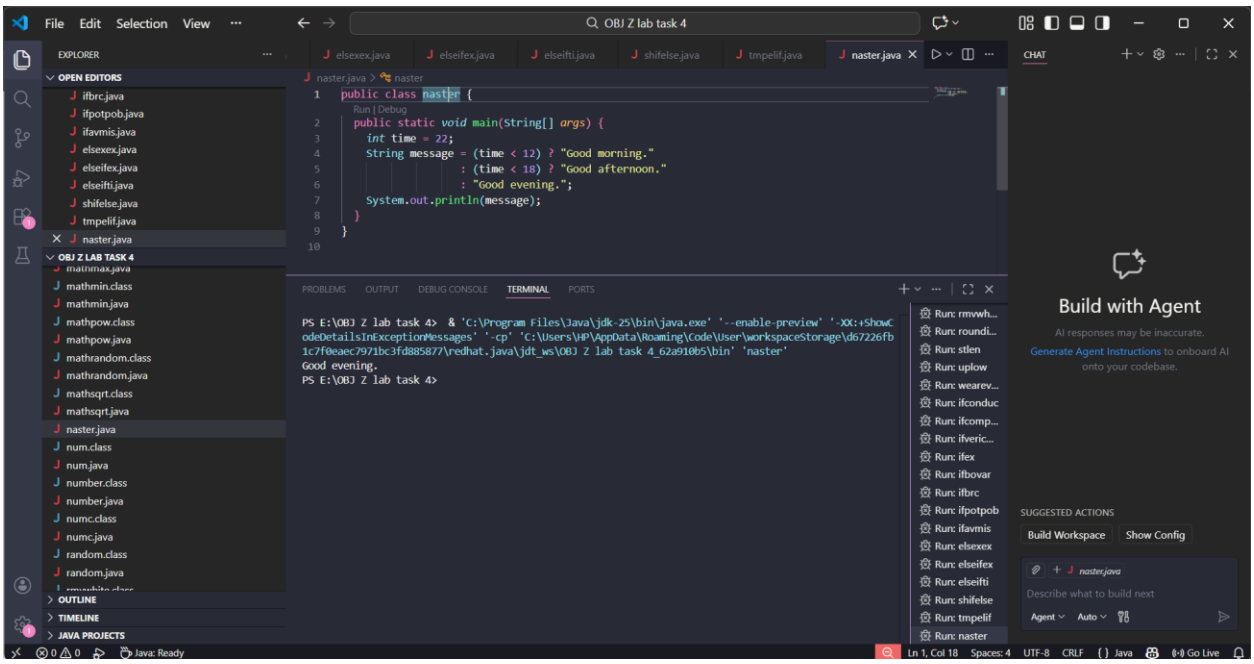
This is another example of an if-else structure checking if time (20) is less than 18. Since it is not, it prints “Good evening.”



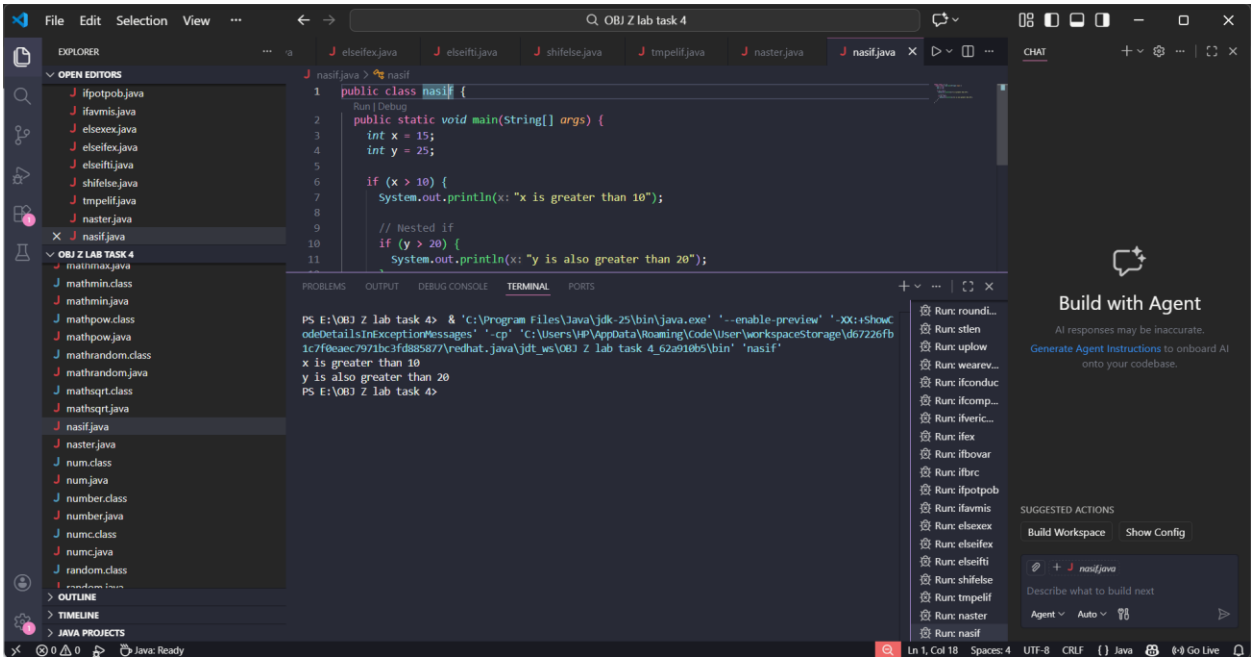
This is another example of an **if-else** structure checking if time (20) is less than 18. Since it is not, it prints "Good evening."



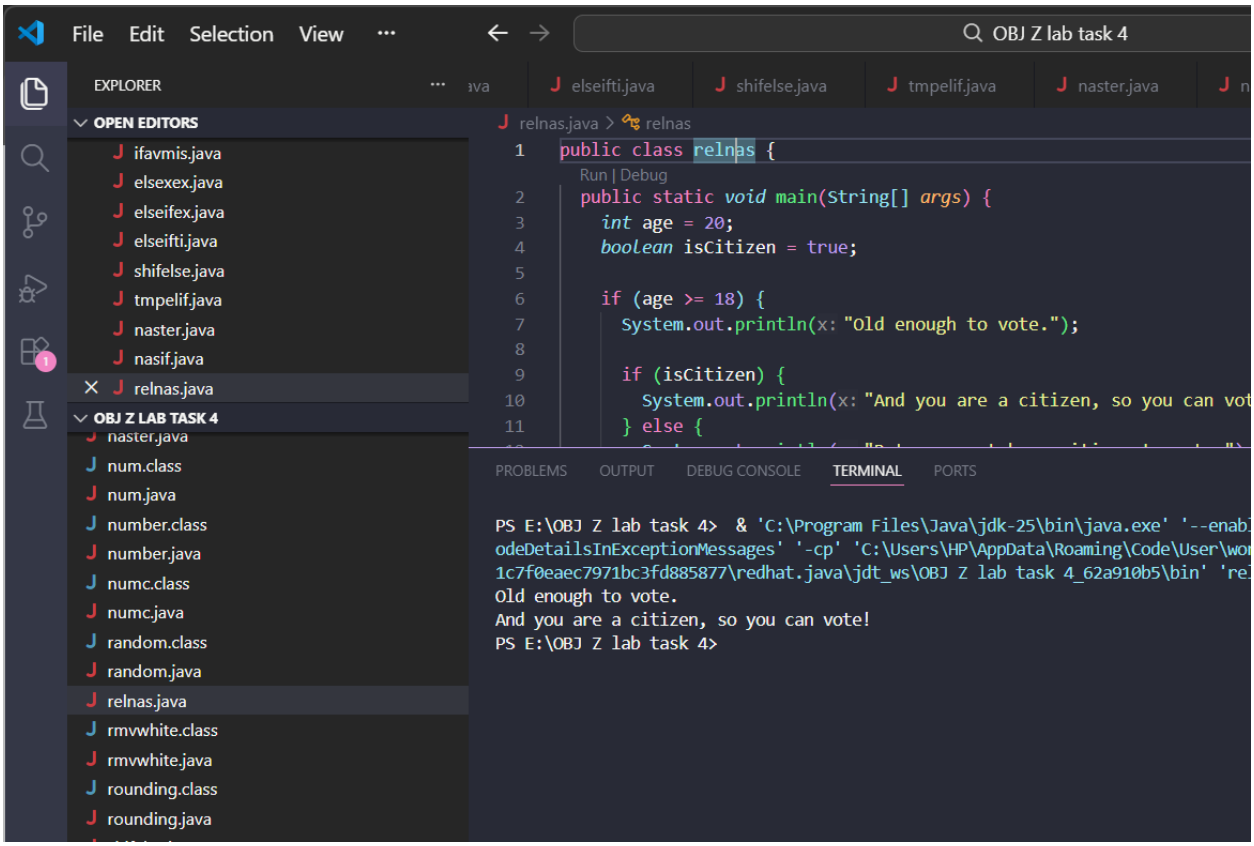
This introduces the **Ternary Operator** (? :), which is a shorthand way of writing an if-else statement in a single line of code.



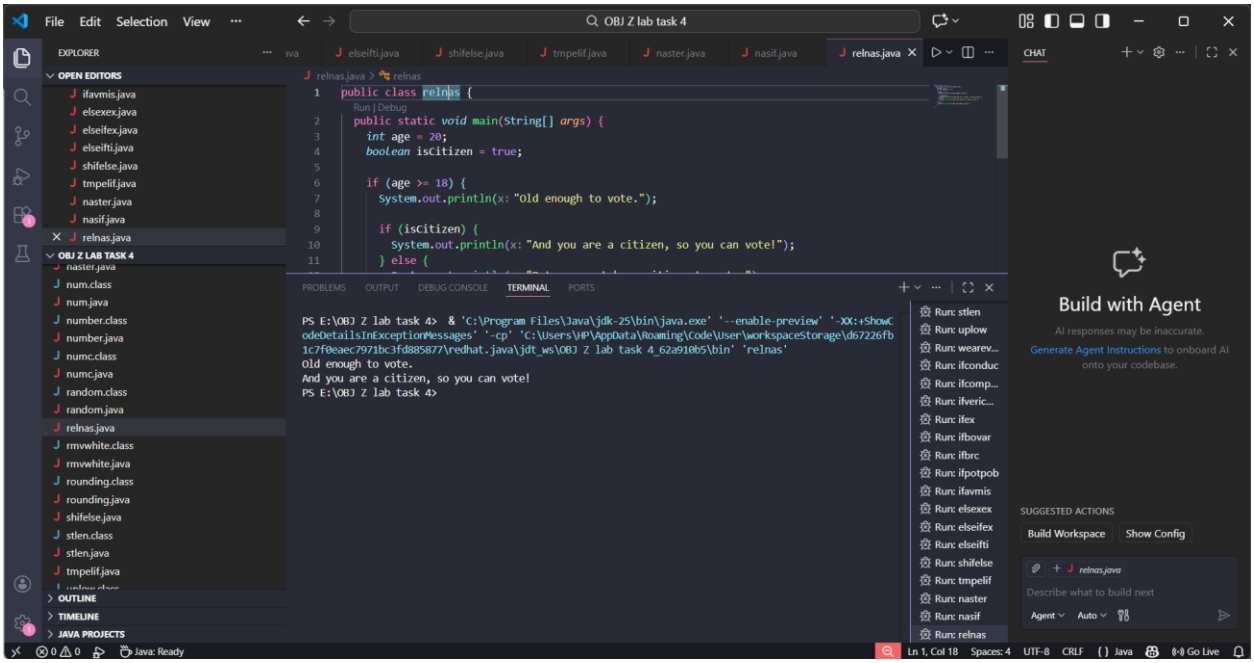
This also uses the **Ternary Operator** to assign a value to the message variable based on the value of time.



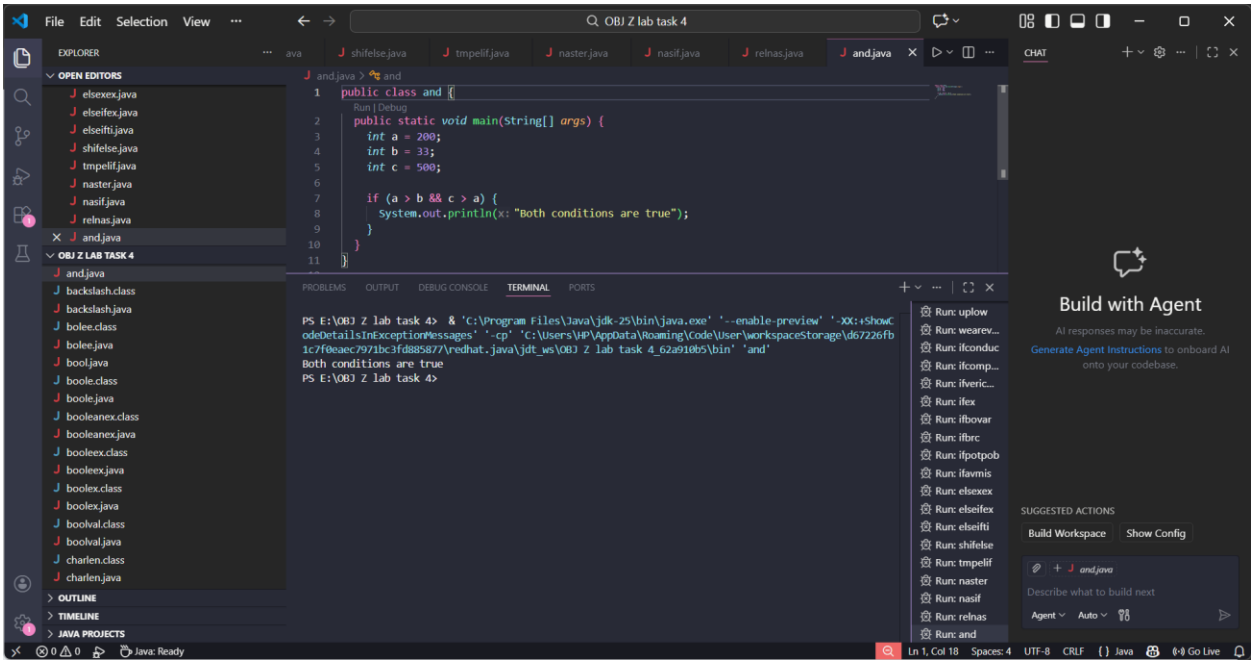
This demonstrates **Nested If** statements (an if inside another if). The inner message is only printed because both conditions (x > 10 and y > 20) are true.



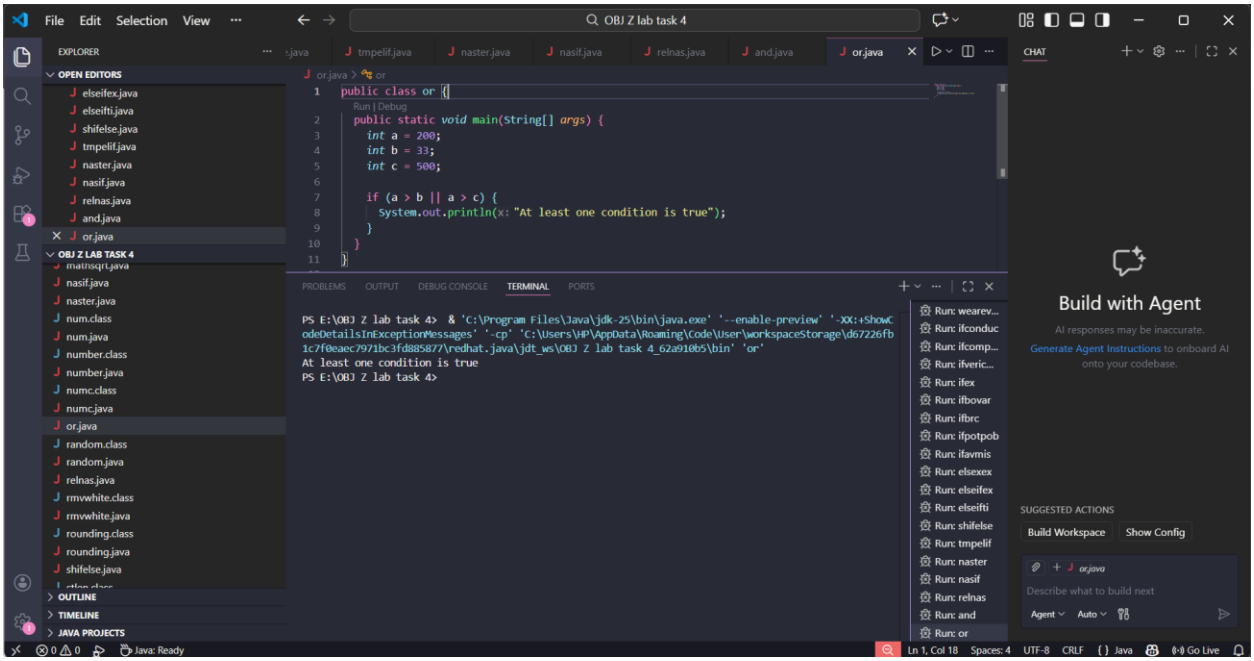
This combines relational operators (>=) with boolean variables. It checks if the user is old enough AND if they are a citizen before allowing them to vote.



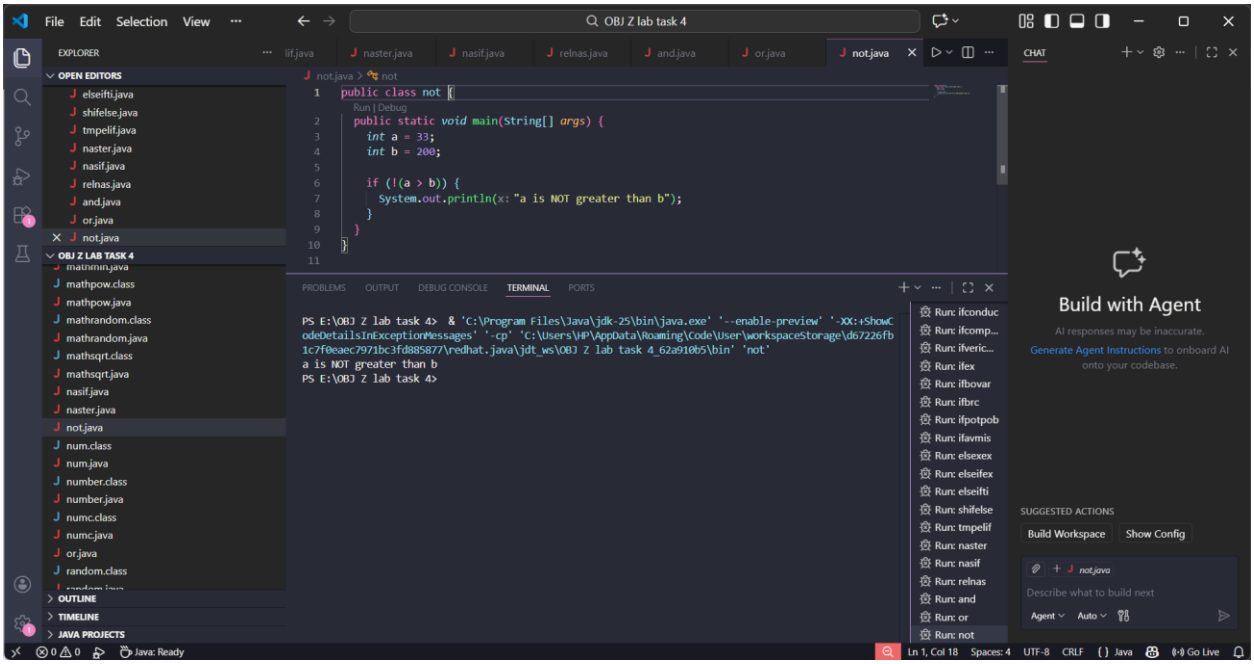
- This uses the **AND operator** (&&). The program prints "Both conditions are true" only because a is greater than b AND a is greater than c.



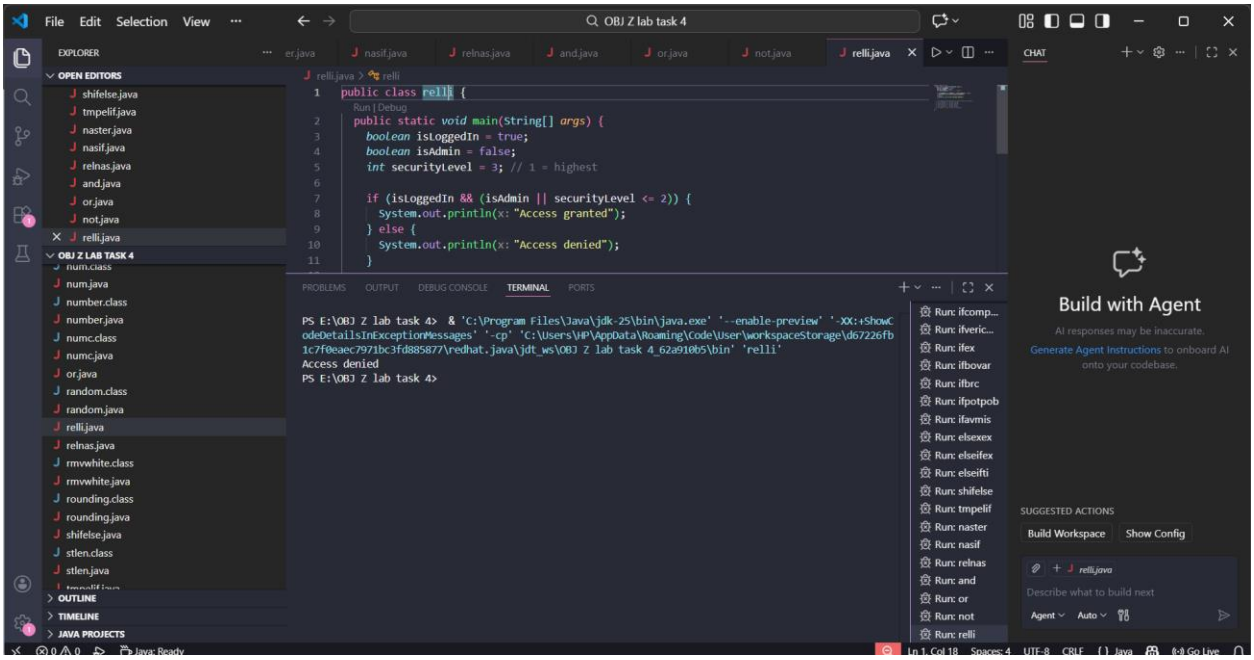
This uses the OR operator (||). It returns true if at least one of the conditions is true.



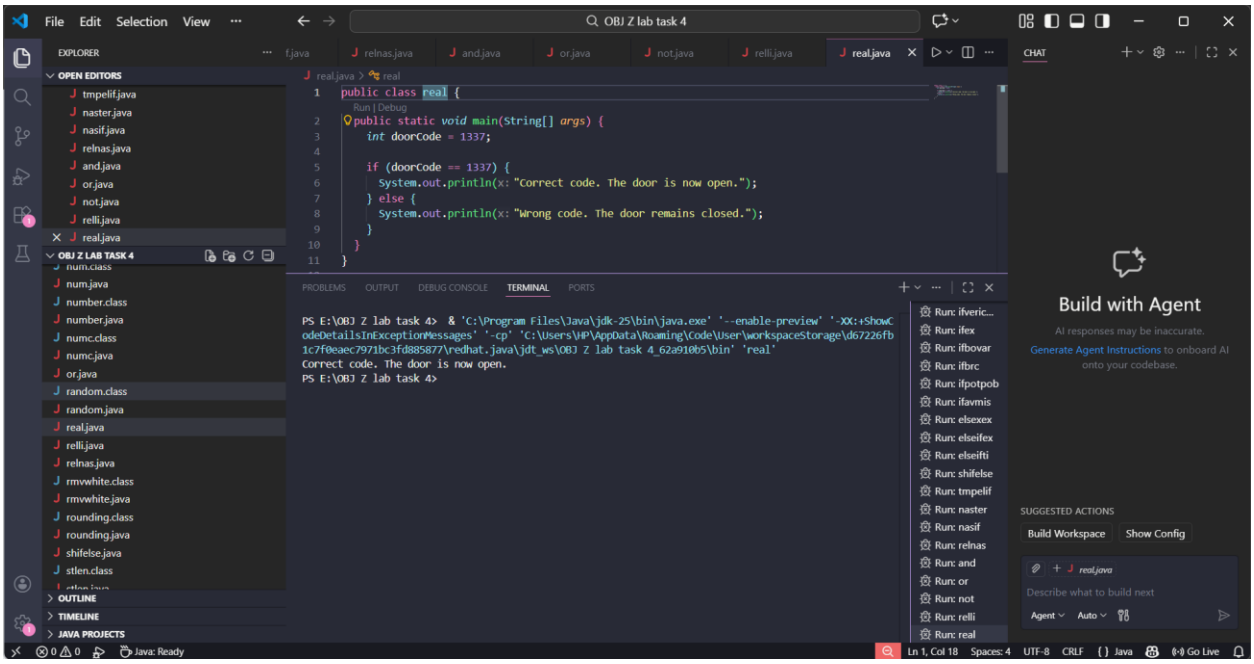
This uses the NOT operator (!). It reverses the result, so checking !(a > b) essentially checks if a is NOT greater than b.



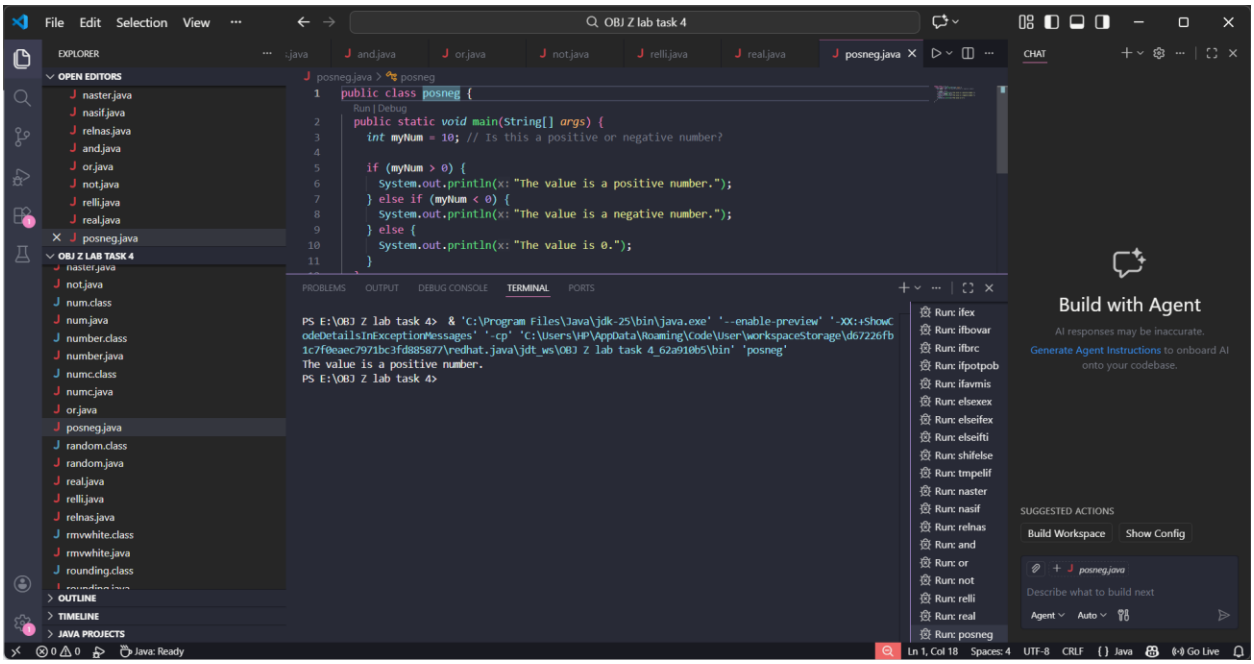
This is a complex logical check. It uses && (AND) and || (OR) to verify if a user is logged in AND has either admin privileges or a high security level.



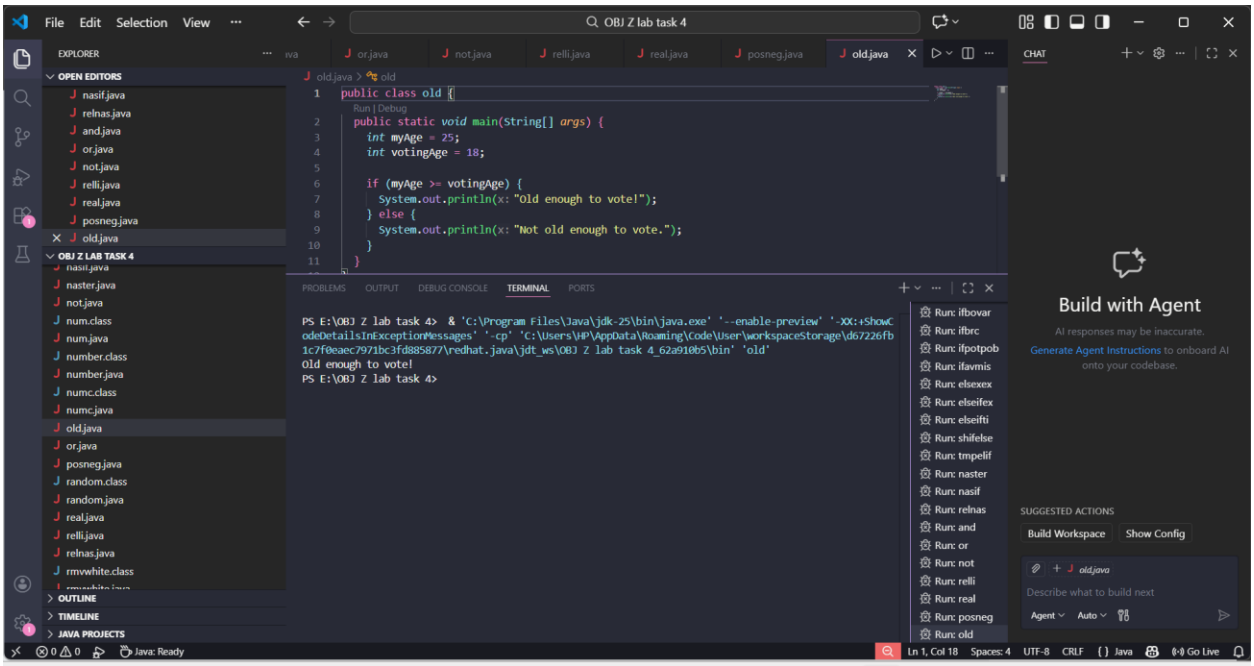
A practical example of an equality check (==). It acts like a password system, checking if the doorCode matches 1337 to open the door.



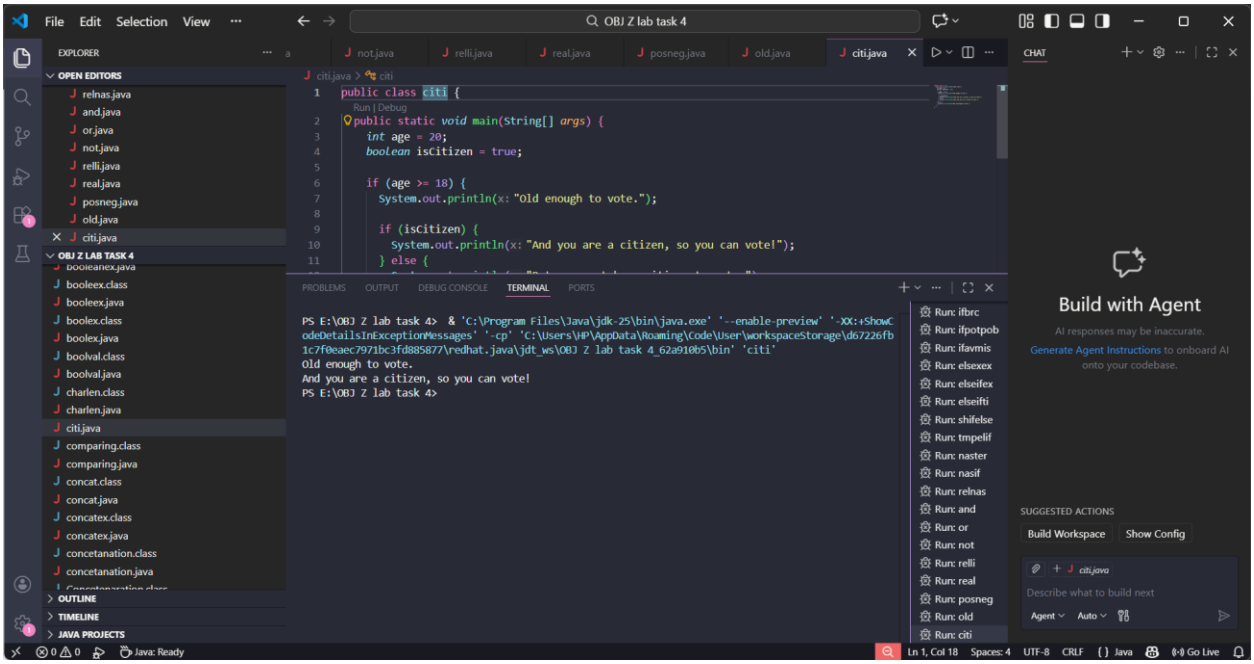
This uses an if-else ladder to determine if a number is Positive, Negative, or Zero.



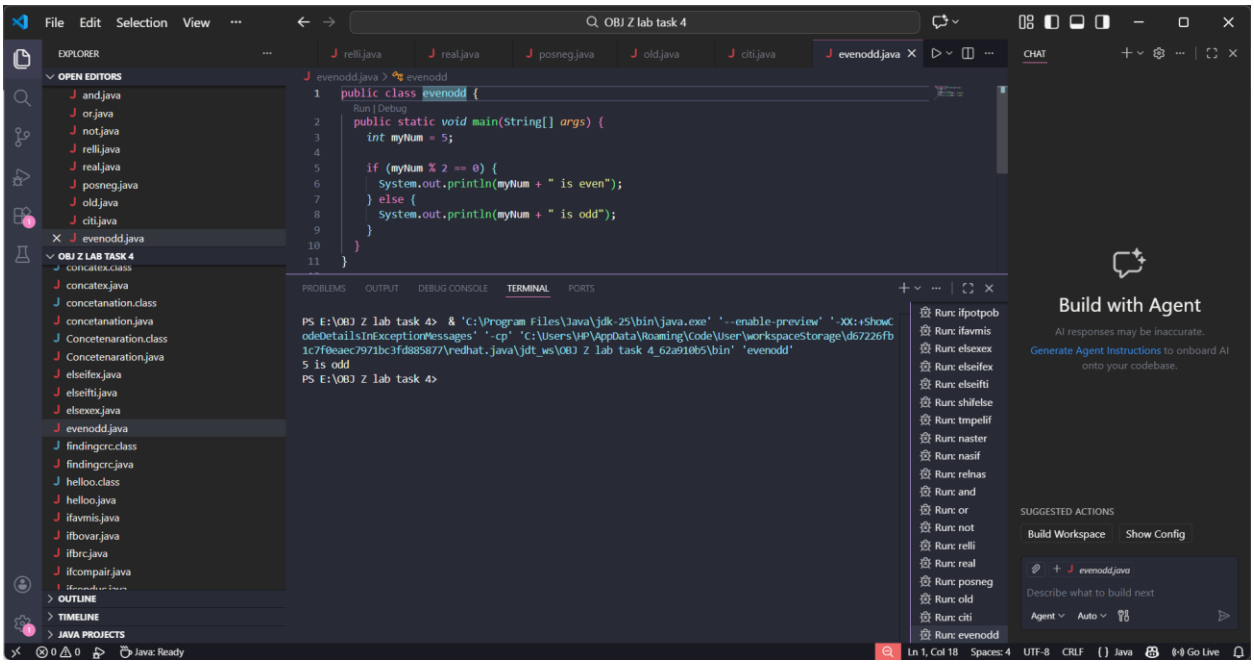
Checks if a person is old enough to vote by comparing myAge against votingAge.



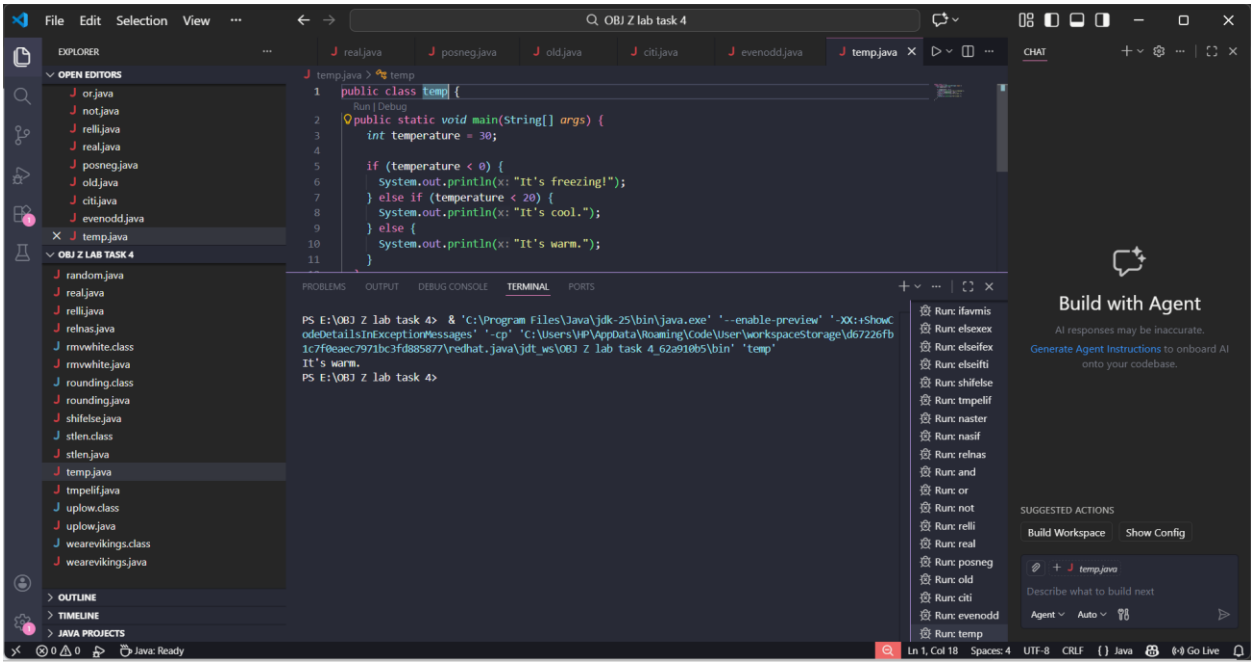
A verification script that checks two requirements: the user must be 18+ AND isCitizen must be true.



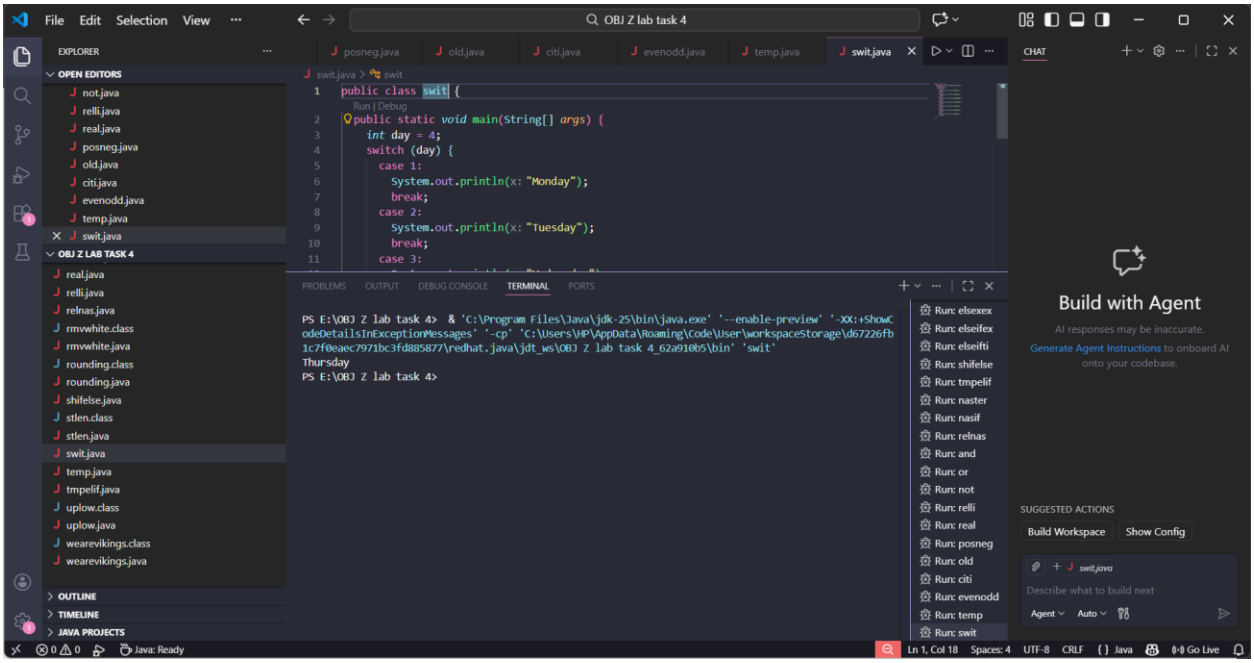
A verification script that checks two requirements: the user must be 18+ AND isCitizen must be true.



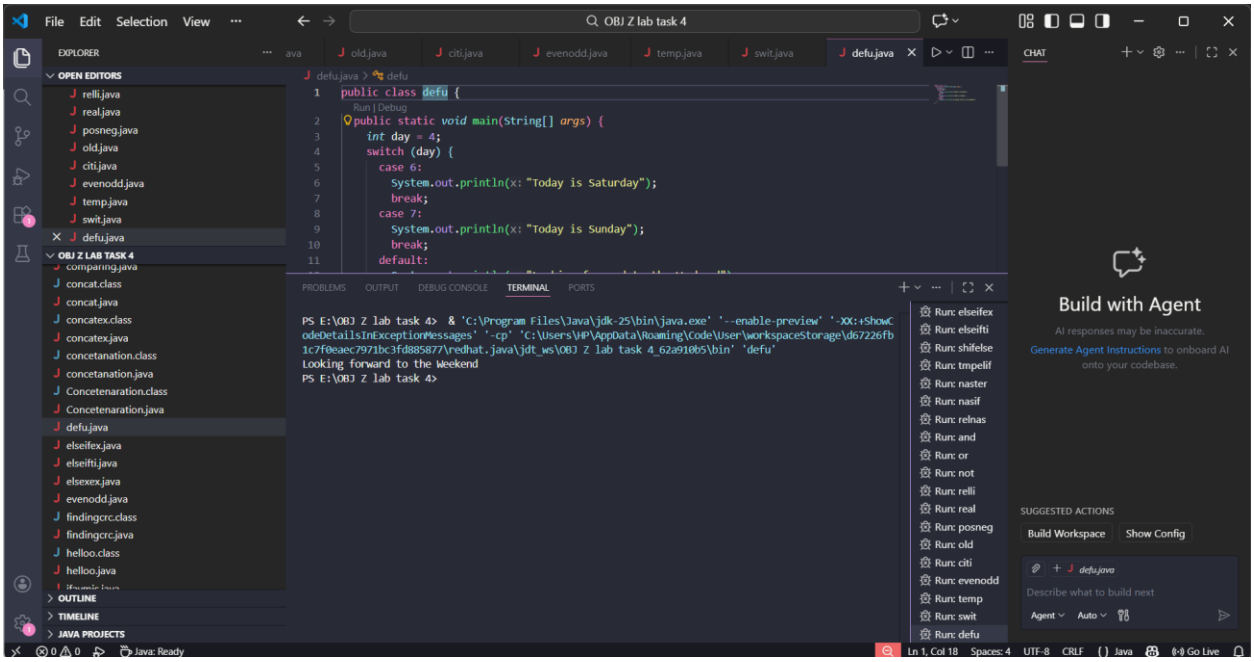
This uses the Modulo Operator (%). myNum % 2 == 0 checks if a number divides by 2 with no remainder, which is the standard way to check for Even numbers



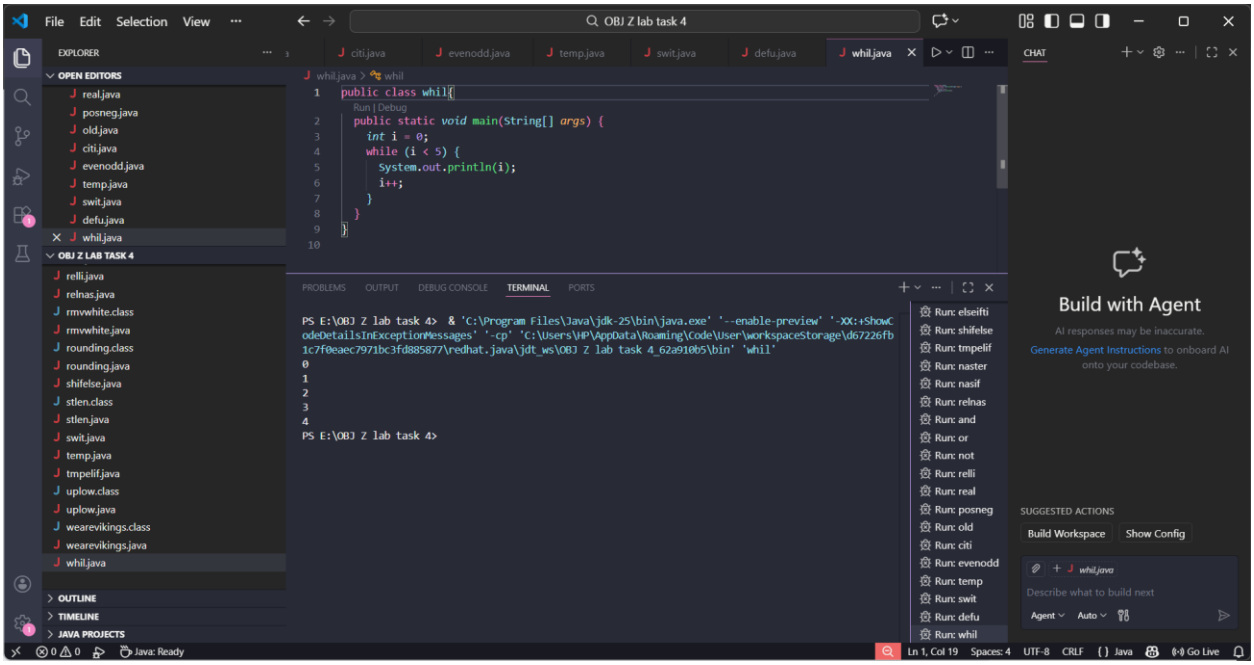
A weather program that checks ranges of numbers to classify the temperature as freezing, cool, or warm.



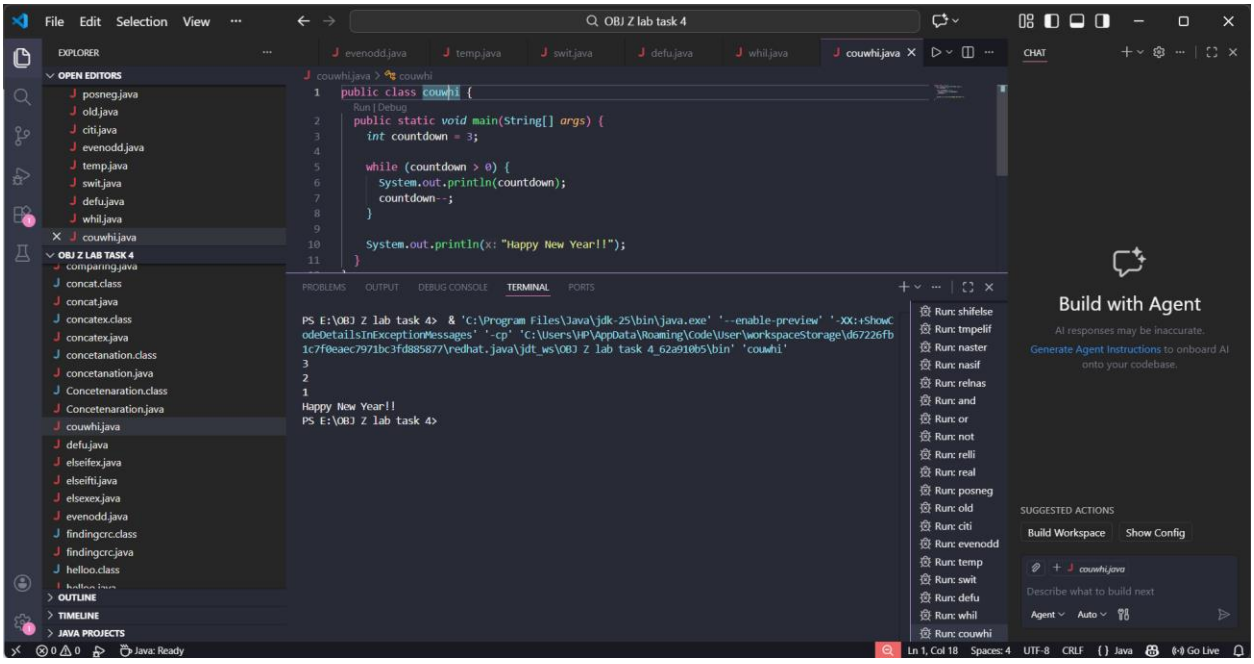
This uses a Switch Statement to check the value of day (4) and print the corresponding day name (Thursday).



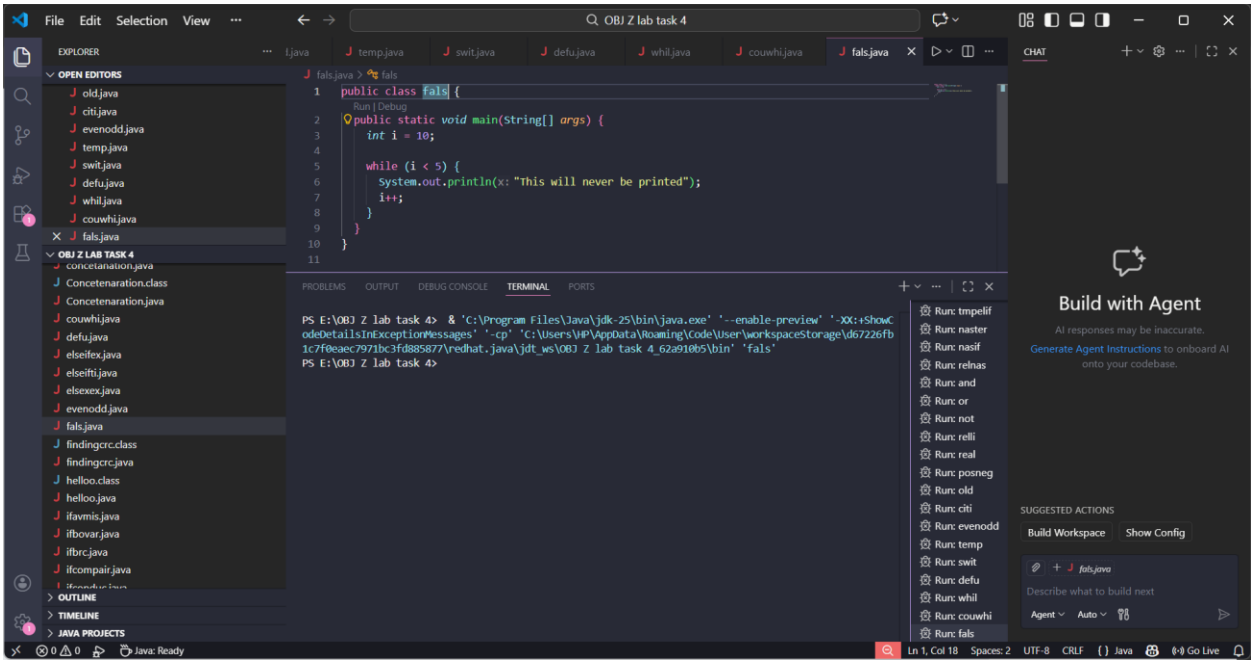
A countdown program using a while loop. It prints numbers starting from 3 and decreases them (countdown--) until it hits zero.



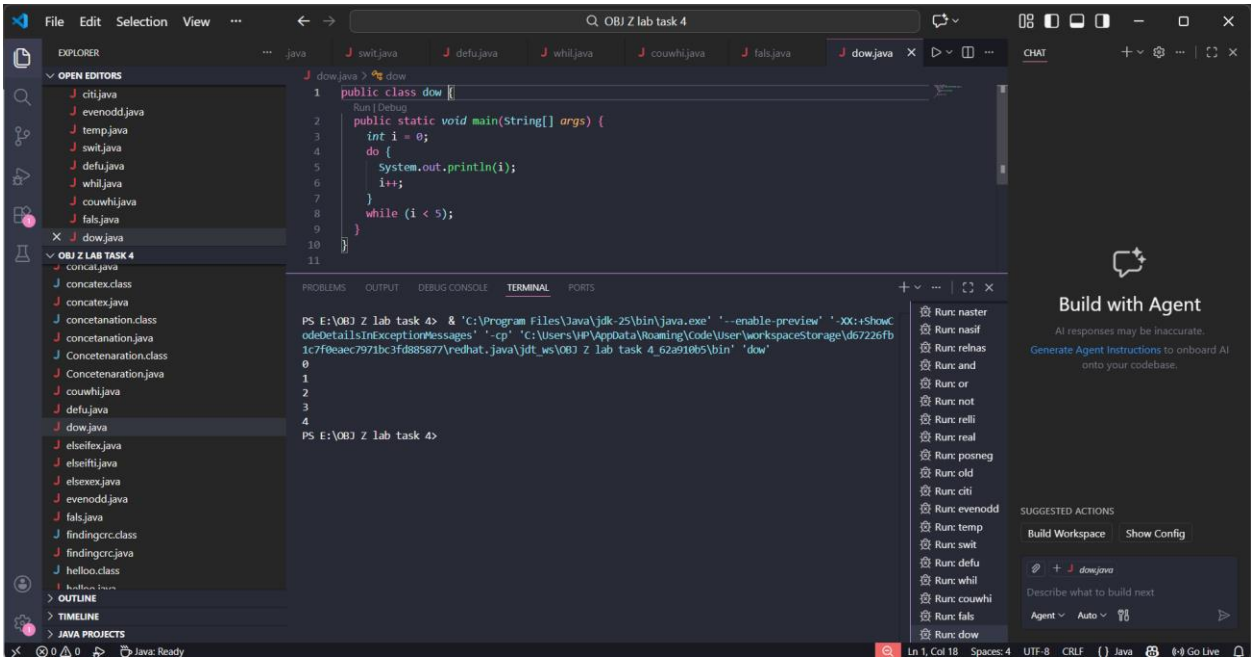
A standard While Loop. It continues to print and increment I as long as I is less than 5.



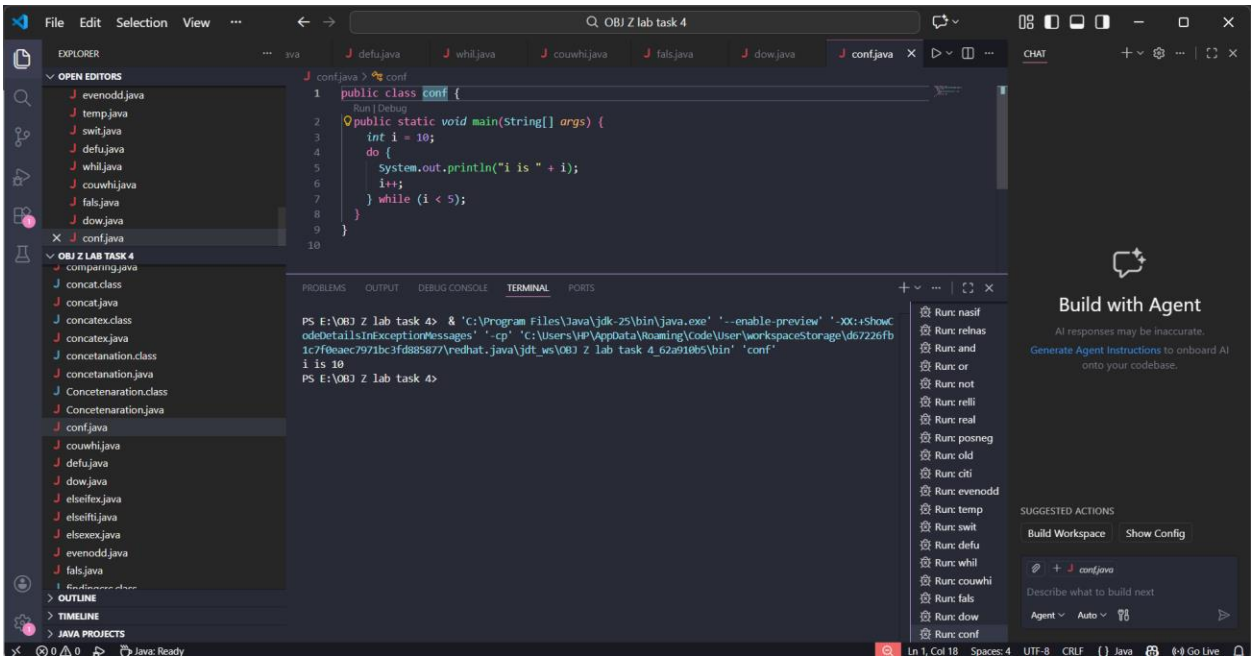
A countdown program using a while loop. It prints numbers starting from 3 and decreases them (countdown--) until it hits zero.



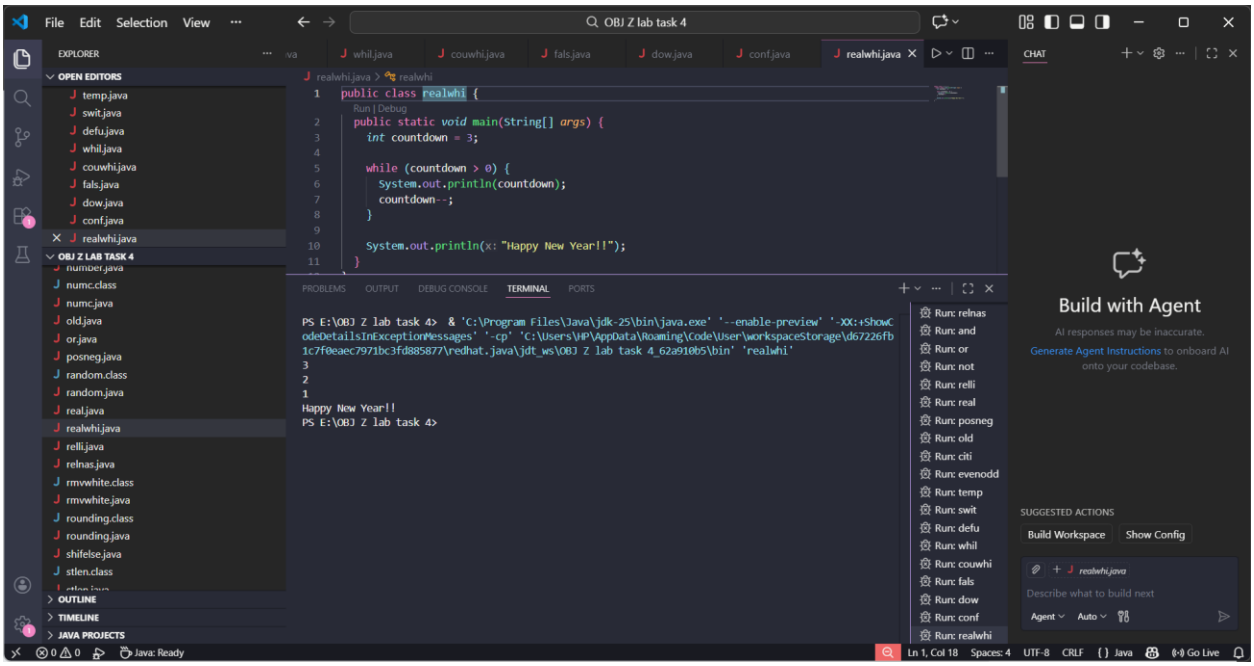
Demonstrates that a While Loop checks the condition before running. Since I (10) is not less than 5, the code inside the loop never runs.



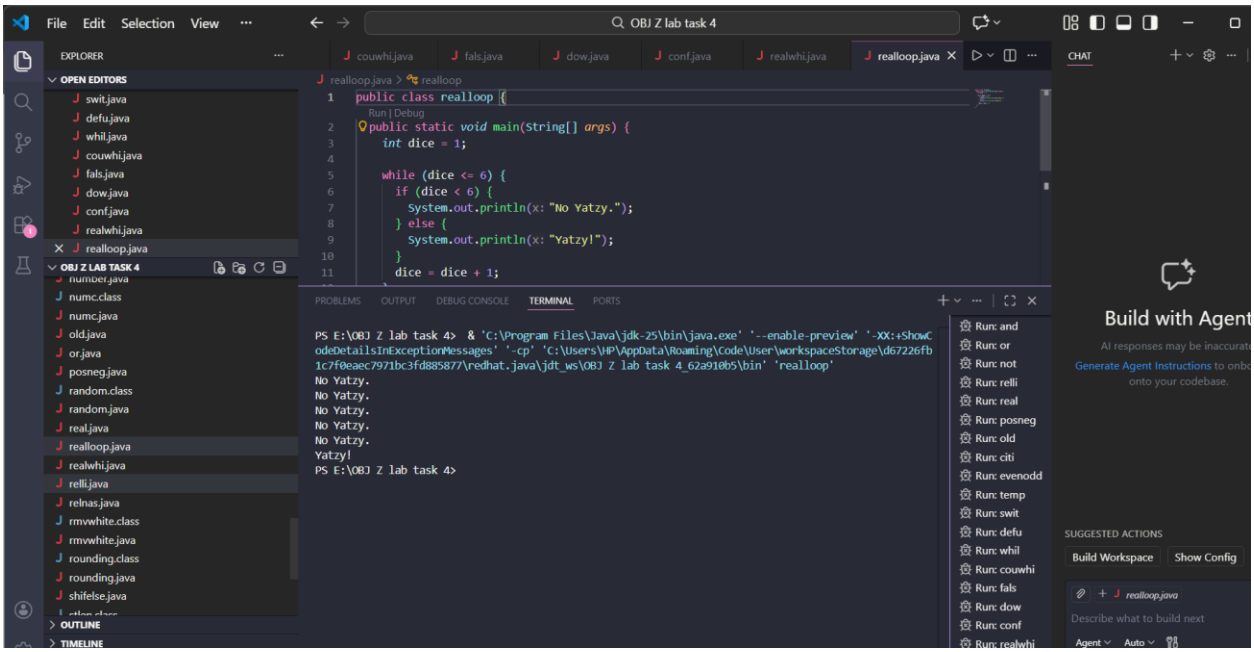
- A **Do-While Loop**. This loop guarantees the code runs at least one time before checking the condition.



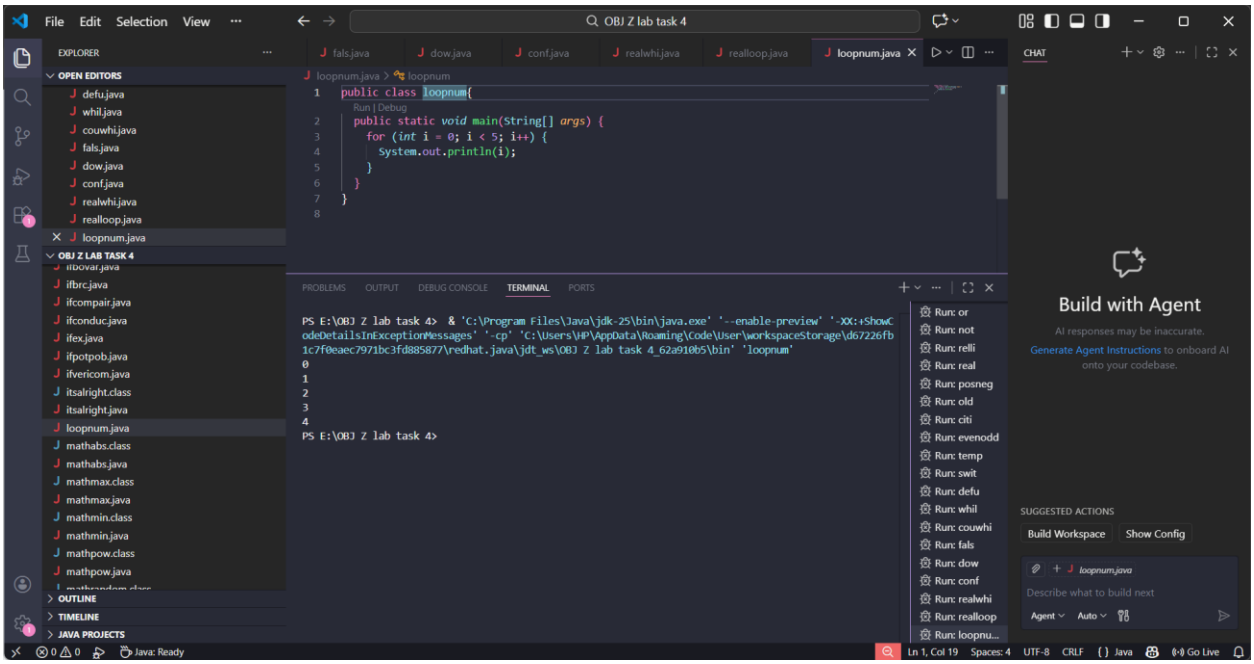
Another Do-While example showing that the loop executes the print statement and increment before verifying if I < 5.



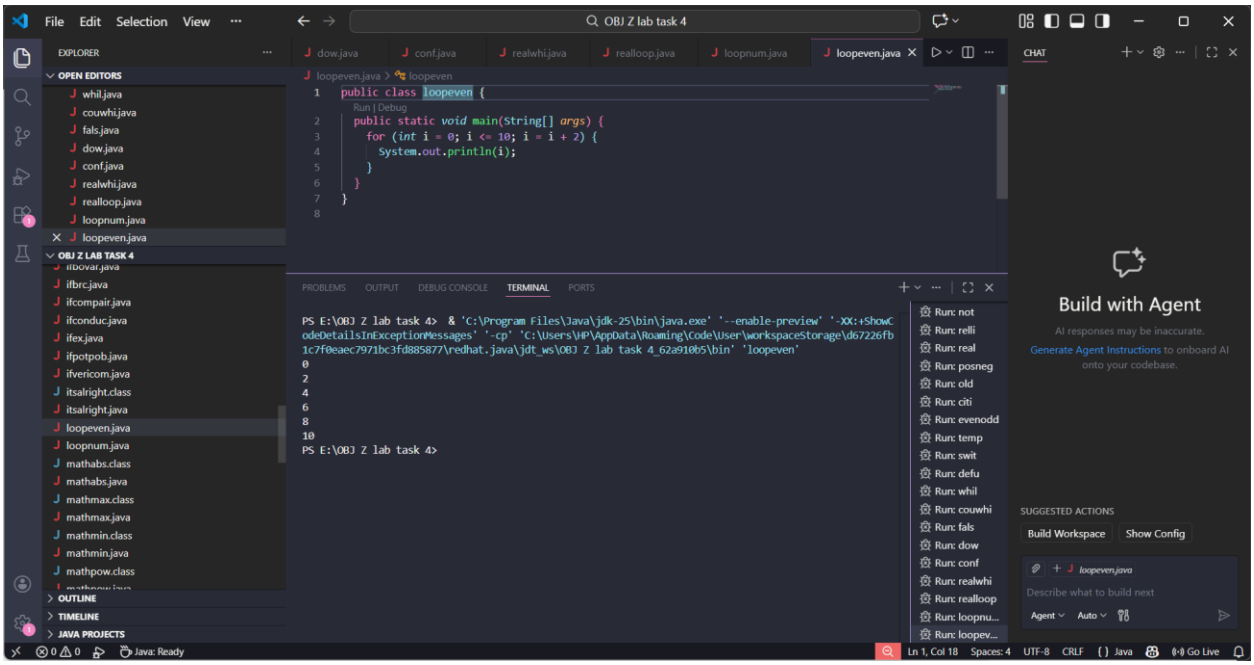
- A practical loop that keeps running while countdown > 0, simulating a timer before printing "Happy New Year!!".



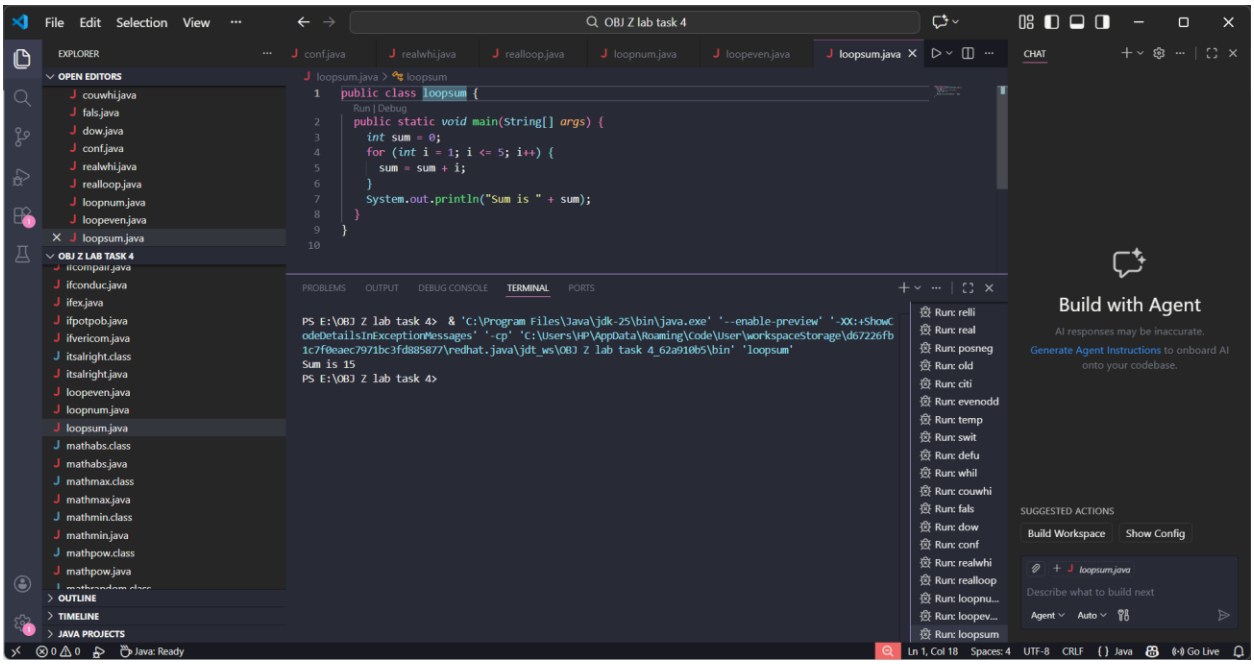
- A simulation of rolling dice until a specific number (Yatzy/6) is reached using a while loop.



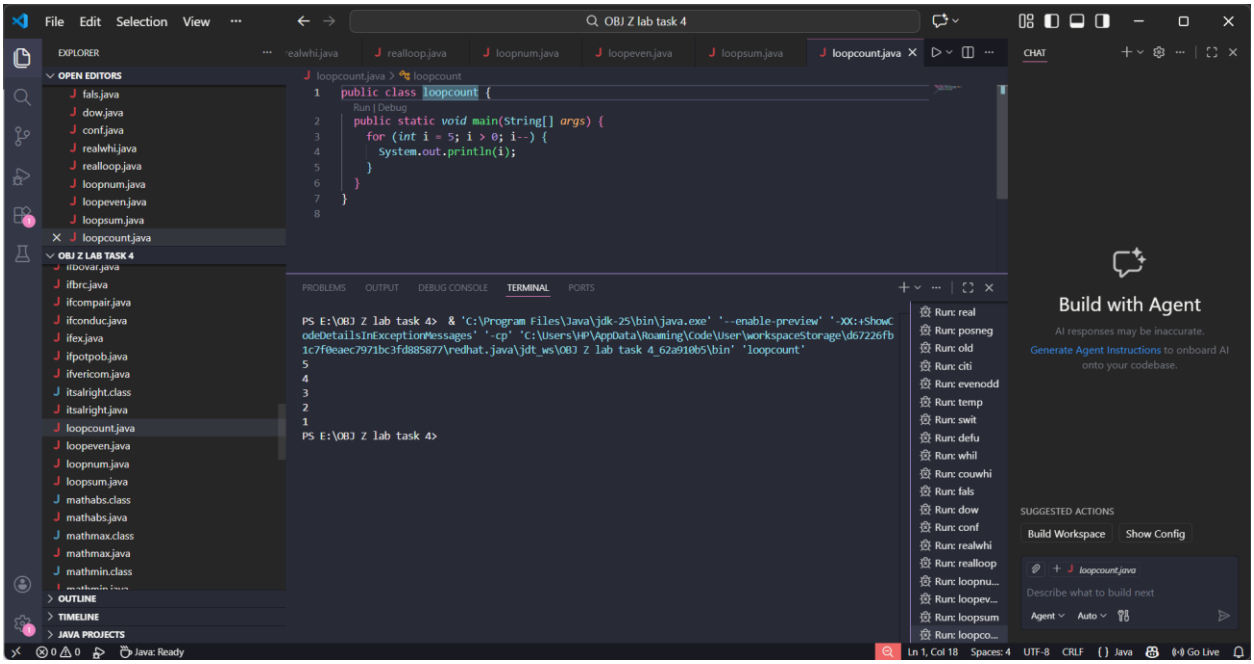
- A standard **For Loop**. It initializes i to 0 and runs as long as i < 5, printing the number each time.



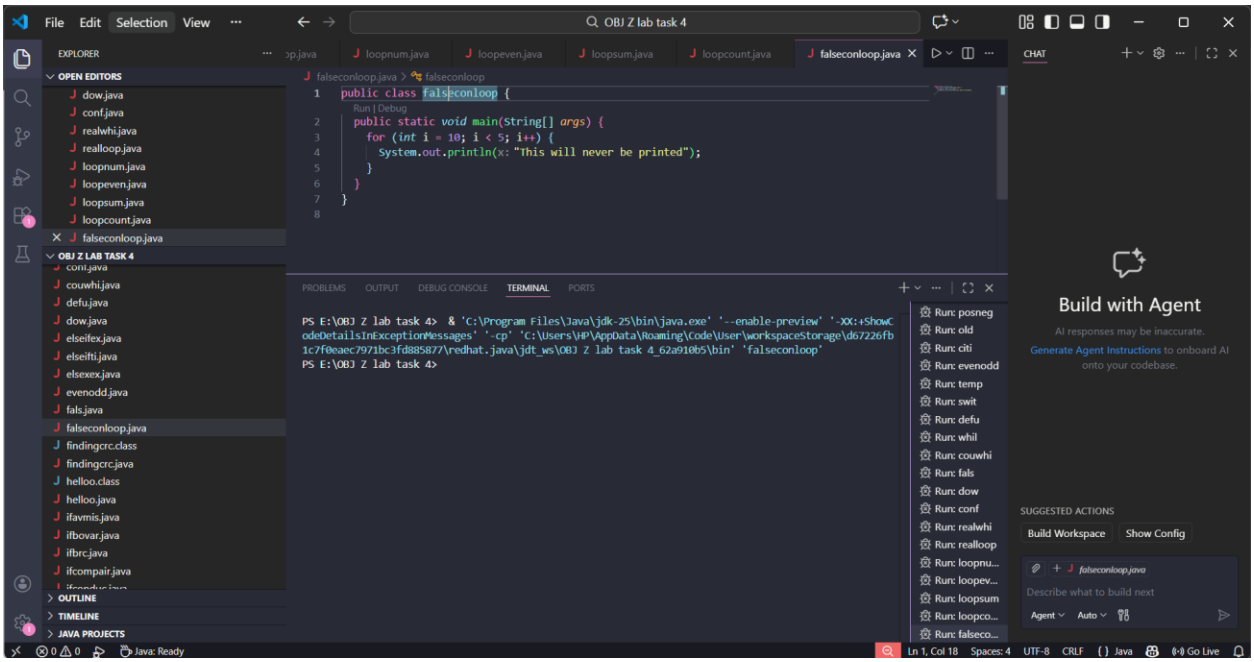
- A for loop that prints only **Even Numbers** between 0 and 10 by incrementing i by 2 (i = i + 2) in each step.



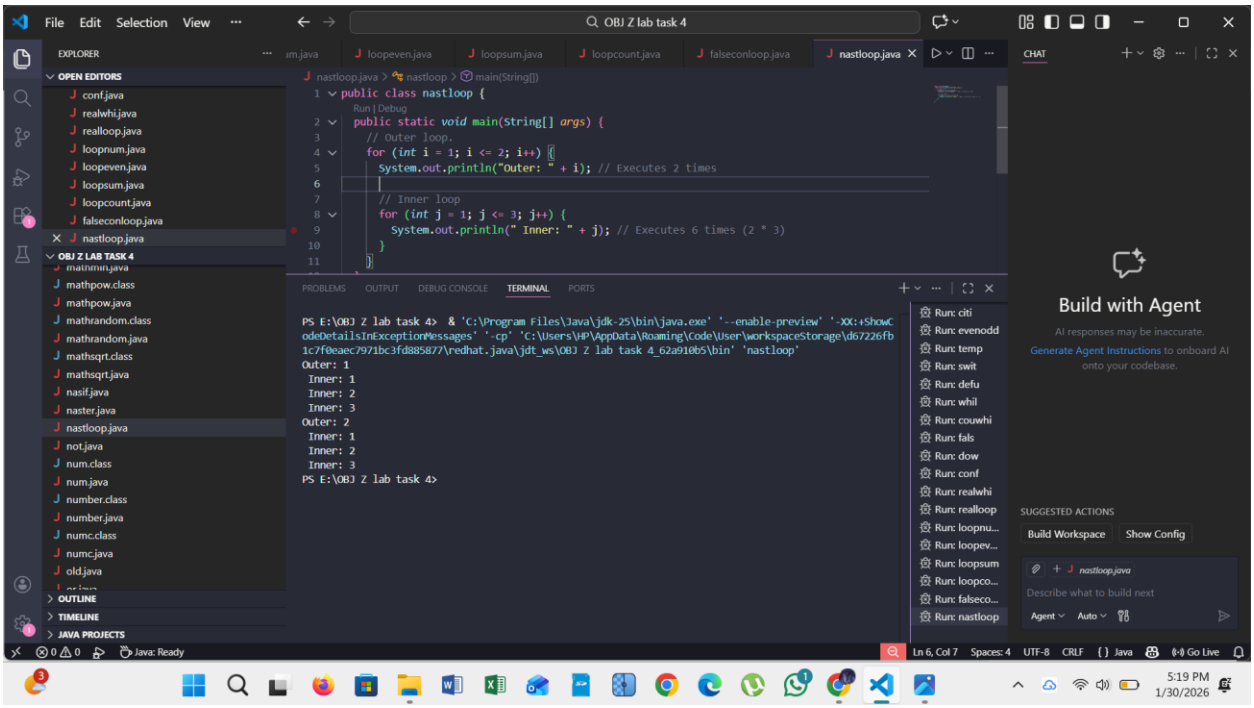
- Uses a for loop to calculate the **Sum** of numbers 1 to 5. It adds the value of i to the sum variable in every iteration.



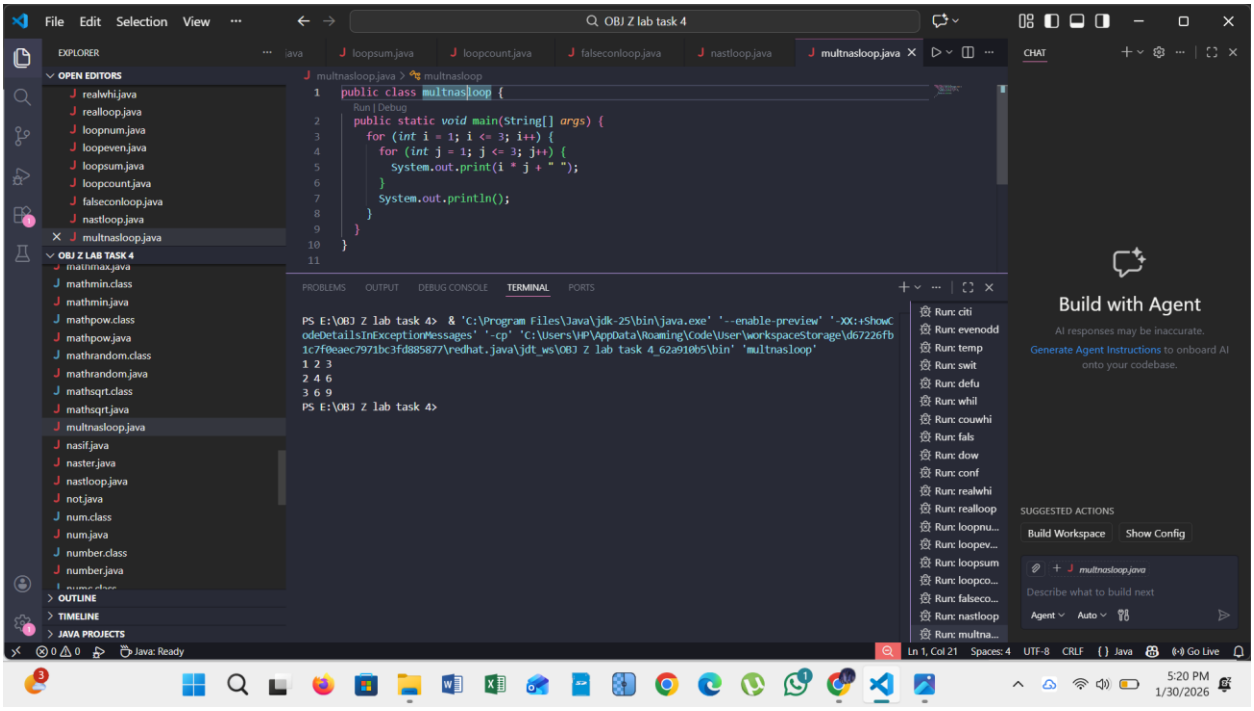
- A **decrementing** for loop. It starts at 5 and counts down to 1 using i--.



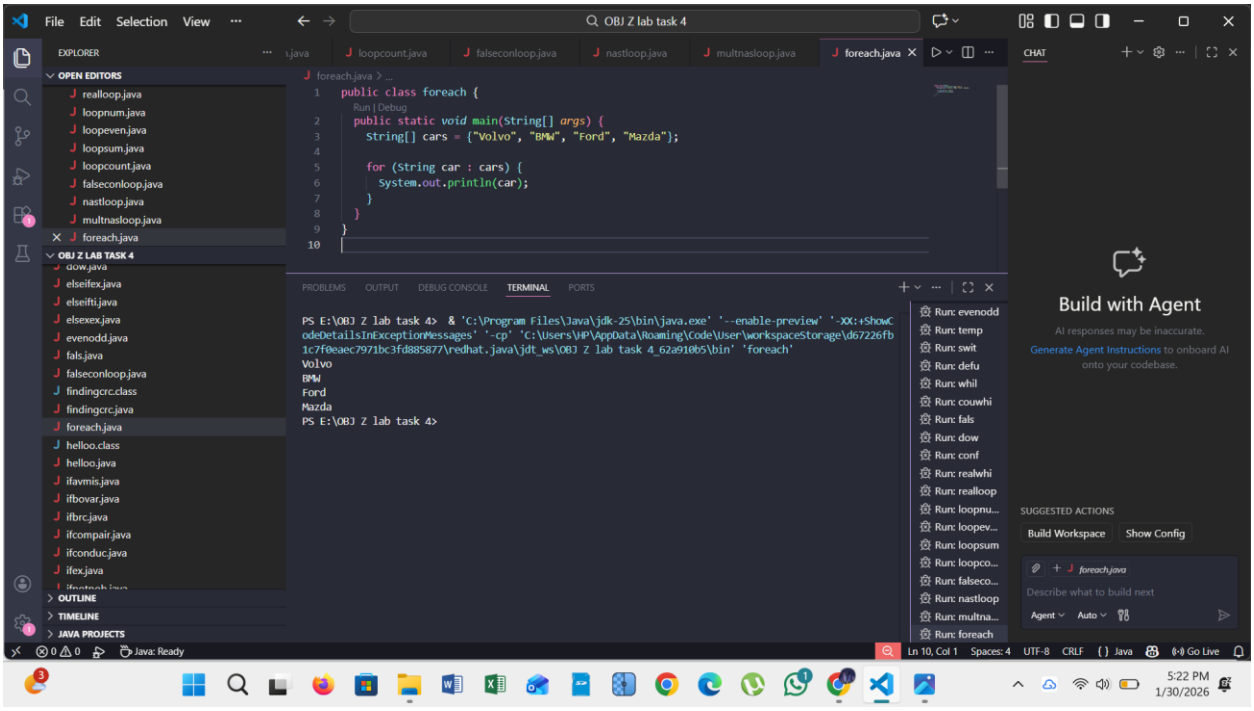
- Shows a loop with a condition that is false from the start (i < 5 when i is 10), so the loop body never executes.



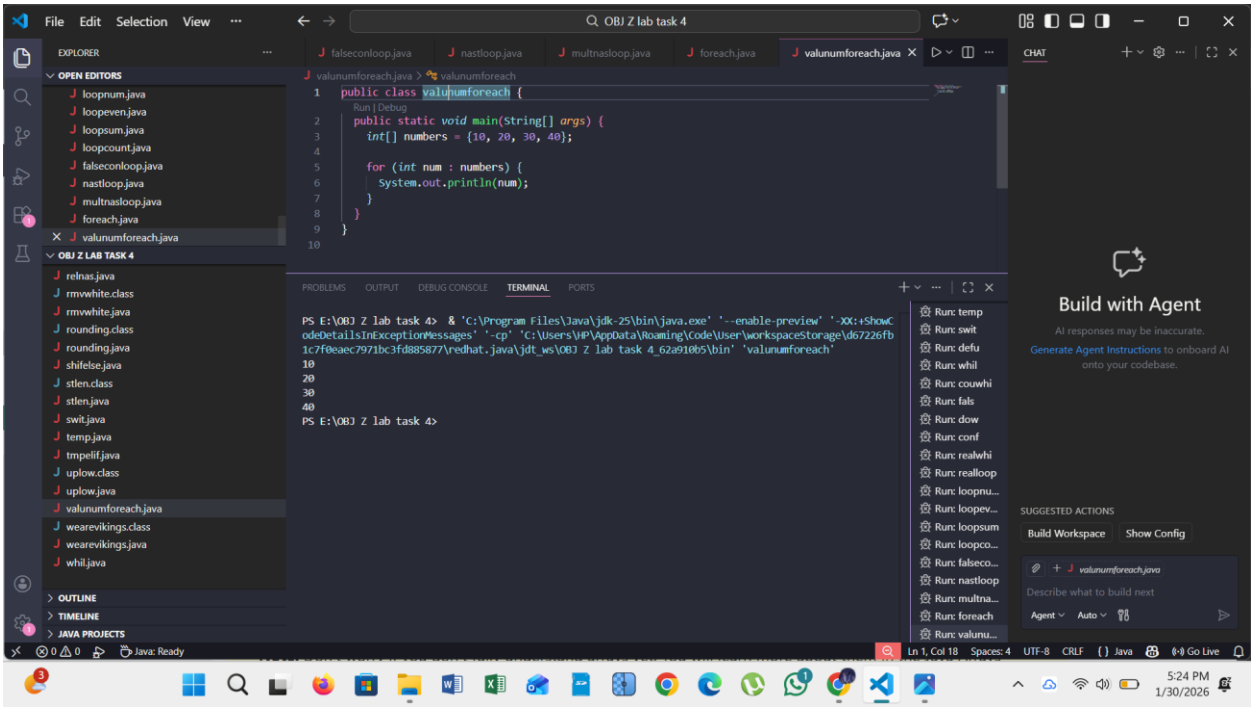
- A **Nested Loop** (a loop inside a loop). The inner loop runs completely (3 times) for every single iteration of the outer loop.



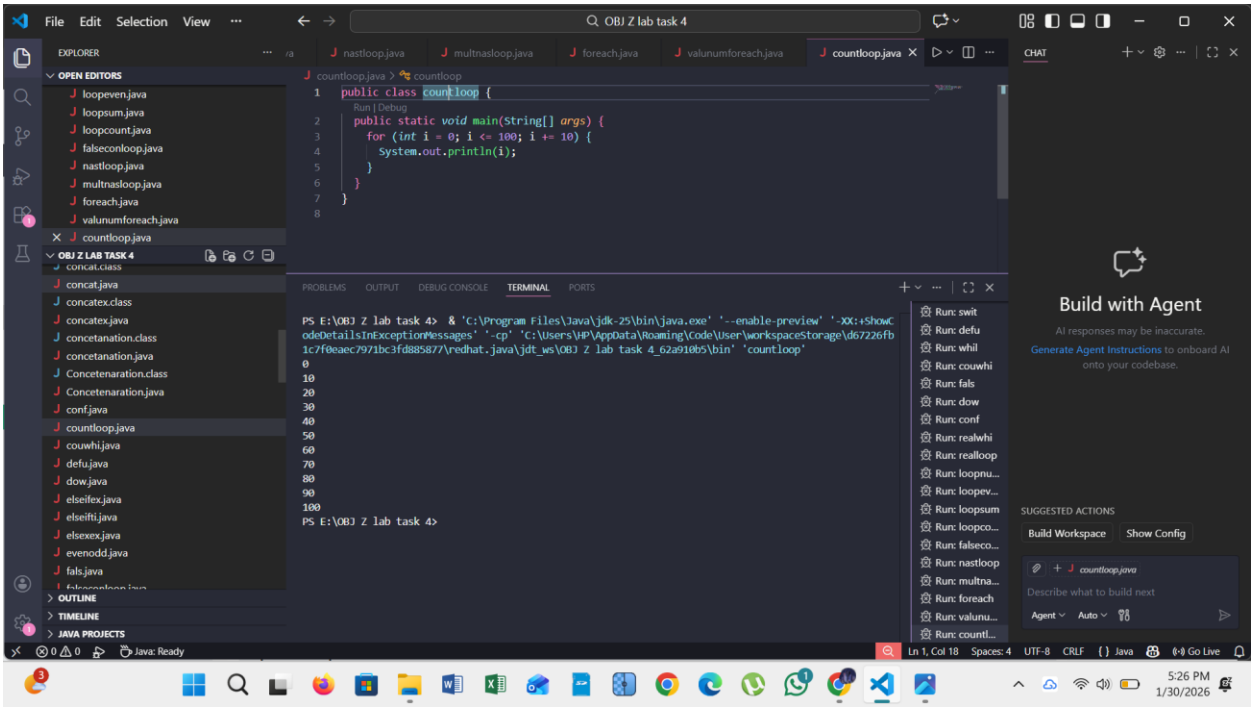
- Uses nested loops to print a number pattern, showing how inner loops work for grid-like logic.



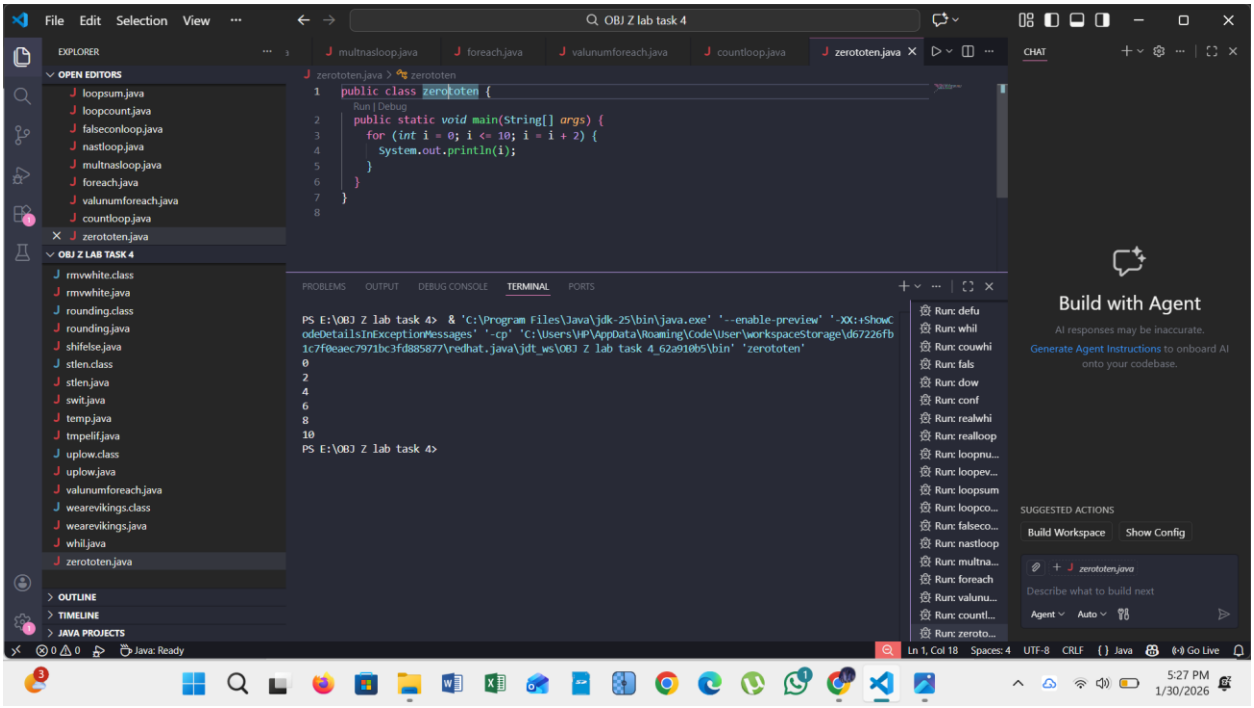
- These demonstrate the **For-Each Loop**. This is a special loop used to iterate through every item in an array (like cars or numbers) without needing an index counter.



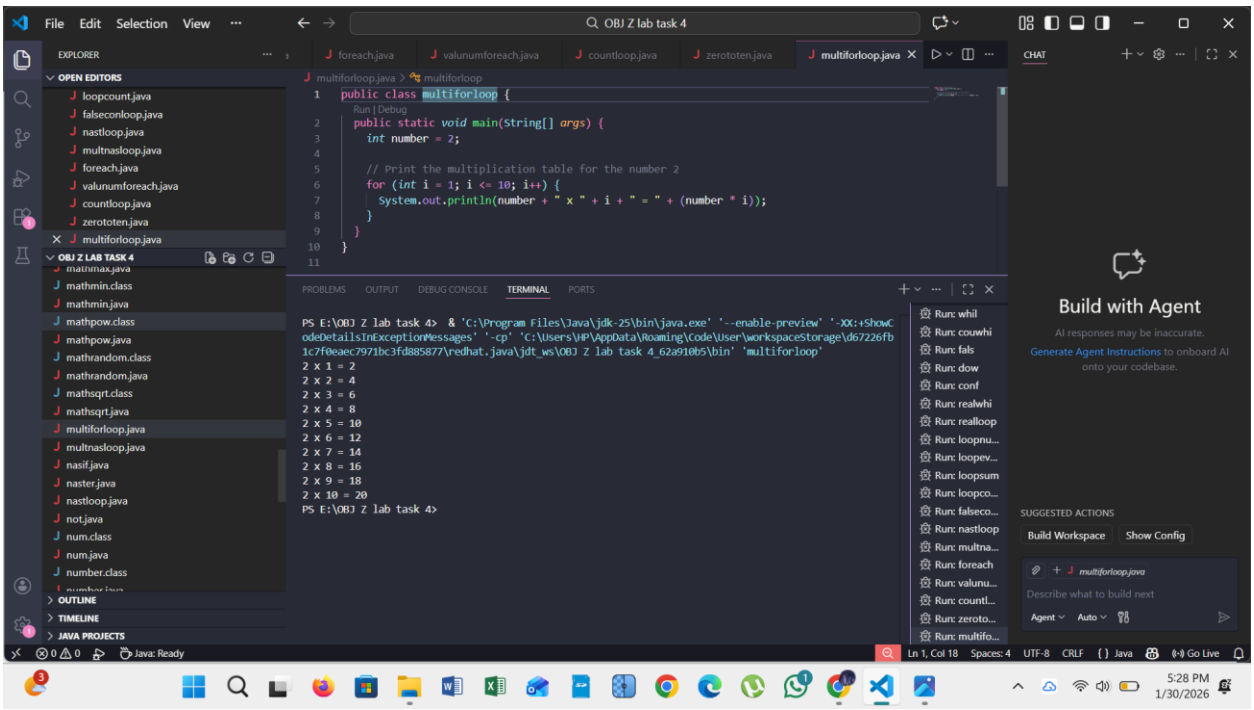
- These demonstrate the **For-Each Loop**. This is a special loop used to iterate through every item in an array (like cars or numbers) without needing an index counter.



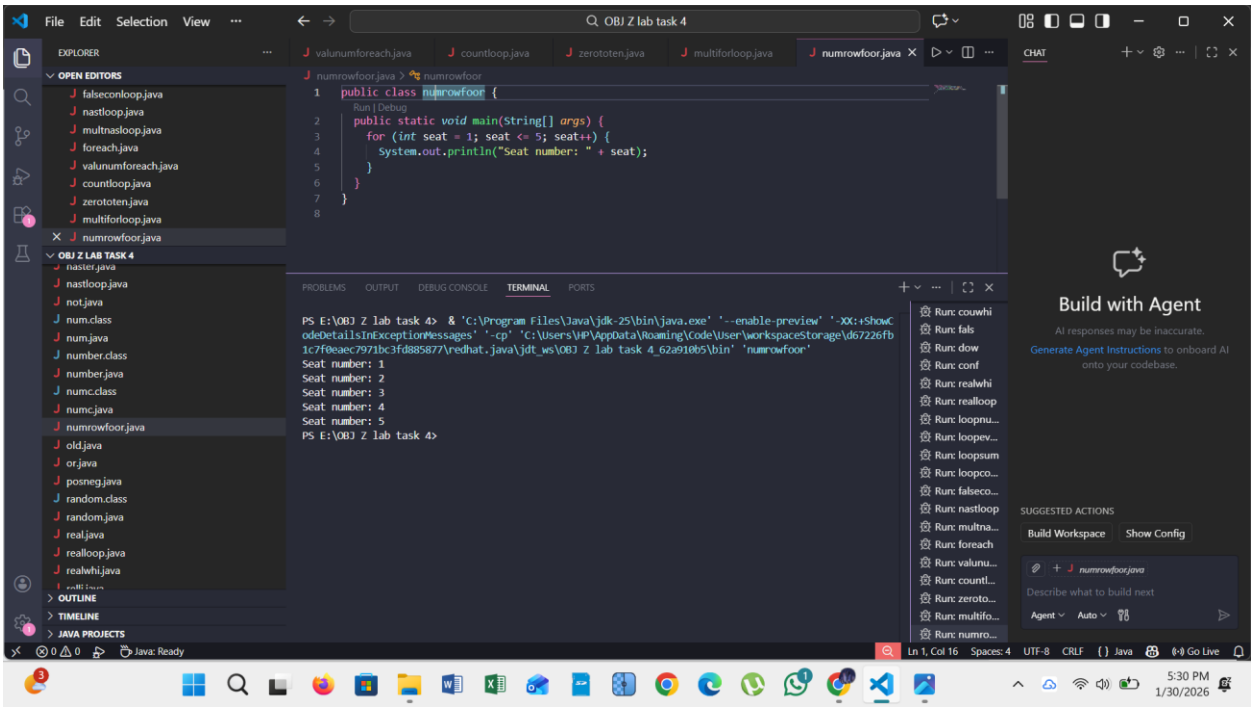
A for loop that counts from 0 to 100 in steps of 10 (I += 10).



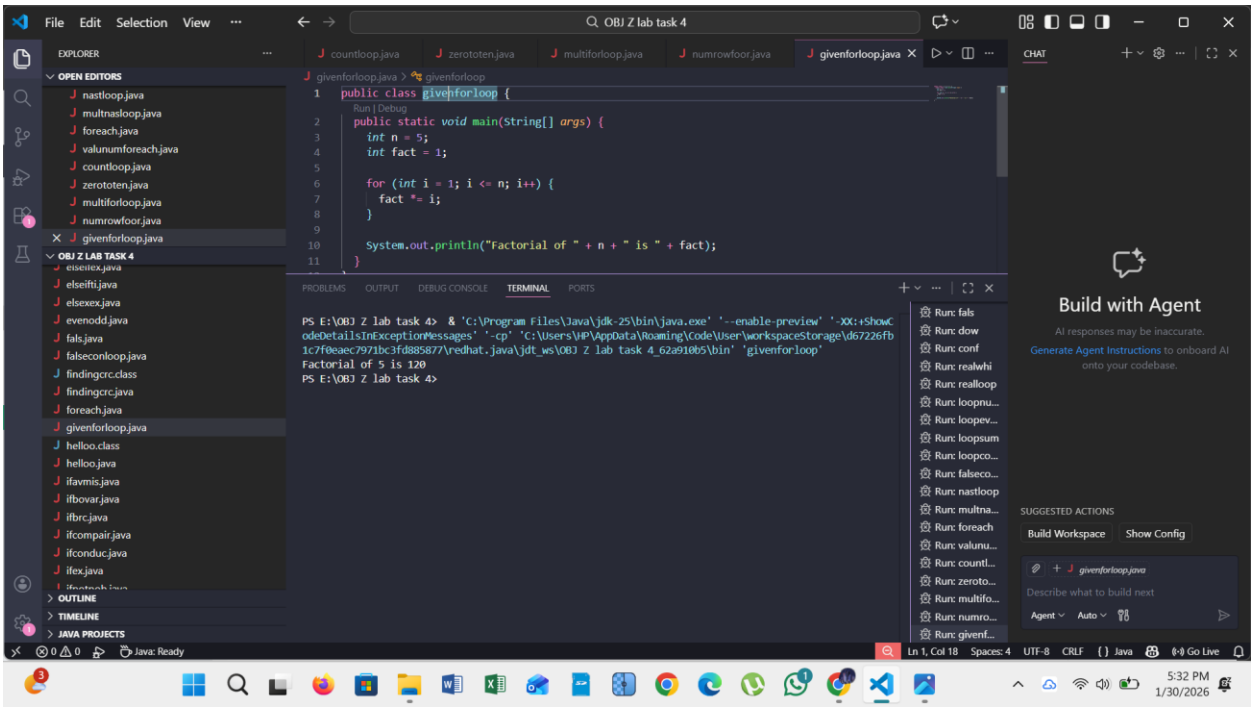
A basic loop counting from 0 to 10 by incrementing by 2.



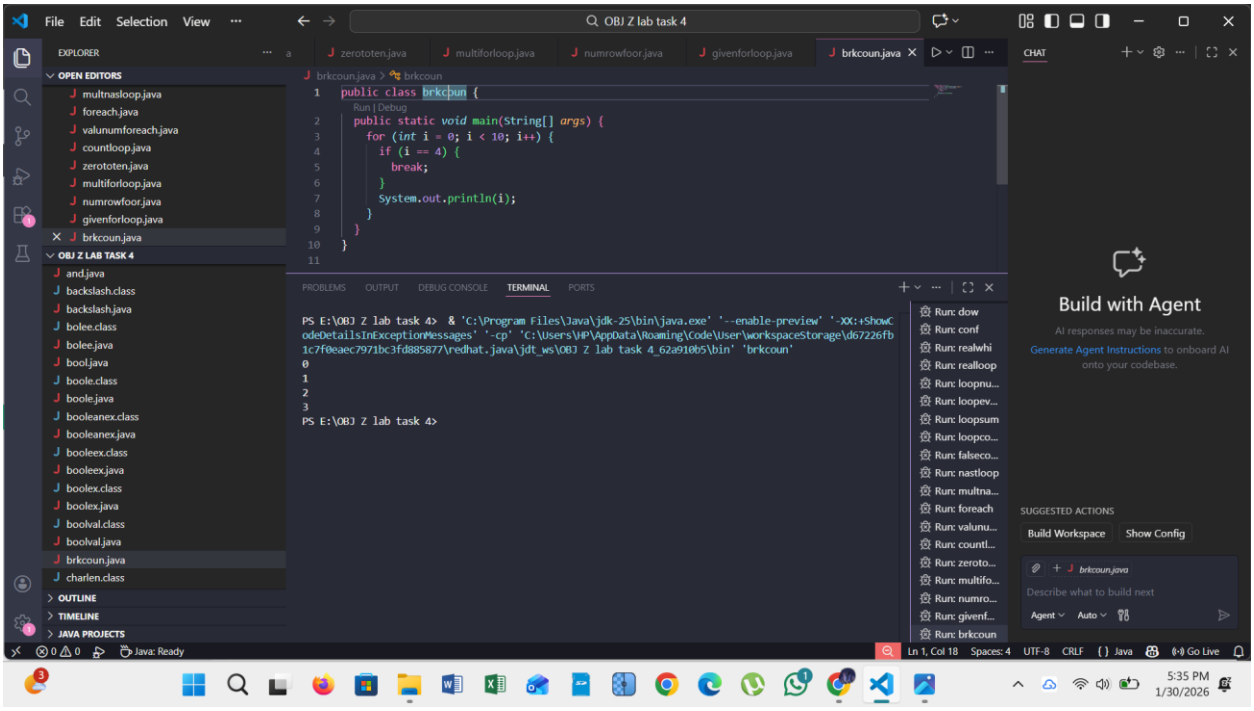
- A program that prints the **Multiplication Table** for the number 2 using a for loop.



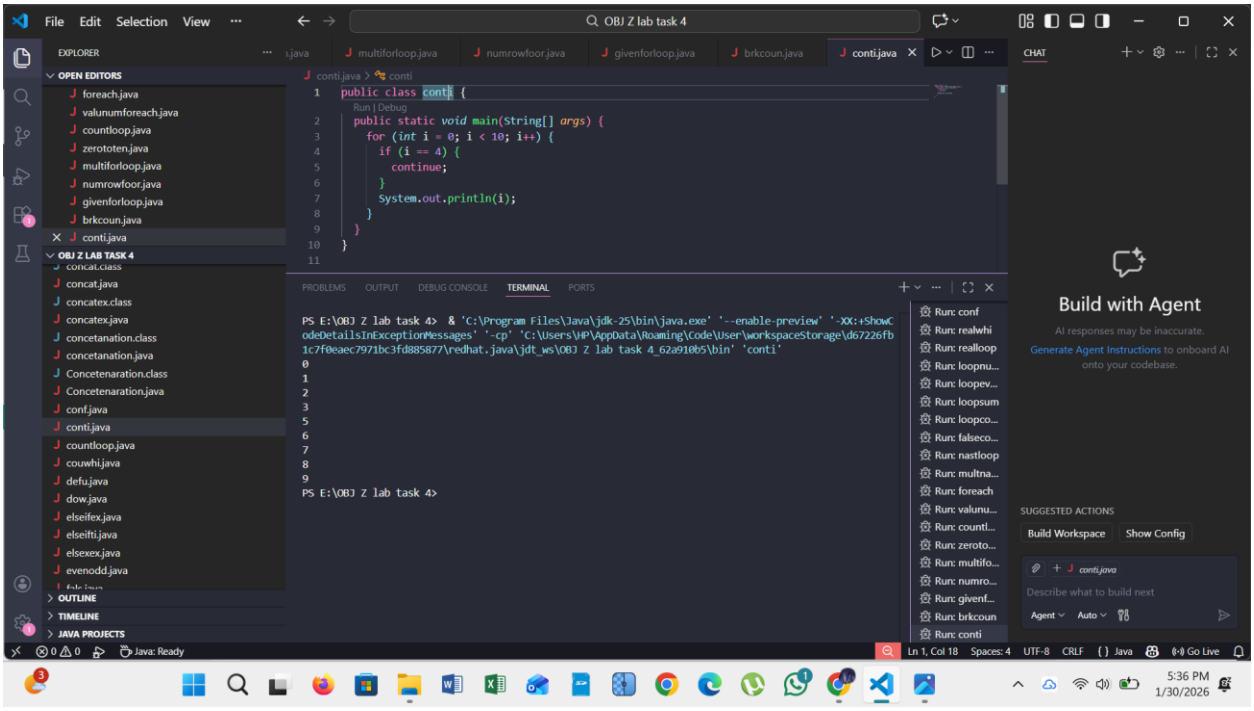
- Prints seat numbers 1 through 5 to verify loop iteration logic.



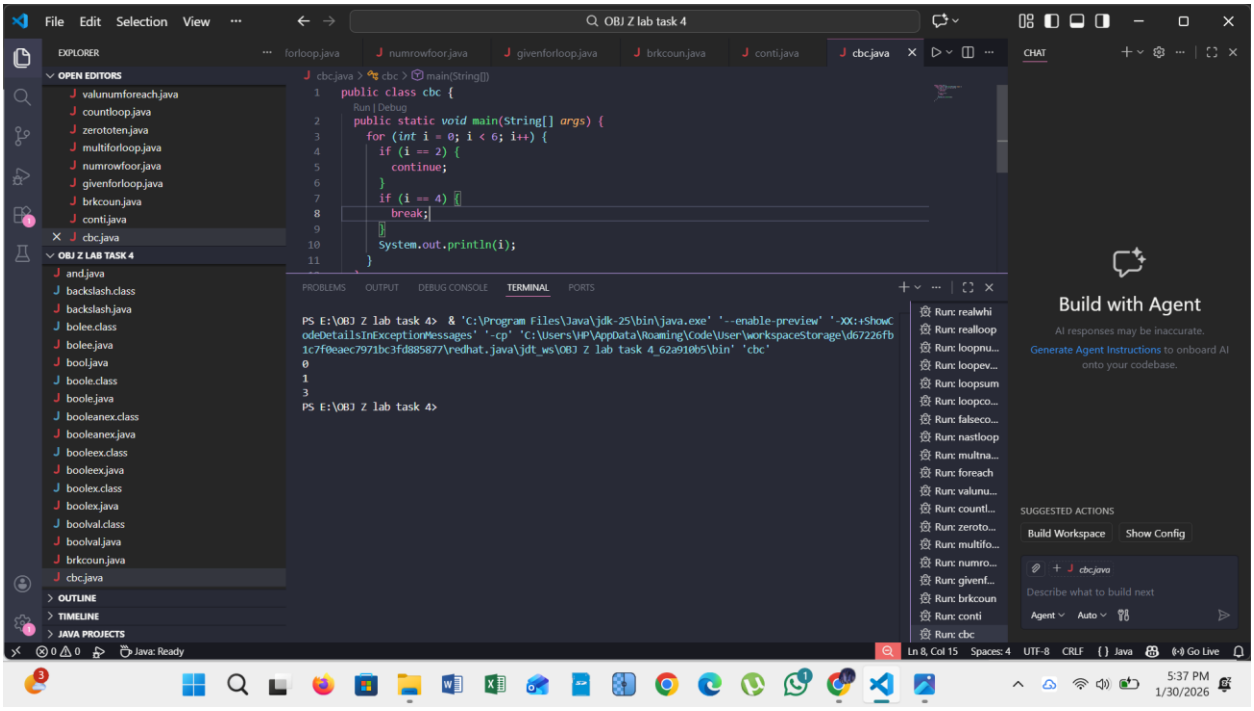
- Calculates the **Factorial** of a number (5!) by multiplying fact by i in every iteration.



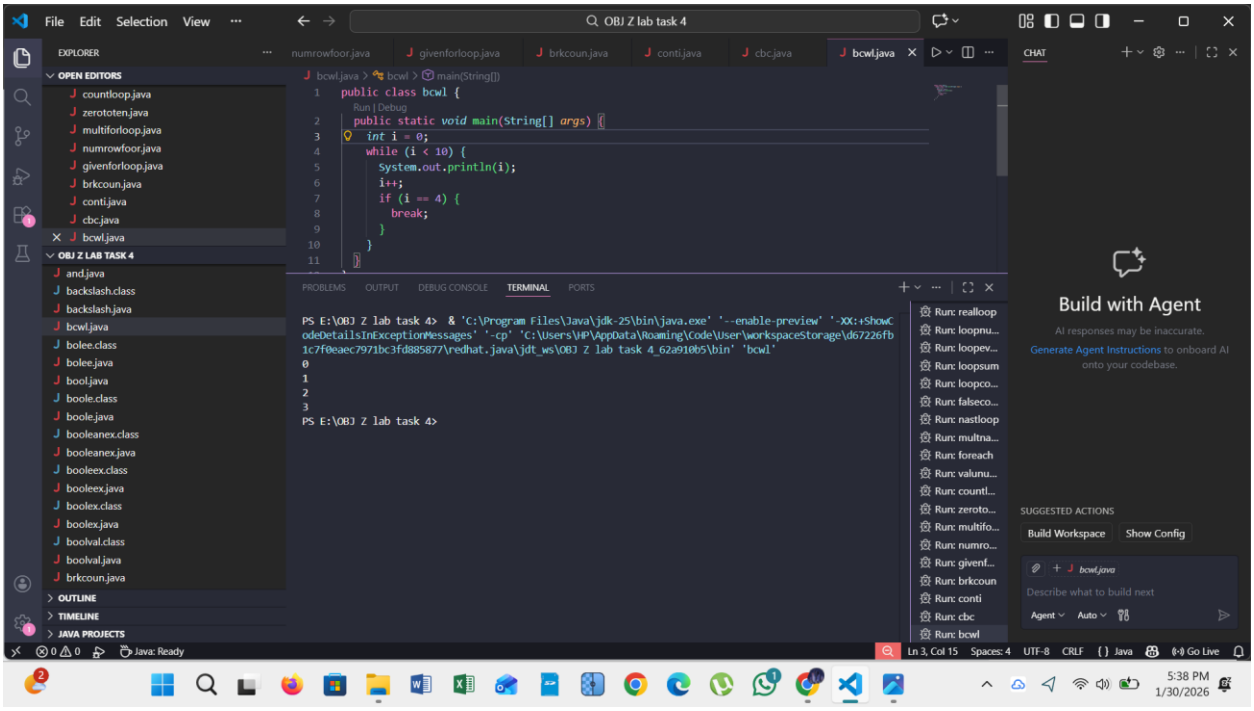
- Demonstrates the **break** statement. The loop stops completely when i equals 4.



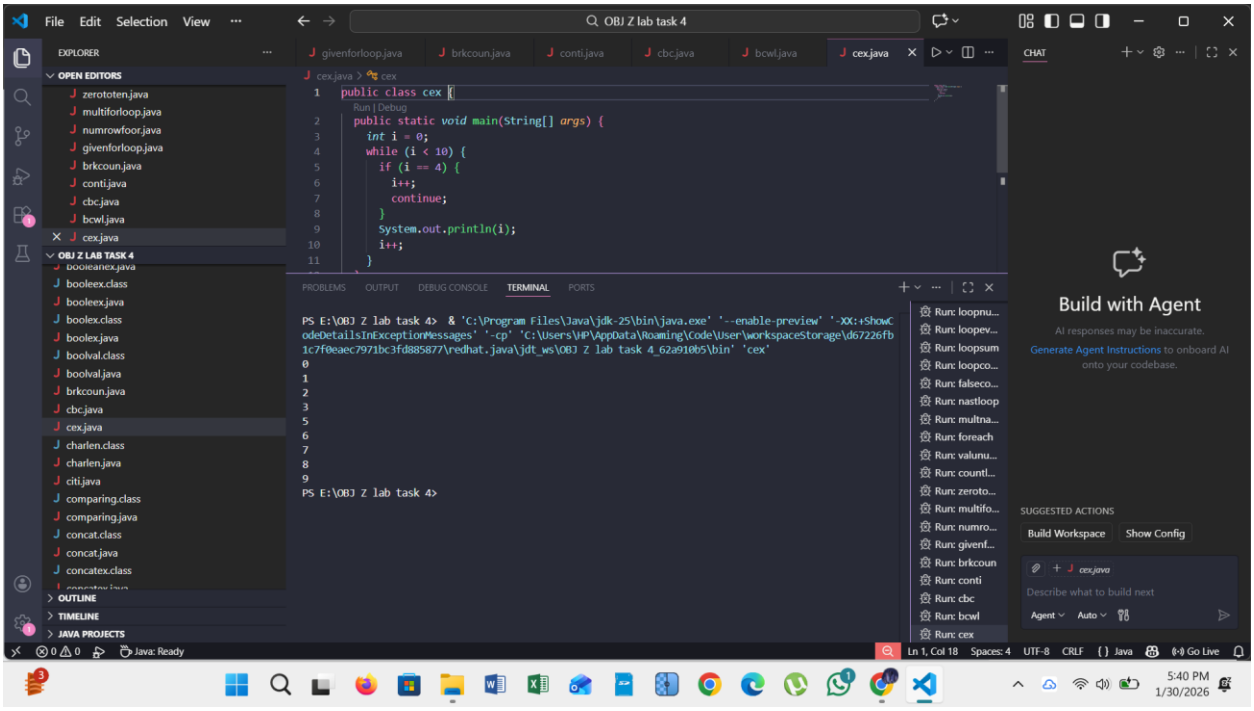
- Demonstrates the **continue** statement. When i equals 4, it skips that specific iteration but continues with the rest of the loop.



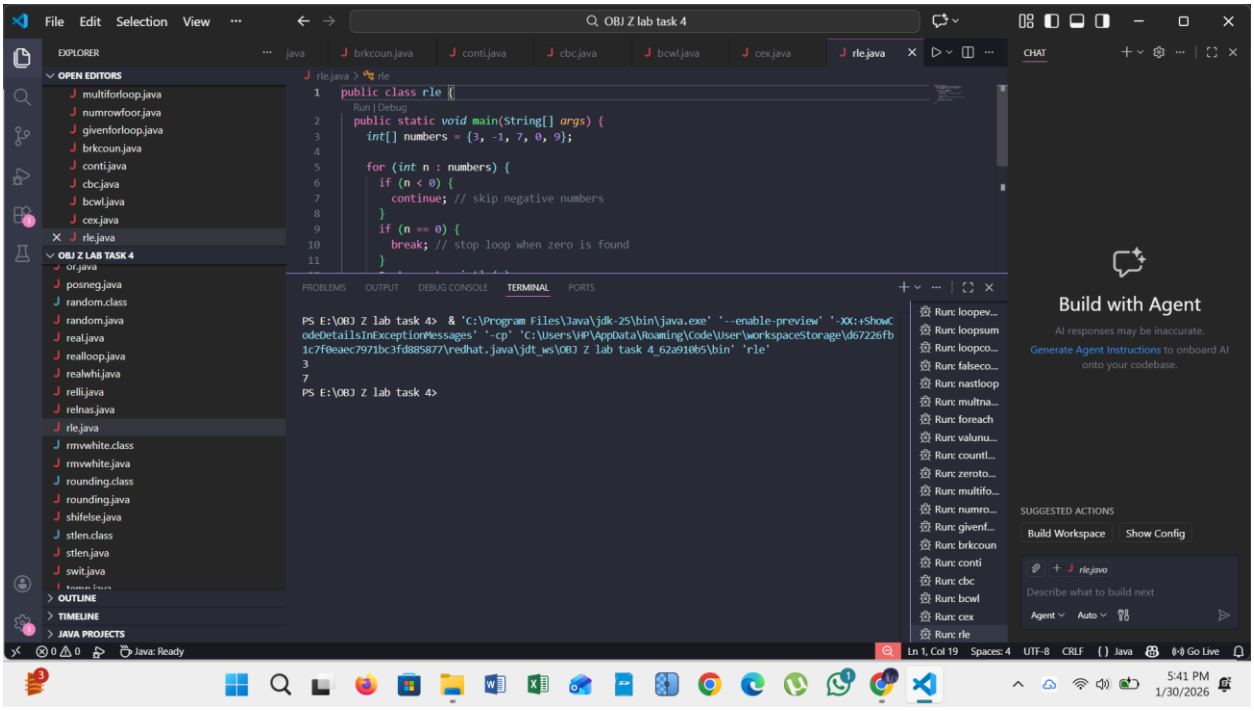
- A combined example using both continue (to skip 2) and break (to stop at 4).



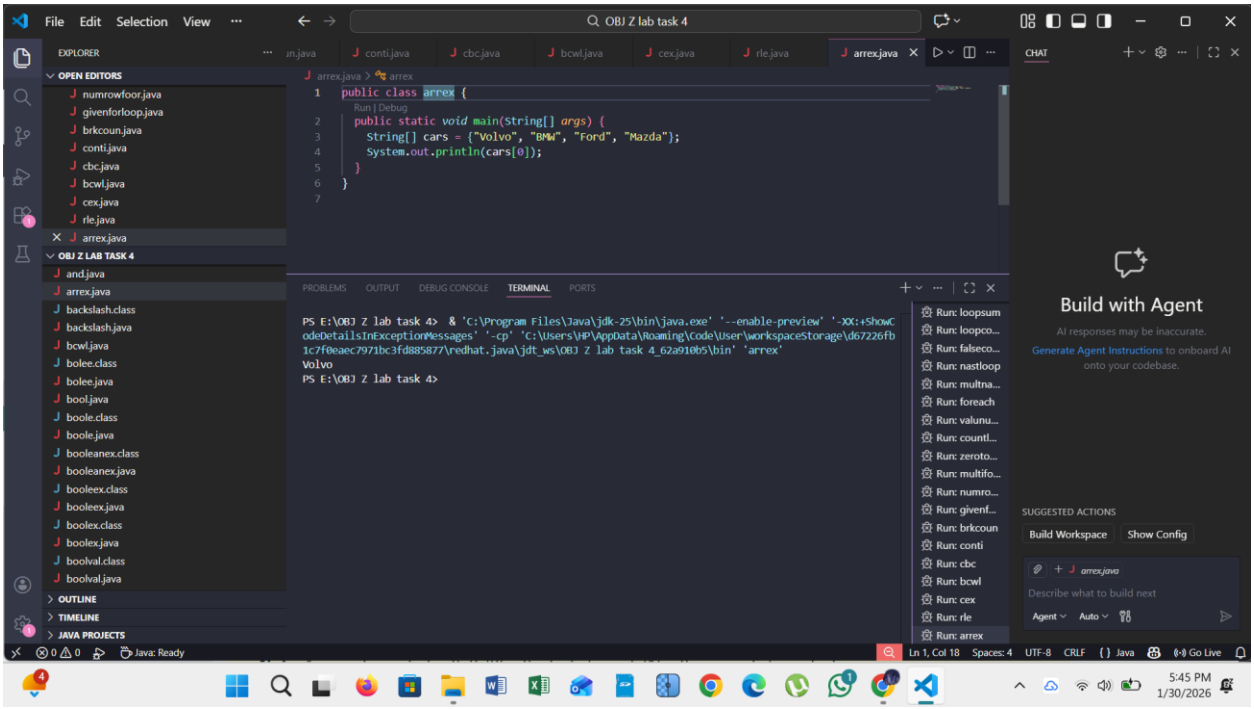
- Shows how break functions inside a while loop to exit the loop early when a condition is met.



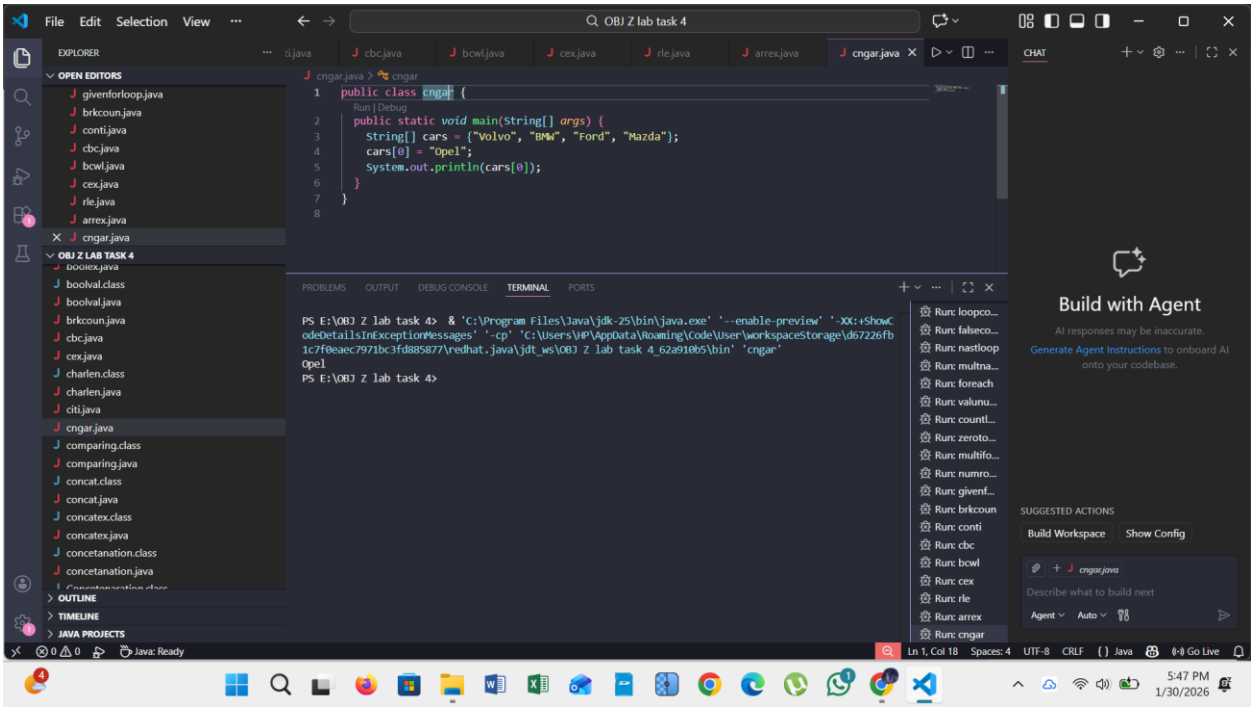
- Shows how continue functions inside a while loop to skip an iteration.



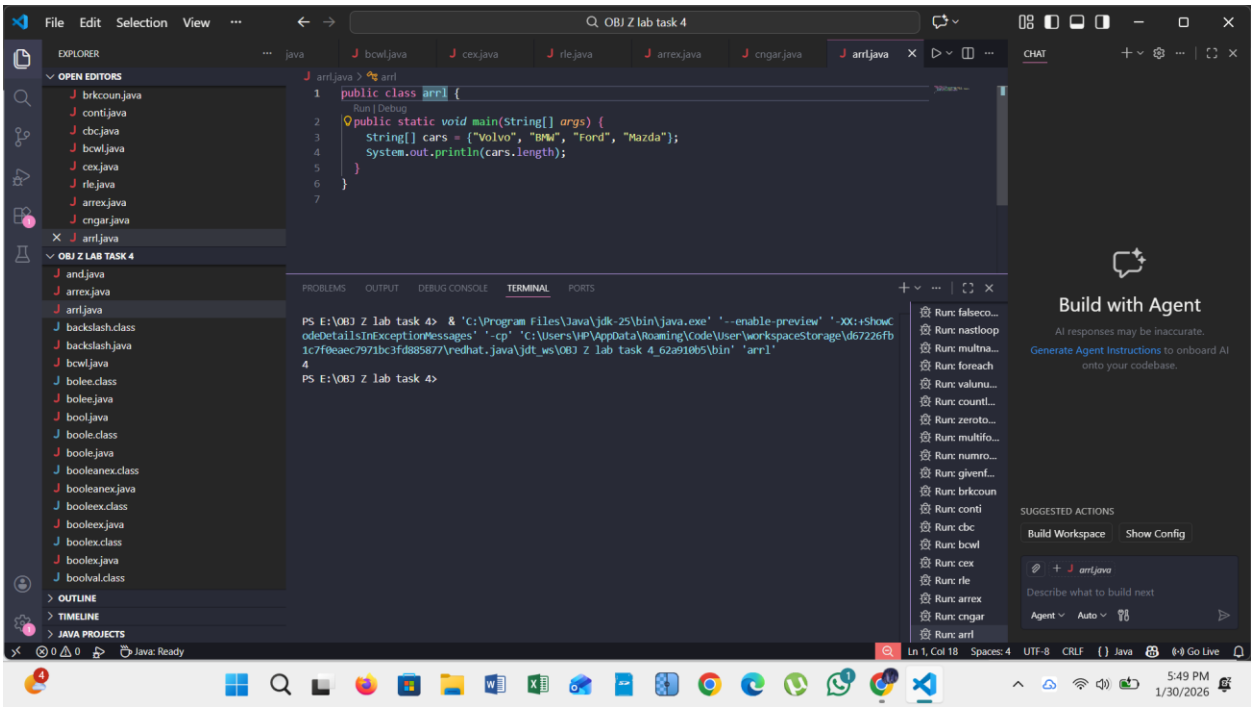
- Loops through an array of numbers. It uses continue to skip negative numbers and break to stop if a zero is found.



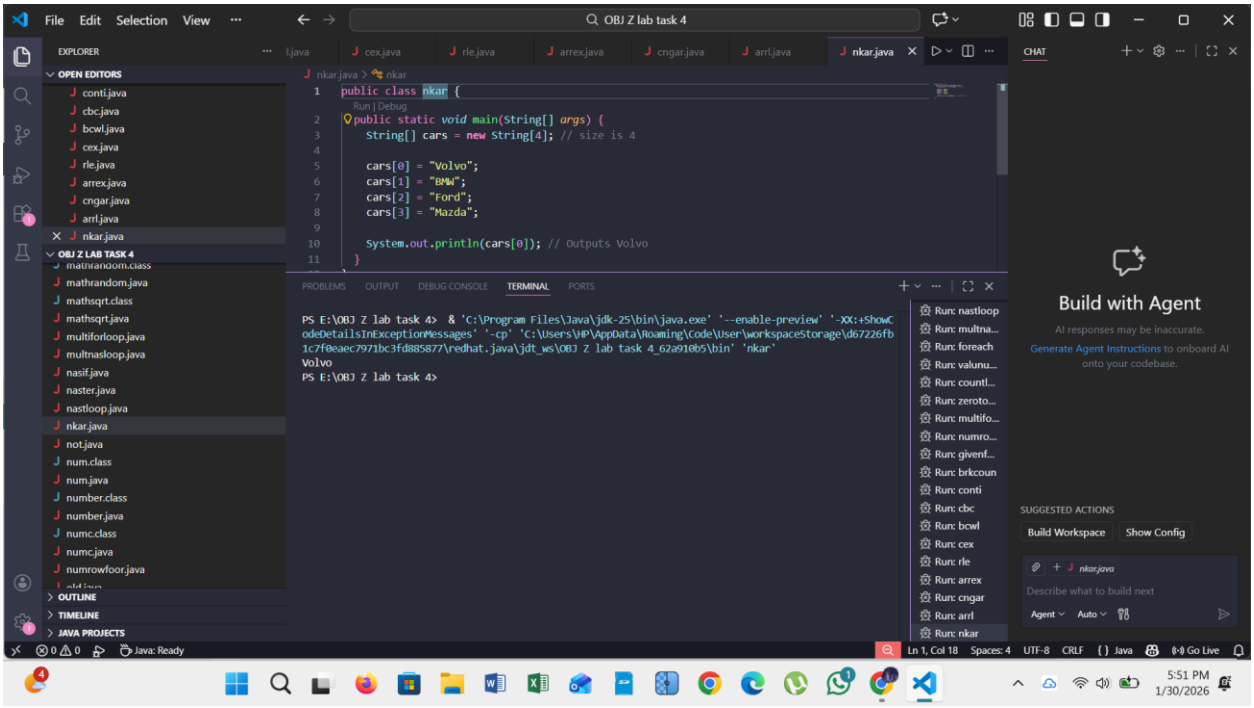
- Demonstrates how to **access** an array element using its index. cars[0] gets "Volvo".



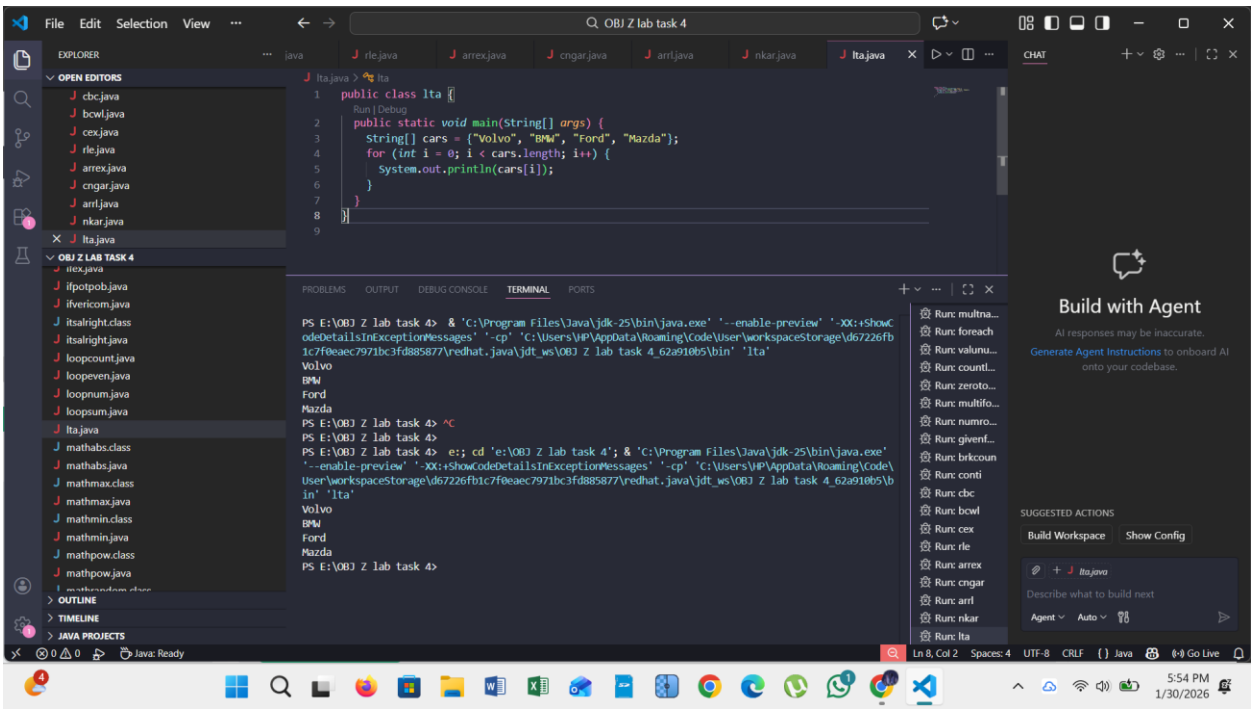
- Shows how to **change** an array element. It assigns "Opel" to index 0, replacing "Volvo".



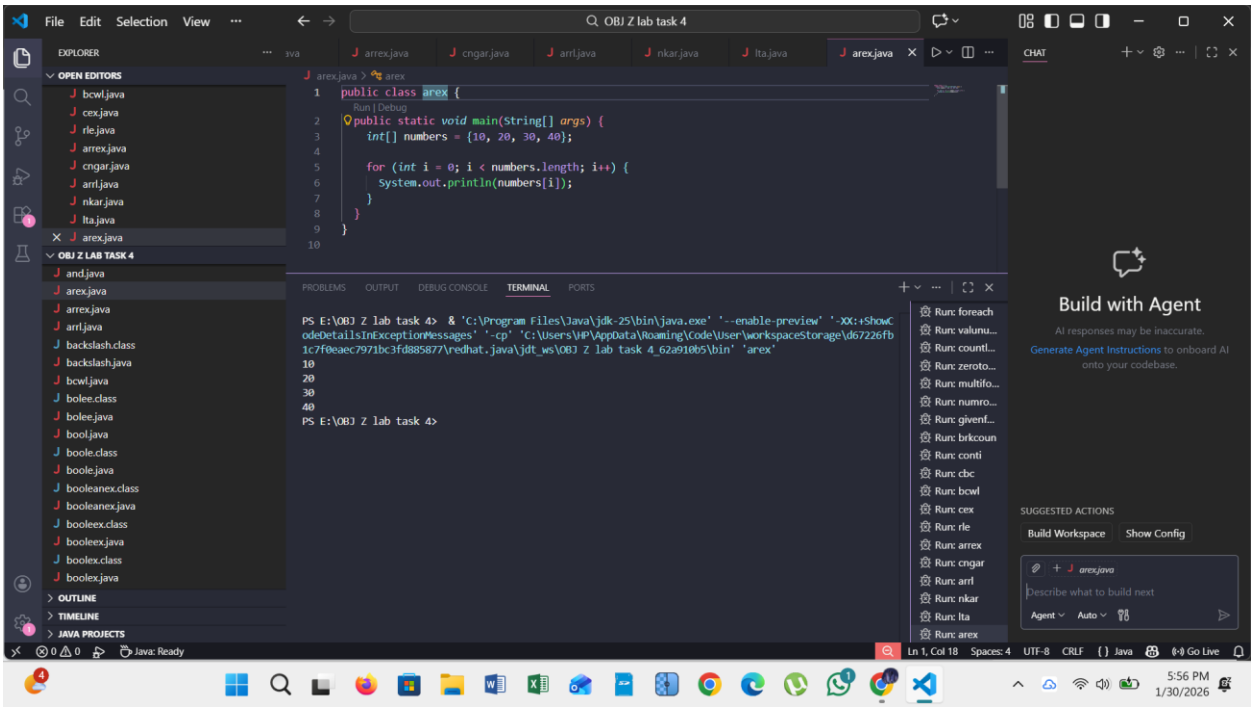
Uses the .length property to find out how many items are in the array (4 items).



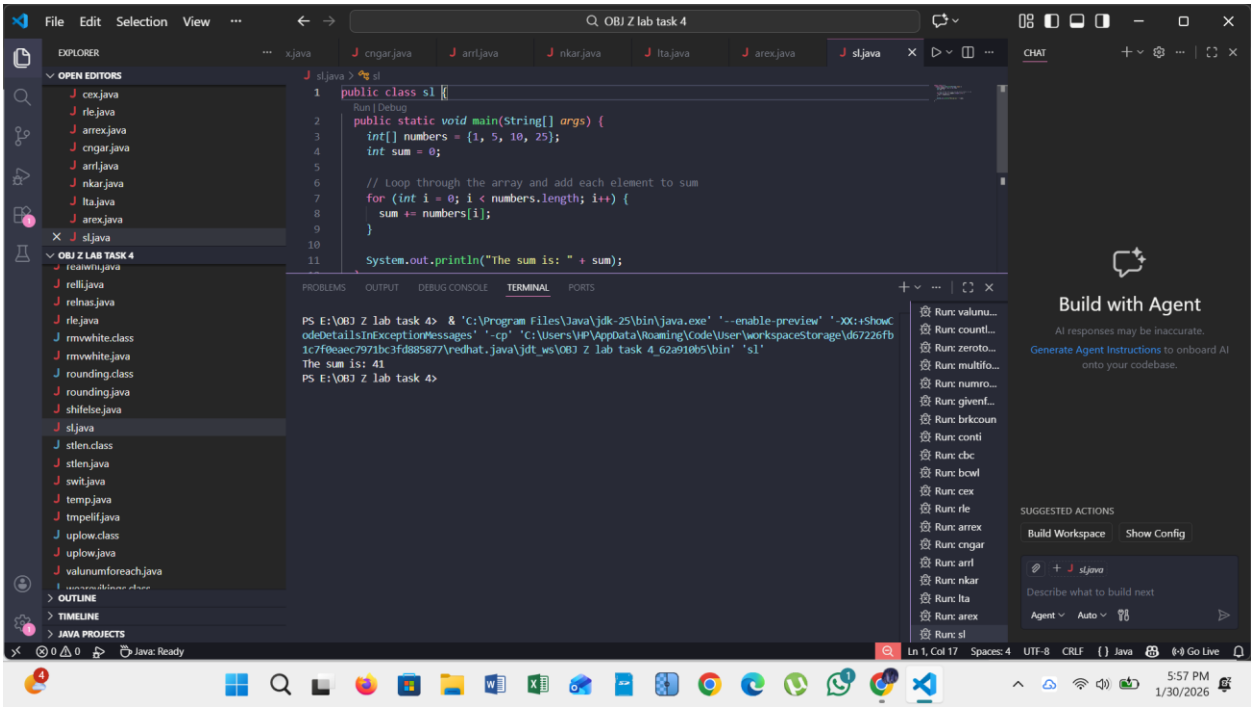
- Demonstrates how to declare an array with a fixed size (new String[4]) and then fill in the values manually.



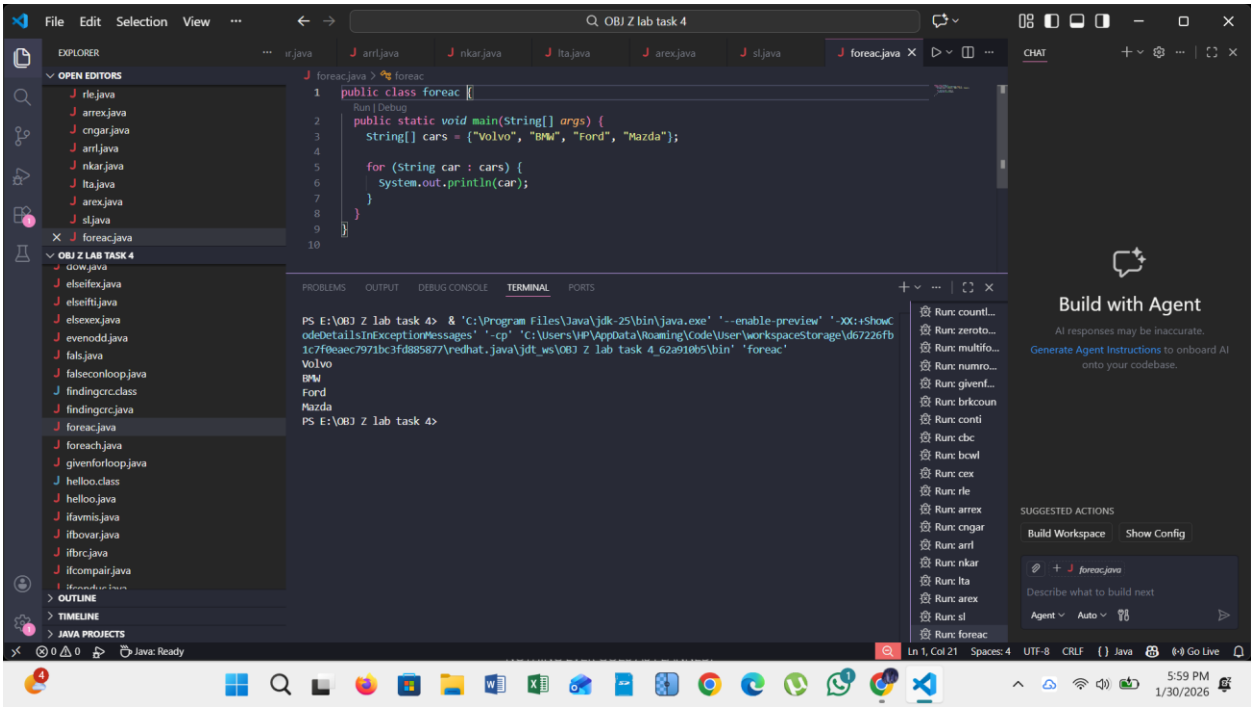
- Uses a standard for loop to iterate through an array using cars.length as the limit.



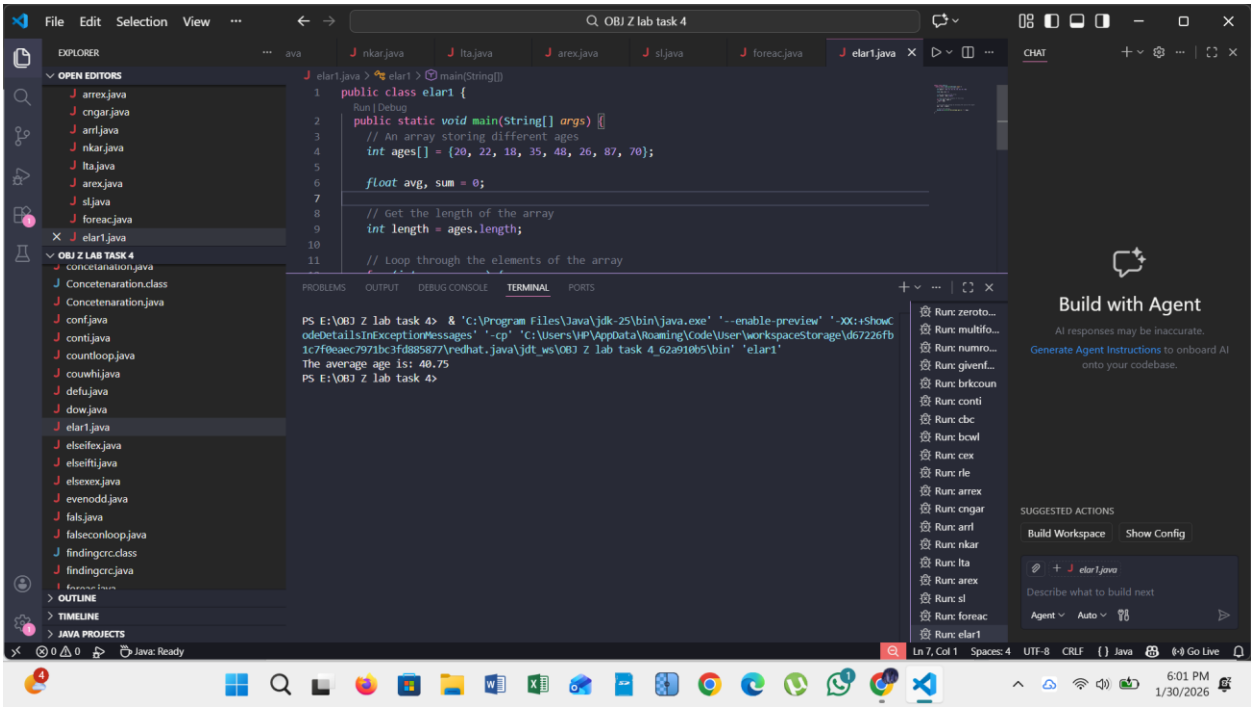
- Iterates through an integer array {10, 20, 30, 40} and prints each number.



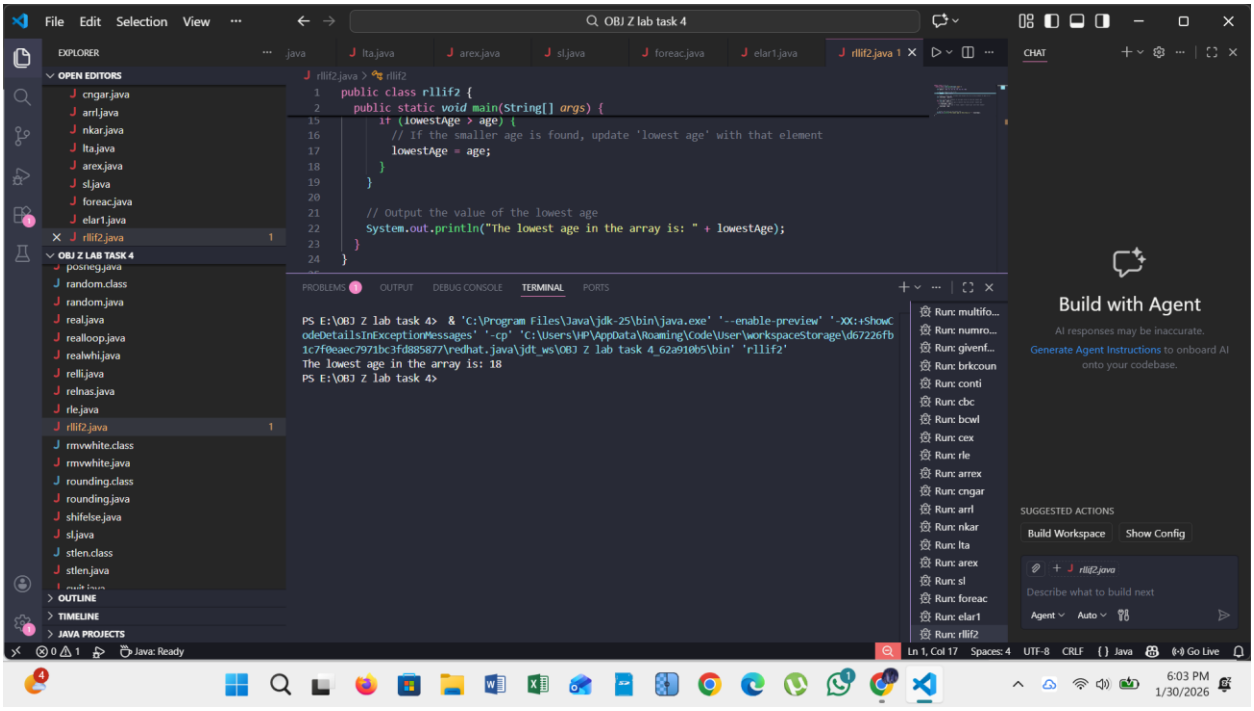
- Calculates the **Sum** of all numbers inside an integer array using a loop.



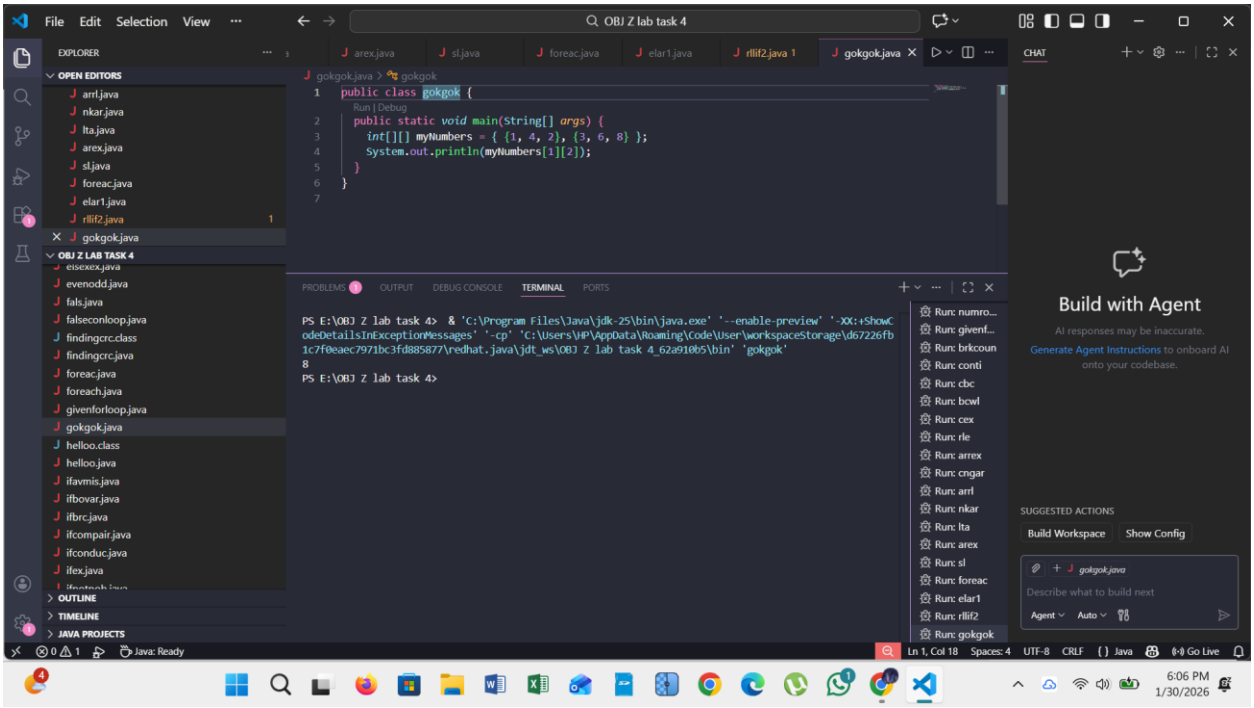
- Uses a **For-Each** loop to print every car brand in the cars array.



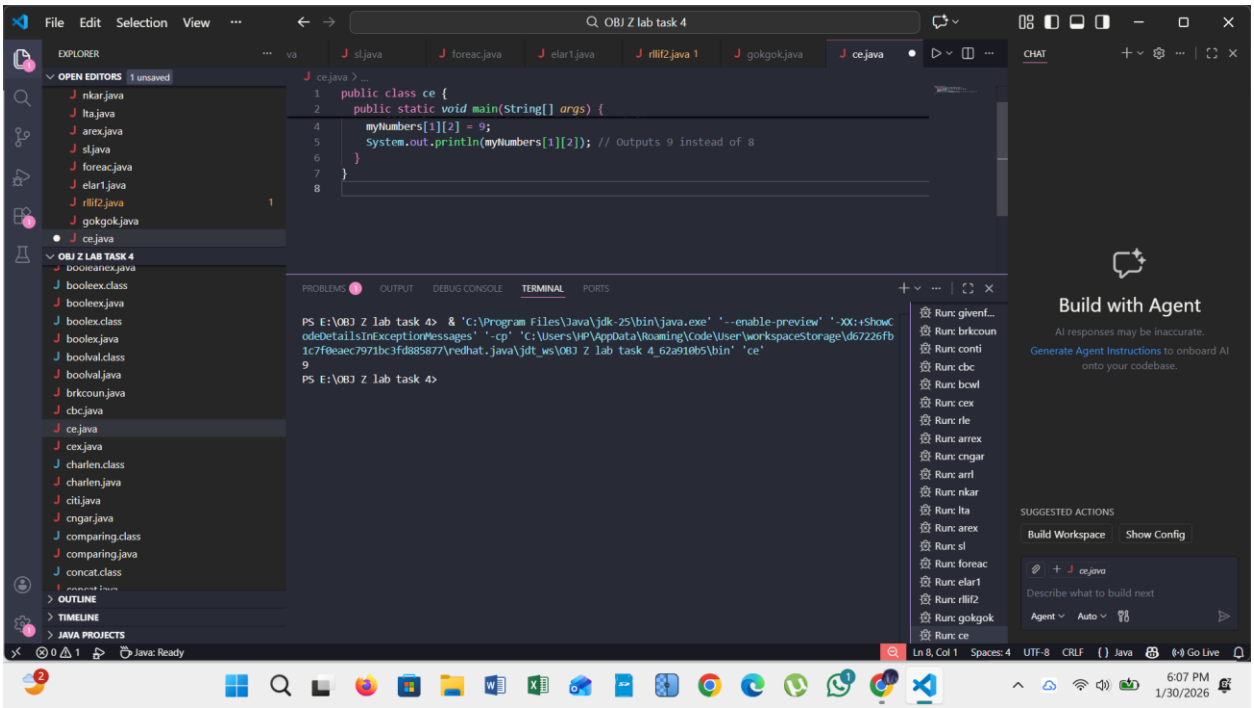
- Calculates the **Average** value of numbers in an array by summing them up and dividing by the array length.



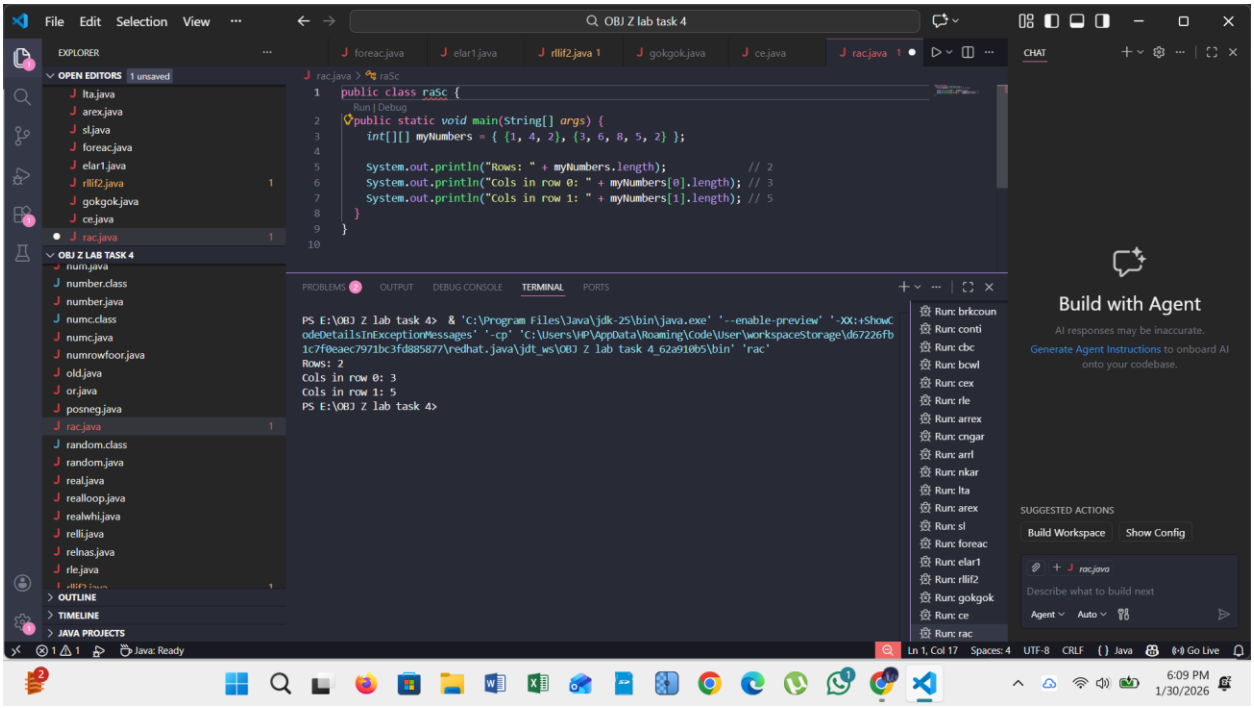
- Finds the **Lowest Age** in an array. It assumes the first number is the lowest, then compares it against every other number.



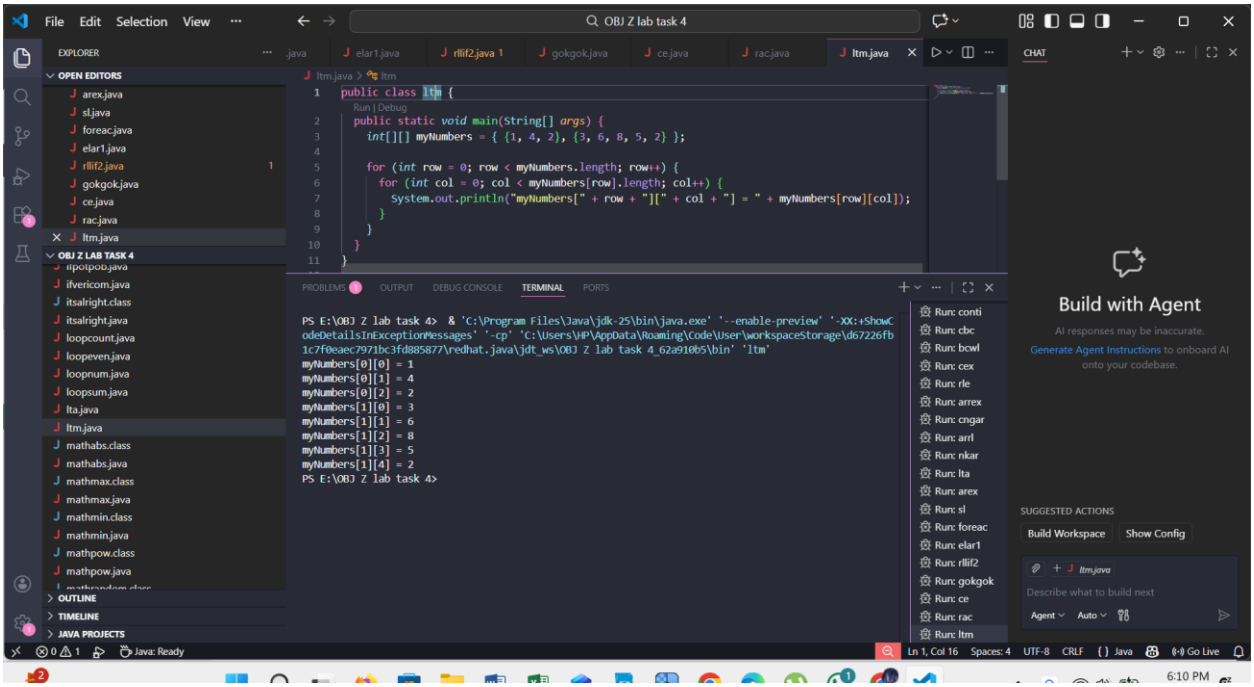
Demonstrates a Multidimensional Array (an array of arrays). myNumbers[1][2] accesses the second list, third item (value 8).



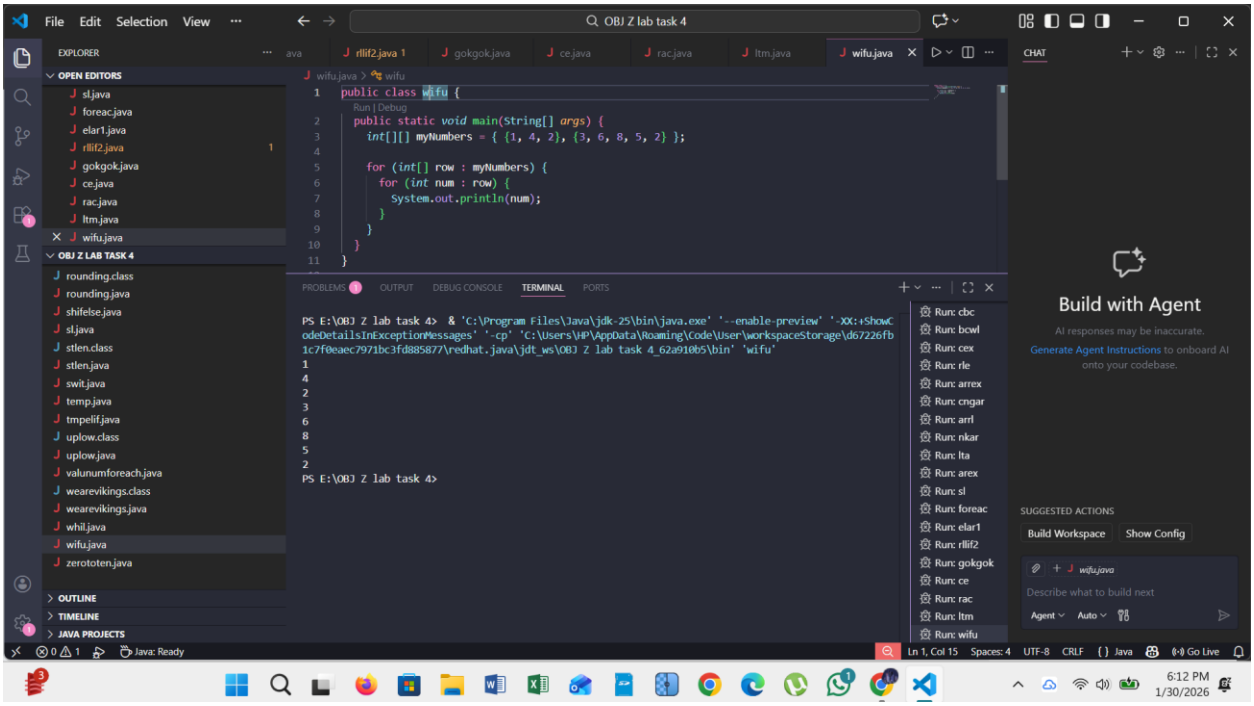
-
  - It skips printing the number 4 but continues counting the rest



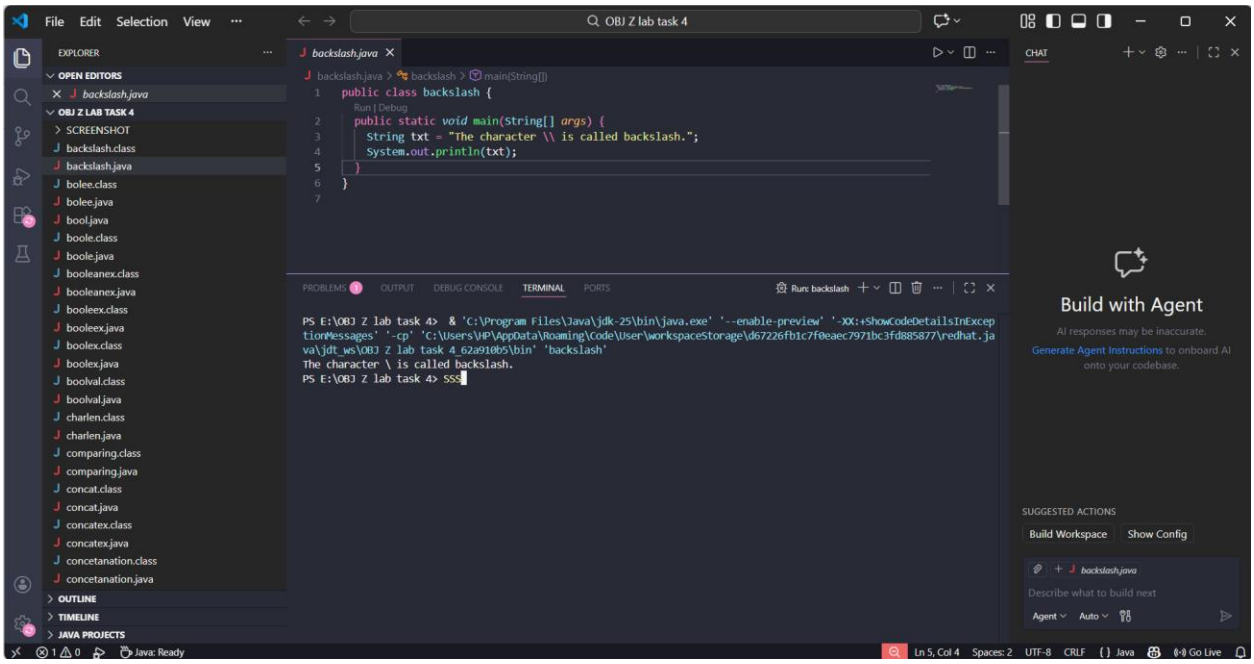
Checks a door code (1337). Rle.java iterates through an array skipping negative numbers.



-
  - checks a door code (1337). rle.java iterates through an array skipping negative numbers.



Whil.java demonstrates a basic while loop. Wearevikings.java demonstrates printing strings with quotes.



In Java, a single backslash \ is an Escape Character. It is used to introduce special characters like \" (quote), \n (new line), or \t (tab).