



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

Screenshot of the VS Code interface showing the code for `bolee.java`. The code contains a single line: `System.out.println(15 == 10);`. The terminal output shows the result as `false`, indicating that 15 is not equal to 10.

```
public class boolee{    public static void main(String[] args) {        System.out.println(15 == 10); // returns false, because 10 is not equal to 15    }}
```

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

Screenshot of the VS Code interface showing the code for `bool.java`. The code contains a line: `System.out.println(x == 10);`. The terminal output shows the result as `true`, indicating that the value of `x` is equal to 10.

```
public class bool{    public static void main(String[] args) {        int x = 10;        System.out.println(x == 10); // returns true, because the value of x is equal to 10    }}
```

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

The screenshot shows a Java application running in a terminal window. The code in `boole.java` is as follows:

```
public class boole {
    public static void main(String[] args) {
        System.out.println(10 > 9); // returns true, because 10 is higher than 9
    }
}
```

The terminal output shows the command run and the resulting output:

```
PS E:\OBJ Z lab task 4> & C:\Program Files\Java\jdk-25\bin\java.exe '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\VIP\AppData\Roaming\Code\User\workspaceStorage\d67226fb1c7feaecc7971bc3f88587\redhat.java\dt_ws\OBJ Z lab task 4_62a910b5\bin' 'boole'
true
PS E:\OBJ Z lab task 4>
```

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

A screenshot of a Java code editor interface. The central area shows a code editor with the following Java code:

```
public class booleanex {
    public static void main(String[] args) {
        int myAge = 25;
        int votingAge = 18;

        if (myAge >= votingAge) {
            System.out.println("Old enough to vote!");
        } else {
            System.out.println("Not old enough to vote.");
        }
    }
}
```

The code editor has tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, and PORTS. The TERMINAL tab is active, displaying the command line output:

```
PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '--showCodeDetailsInExceptionMessages' '-cp' 'C:/Users/VIP/AppData/Roaming/Code/User/workspaceStorage/d67226fb1c7f0eac797bc3fd885877/redhat.java/jdt-ws/OBJ Z lab task 4.62a910b5/bin' 'booleanex'
old enough to vote!
```

The right side of the interface includes a 'Build with Agent' section with options like 'Run: backsl...', 'Run: bole...', 'Run: bool...', and 'Run: boolea...'. Below this is a 'SUGGESTED ACTIONS' section with a 'Build Workspace' button and a 'Show Config' button. A tooltip for 'booleanex.java' suggests 'Describe what to build next'.

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

The screenshot shows a Java code editor interface with the following details:

- File Menu:** File, Edit, Selection, View, ...
- Title Bar:** Q OBJ Z lab task 4
- Editor Area:** Displays a Java file named booleex.java with the following code:

```
public class booleex{    public static void main(String[] args) {        int myAge = 25;        int votingAge = 18;        System.out.println(myAge >= votingAge); // returns true (25 year olds are allowed to vote)    }}
```
- Terminal Tab:** Shows the command PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '--showCodeDetailsInExceptionMessages' '-cp' 'C:\Users\HP\AppData\Roaming\Code\User\workspaceStorage\d67226fb1c7f0eaecc797bc3fd8b85877\nredhat.java\jdt_ws\OBJ Z lab task 4_62a910b5\bin' 'booleex' followed by the output true.
- Output Tab:** Shows Java: Ready
- Suggested Actions:** Build Workspace, Show Config, + J booleex.java (Describe what to build next), Agent, Auto, Go Live
- Build with Agent:** AI responses may be inaccurate. Generate Agent Instructions to onboard AI onto your codebase.

This cod check veriables to compare my voatingage

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows a tree view of files and folders. Opened files include `boolex.java`, `backslash.java`, `bool.java`, `bole.class`, `bole.java`, `booleanex.class`, `booleanex.java`, `boolex.class`, `boole.java`, `boolval.class`, `boolval.java`, `charlen.class`, `charlen.java`, `comparing.class`, `comparing.java`, `concat.class`, `concat.java`, `concatex.class`, `concatex.java`, and `concatenation.class`. A folder named `OBJ Z LAB TASK 4` is expanded.
- Editor:** Displays the `boolex.java` file with the following code:

```
public class boolex{
    public static void main(String[] args) {
        int x = 10;
        int y = 9;
        System.out.println(x > y); // returns true, because 10 is higher than 9
    }
}
```
- Terminal:** Shows the command-line output:

```
PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:/Users/VAH/AppData/Roaming/Code/User/workspaceStorage/d67226fb1c7f0eaecc971bc3fd885877/reddhat.java/jdt_ws/Obj Z lab task 4.62a9105/bin' 'boolex'
true
PS E:\OBJ Z lab task 4>
```
- Right Panel:** Includes a "Build with Agent" section with a dropdown menu showing options like "Run: backsl...", "Run: bole...", etc. It also has a "SUGGESTED ACTIONS" section with a "Build Workspace" button and a "Describe what to build next" input field.

Checking variable using `>` this operator .

The screenshot shows a Java code editor interface with the following details:

- File Bar:** File, Edit, Selection, View, ...
- Title Bar:** Q OBJ Z lab task 4
- Explorer:** Shows a tree view of files and projects, including **OPEN EDITORS** (bolee.java, booleval.java), **OBJ Z LAB TASK 4** (backslash.class, backslash.java, bolee.class, boole.java, boole.class, booleeval.class, booleanex.class, booleanex.java, boolex.class, boolex.java, charlen.class, charlen.java, comparing.class, comparing.java, concat.class, concat.java, concatex.class, concatex.java, concatenation.class), and **OUTLINE**, **TIMELINE**, **JAVA PROJECTS**.
- Editor Area:** Displays two open files:
 - boole.java:** A simple class definition.
 - booleval.java:** Contains the following code:

```
public class booleval {  
    public static void main(String[] args) {  
        boolean isJavaFun = true;  
        boolean isFishTasty = false;  
        System.out.println(isJavaFun);  
        System.out.println(isFishTasty);  
    }  
}
```
- Terminal:** Shows the command line output:

```
PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '--showCodeDetailsInExceptionMessages' '-cp' 'C:/Users/VIP/AppData/Roaming/Code/User/workspaceStorage/d67226fb1c7f0eac797bc3fd805877/redhat.java/jdt-ws/Obj Z lab task 4_62a9105/bin' 'boole'  
true
```
- Suggested Actions:** A sidebar with options like Run: backslash, Run: boole, Run: booleeval, Run: booleanex, Run: boolex, and Run: boolex.
- Bottom Bar:** Includes links for Build Workspace and Show Config, along with workspace statistics (Ln 1, Col 1, Spaces: 2, CRLF, Java, Go Live).

Demonstrates the basic declaration of true and false boolean types.

uses

The screenshot shows a Java code editor interface with the following details:

- File Menu:** File, Edit, Selection, View, ...
- Title Bar:** Q OBJ Z lab task 4
- Explorer:** Shows a tree view of files and projects. Under "OPEN EDITORS", there is 1 unsaved file: "charlen.java". Other files listed include "bole.java", "charlen.java", "bole.class", "bole.java", "bool.java", "bole.class", "bole.java", "booleanex.class", "boleanex.java", "boleex.class", "boolclass.java", "bool.java", "boolval.class", "boolval.java", "charlen.class", "charlen.java", "comparing.class", "comparing.java", "concat.class", "concat.java", "concatex.class", "concatex.java", "concatenation.class", "concatenation.java", "Concetenaration.class", and "Concetenaration.java".
- Editor Area:** Displays the "charlen.java" code:

```
public class charlen {
    public static void main(String[] args) {
        String txt = "Hello";
        System.out.println(txt.charAt(index: 0)); // H
        System.out.println(txt.charAt(index: 4)); // o
    }
}
```
- Terminal:** Shows the command line output:

```
PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '.\charlen'
H
o
PS E:\OBJ Z lab task 4>
```
- Suggested Actions:** A dropdown menu titled "Build with Agent" lists several options:
 - Run: backsl...
 - Run: bole...
 - Run: boole...
 - Run: boole...
 - Run: boole...
 - Run: boole...
 - Run: charlen
- Bottom Status Bar:** Shows the current line (Ln 7, Col 2), spaces (Spaces: 2), encoding (UTF-8), and file type (Java).

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

The screenshot shows a Microsoft Visual Studio Code (VS Code) interface. The left sidebar displays a file tree with several Java files and class files under the 'OBJ Z LAB TASK 4' project. The main editor area shows the content of 'comparing.java'. The terminal at the bottom shows the execution of the program, printing 'true' and 'false' respectively for the comparisons between 'txt1' and 'txt2', and 'txt3' and 'txt4'. A context menu is open over the terminal output, listing various run configurations for the Java files.

```
public class comparing {
    public static void main(String[] args) {
        String txt1 = "Hello";
        String txt2 = "Hello";
        String txt3 = "Greetings";
        String txt4 = "Great things";
        System.out.println(txt1.equals(txt2));
        System.out.println(txt3.equals(txt4));
    }
}
```

```
PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '--showCodeDetailsInExceptionMessages' '-cp' 'C:/Users/VIP/AppData/Roaming/Code/User/workspaceStorage/d67226fb1c7f0eac7971bc3fd885877/redhat.java/jdt_ws/Obj Z lab task 4.62a910b5/bin' 'comparing'
true
false
```

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

The screenshot shows the Visual Studio Code interface with the following details:

- File Bar:** File, Edit, Selection, View, ...
- Title Bar:** Q_OBJ Z lab task 4
- Explorer:** Shows a tree view of files and folders, including **OBJ Z LAB TASK 4** which contains **boolee.class**, **boolee.java**, **charlen.java**, and **concat.java**.
- Editor:** Three tabs are open: **bolee.java**, **charlen.java**, and **concat.java**. The **concat.java** tab is active, displaying the following code:

```
1 public class concat {  
2     public static void main(String[] args) {  
3         String firstName = "John ";  
4         String lastName = "Doe";  
5         System.out.println(firstName.concat(lastName));  
6     }  
7 }
```
- Terminal:** Shows the command PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '--showCodeDetailsInExceptionMessages' '-cp' 'C:\Users\VA\P\AppData\Roaming\Code\User\workspaceStorage\d67226fb1c7f0eaecc971bc3fd885877\redhat.java\jdt-ws\OBJ Z lab task 4.62a9105\bin' 'concat'. The output is John Doe.
- Suggested Actions:** A dropdown menu titled "Build with Agent" lists several build configurations for the **concat.java** file.
- Bottom Status Bar:** Shows Ln 7, Col 2, Spaces: 2, UTF-8, CRLF, Java, Go Live, and a Java icon.

Concat() method to show output as full .

The screenshot shows the Visual Studio Code interface with the following details:

- File Bar:** File, Edit, Selection, View, ...
- Editor:** Three tabs are open: bolee.java, charlen.java, and concatex.java. concatex.java is the active tab.
- Code:** The concatex.java file contains the following Java code:

```
public class concatex {
    public static void main(String[] args) {
        String a = "Java ";
        String b = "is ";
        String c = "fun!";
        String result = a.concat(b).concat(c);
        System.out.println(result);
    }
}
```
- Terminal:** The terminal shows the command PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '--showCodeDetailsInExceptionMessages' '-cp' 'C:/Users/VIP/AppData/Roaming/Code/User/workspaceStorage/d67226fb1c7f0eaecc797bc3fd885877/redhat.java/jdt-ws/Obj Z lab task 4.62a9105\bin' 'concatex' Java is fun!
- Output:** Shows various run configurations for the concatex.java file.
- Suggested Actions:** A panel on the right suggests building the workspace or the current file.
- Bottom Status Bar:** Ln 5, Col 23, Spaces: 2, UTF-8, CRLF, Java, Go Live, and a status message Java: Ready.

Adjust variable using concat() .

A screenshot of the Visual Studio Code (VS Code) interface. The terminal at the bottom shows the output of running a Java program named 'concatenation'. The code in the editor defines a class 'concatenation' with a main method that prints 'My name is John and I am 25 years old.' to the console. The AI sidebar on the right suggests various run configurations for other Java files in the workspace.

```
PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '--showCodeDetailsInExceptionMessages' '-cp' 'C:\Users\VHP\AppData\Roaming\Code\User\workspaceStorage\d67226fb1c7f0eaecc9791bc3fd889877\redhat.java\jdt-ws\OBJ Z lab task 4.62a910b5\bin' 'concatenation'
My name is John and I am 25 years old.
PS E:\OBJ Z lab task 4>
```

The AI sidebar displays the following suggested actions:

- Run: backsl...
- Run: bole...
- Run: bool...
- Run: boolea...
- Run: boolex...
- Run: boolex...
- Run: charlen...
- Run: compa...
- Run: concat...
- Run: concatex...
- Run: concet...

SUGGESTED ACTIONS

Build Workspace Show Config

0 + J concatenation.java

Describe what to build next

Agent Auto

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

The screenshot shows the Visual Studio Code (VS Code) interface with the following details:

- File Bar:** File, Edit, Selection, View, ...
- Title Bar:** Q OBJ Z lab task 4
- Explorer:** Shows files in the workspace, including `boole.java`, `charlen.java`, `Concetenaration.java`, and many other Java files under `OBJ Z LAB TASK 4`.
- Editor:** Displays the `Concetenaration.java` file with the following code:

```
1 public class Concetenaration {  
2     public static void main(String args[]) {  
3         String firstName = "John";  
4         String lastName = "Doe";  
5         System.out.println(firstName + " " + lastName);  
6     }  
7 }  
8 }
```
- Terminal:** Shows the command line output:

```
PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:/Users/VIP/AppData/Roaming/Code/User/workspaceStorage/d67226fb1cf0eaecc971bc3f3d885877/redhat.java/jdt_ws/OBJ Z lab task 4.62a910b5/bin' 'Concetenaration'  
John Doe  
PS E:\OBJ Z lab task 4>
```
- Suggested Actions:** A sidebar with various run configurations for the current file.
- Bottom Status Bar:** Ln 8, Col 1, Spaces: 2, UTF-8, CRLF, (Java), Go Live, etc.

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

The screenshot shows the Visual Studio Code (VS Code) interface with the following details:

- File Bar:** File, Edit, Selection, View, ...
- Title Bar:** Q OBJ Z lab task 4
- Explorer:** Shows files in the workspace, including `boole.java`, `charlen.java`, `findingcrc.java`, and many other Java files under `OBJ Z LAB TASK 4`.
- Editor:** Displays the `findingcrc.java` file with the following code:

```
1 public class findingcrc {  
2     public static void main(String[] args) {  
3         String txt = "Please locate where 'locate' occurs!";  
4         System.out.println(txt.indexOf("locate"));  
5     }  
6 }  
7 }
```
- Terminal:** Shows the command line output:

```
PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:/Users/VIP/AppData/Roaming/Code/User/workspaceStorage/d67226fb1cf0eaecc971bc3f3d885877/redhat.java/jdt_ws/OBJ Z lab task 4.62a910b5/bin' 'findingcrc'  
7  
PS E:\OBJ Z lab task 4>
```
- Suggested Actions:** A sidebar with various run configurations for the current file.
- Bottom Status Bar:** Ln 7, Col 1, Spaces: 2, UTF-8, CRLF, (Java), Go Live, etc.

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 screenshot shows the VS Code interface with the terminal tab active. The terminal window displays the output of a Java program named `heloo.java`. The code defines a class `heloo` with a `main` method that prints the string "Hello". The terminal shows the command run and the resulting output "Hello".

```
PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '--XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:/Users/VIP/AppData/Roaming/Code/User/workspaceStorage/d67226fb1c7f0eaecc971bc3fd885877/redhat.java/jdt_ws/OBJ Z lab task 4.62a910b5/bin' 'heloo'
Hello
PS E:\OBJ Z lab task 4>
```

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

This screenshot shows the VS Code interface with the terminal tab active. The terminal window displays the output of a Java program named `itsalright.java`. The code defines a class `itsalright` with a `main` method that prints the string "It's alright.". The terminal shows the command run and the resulting output "It's alright.".

```
PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '--XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:/Users/VIP/AppData/Roaming/Code/User/workspaceStorage/d67226fb1c7f0eaecc971bc3fd885877/redhat.java/jdt_ws/OBJ Z lab task 4.62a910b5/bin' 'itsalright'
It's alright.
PS E:\OBJ Z lab task 4>
```

This screenshot shows VS Code running `itsalright.java` and printing "**It's alright.**" in the terminal.

The screenshot shows the Eclipse IDE interface with the title bar "OBJ Z lab task 4". The left sidebar lists files under "OPEN EDITORS" and "OBJ Z LAB TASK 4". The main editor window displays the following Java code:

```
public class mathabs {
    public static void main(String[] args) {
        System.out.println(Math.abs(-4.7));
    }
}
```

The "TERMINAL" tab shows the command run: PS E:\OBJ Z lab task 4 & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '--XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\V\P\AppData\Roaming\Code\User\workspaceStorage\d67226fb1cf0eaec971bc3fd885877\redhat.java\jdt_ws\OBJ Z lab task 4.62a910b5\bin' 'mathabs'. The output is 4.7. The status bar at the bottom indicates "java: Ready".

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

The screenshot shows the Eclipse IDE interface with the title bar "OBJ Z lab task 4". The left sidebar lists files under "OPEN EDITORS" and "OBJ Z LAB TASK 4". The main editor window displays the following Java code:

```
public class mathmax {
    public static void main(String[] args) {
        System.out.println(Math.max(5, 10));
    }
}
```

The "TERMINAL" tab shows the command run: PS E:\OBJ Z lab task 4 & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '--XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\V\P\AppData\Roaming\Code\User\workspaceStorage\d67226fb1cf0eaec971bc3fd885877\redhat.java\jdt_ws\OBJ Z lab task 4.62a910b5\bin' 'mathmax'. The output is 10. The status bar at the bottom indicates "java: Ready".

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

The screenshot shows a Java code editor interface with a dark theme. In the center, there is an open editor for a file named `mathmin.java`. The code contains a single class definition:`public class mathmin {
 public static void main(String[] args) {
 System.out.println(Math.min(a: 5, b: 10));
 }
}`

Below the editor, the terminal window shows the output of running the program:`PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '>xx:+showodeDetailsInExceptionMessages' '-cp' 'C:\Users\VP\AppData\Roaming\Code\User\workspaceStorage\d67226fb1c7f0eaec7971bc3fd805877\redhat\java\jdt_ws\OBJ Z lab task 4_62a910b5\bin' 'mathmin'
5
PS E:\OBJ Z lab task 4>`

On the right side of the interface, there is a "Build with Agent" panel and a "SUGGESTED ACTIONS" panel.

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

The screenshot shows a Java code editor interface with a dark theme. In the center, there is an open editor for a file named `mathpow.java`. The code contains a single class definition:`public class mathpow {
 public static void main(String[] args) {
 System.out.println(Math.pow(a: 2, b: 8));
 }
}`

Below the editor, the terminal window shows the output of running the program:`PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '>xx:+showodeDetailsInExceptionMessages' '-cp' 'C:\Users\VP\AppData\Roaming\Code\User\workspaceStorage\d67226fb1c7f0eaec7971bc3fd805877\redhat\java\jdt_ws\OBJ Z lab task 4_62a910b5\bin' 'mathpow'
256.0
PS E:\OBJ Z lab task 4>`

On the right side of the interface, there is a "Build with Agent" panel and a "SUGGESTED ACTIONS" panel.

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

```
public class mathrandom {  
    public static void main(String[] args) {  
        System.out.println(Math.random());  
    }  
}
```

PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\VP\AppData\Roaming\Code\User\workspaceStorage\d67226fb1c7f0eaec797bc3fd885877\redhat.java\jdt-ws\OBJ Z lab task 4.62a910b5\bin' 'mathrandom'
0.4008264572942845
PS E:\OBJ Z lab task 4>

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 $\approx 0.40088\dots$).

```
public class mathsqrt {  
    public static void main(String[] args) {  
        System.out.println(Math.sqrt(8.0));  
    }  
}
```

PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\VP\AppData\Roaming\Code\User\workspaceStorage\d67226fb1c7f0eaec797bc3fd885877\redhat.java\jdt-ws\OBJ Z lab task 4.62a910b5\bin' 'mathsqrt'
8.0
PS E:\OBJ Z lab task 4>

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 $\approx 0.40088\dots$).

A screenshot of the Visual Studio Code (VS Code) interface. The title bar says "Q. OBJ Z lab task 4". The left sidebar shows an "EXPLORER" view with files like bolee.java, charlen.java, mathabs.java, and num.java. The main editor area contains the following Java code:

```
public class num {
    public static void main(String[] args) {
        String x = "10";
        String y = "20";
        String z = x + y;
        System.out.println(z);
    }
}
```

The terminal at the bottom shows the output of running the program: "1020". A context menu is open over the code, listing suggestions such as Run: boolee..., Run: boolex..., Run: concat..., Run: concatex..., Run: Concret..., Run: finding..., Run: helloo..., Run: itsalright..., Run: mathabs..., Run: mathm...", Run: mathp...", Run: maths...", and Run: num...".

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

A screenshot of the Visual Studio Code (VS Code) interface. The title bar says "Q. OBJ Z lab task 4". The left sidebar shows an "EXPLORER" view with files like bolee.java, charlen.java, mathabs.java, and number.java. The main editor area contains the following Java code:

```
public class number {
    public static void main(String[] args) {
        int x = 10;
        int y = 20;
        int z = x + y;
        System.out.println(z);
    }
}
```

The terminal at the bottom shows the output of running the program: "30". A context menu is open over the code, listing suggestions such as Run: boolee..., Run: boolex..., Run: charlen..., Run: compa..., Run: concat..., Run: concatex..., Run: Concret..., Run: finding..., Run: helloo..., Run: itsalright..., Run: mathabs..., Run: mathm...", Run: mathp...", Run: maths...", and Run: num...".

Adding two variables usig + operator .

Add two variables using + operator and print as z.

The screenshot shows the VS Code interface with the following details:

- File Bar:** File, Edit, Selection, View, ...
- Editor:** numc.java (highlighted in the Explorer)
- Code:**

```
public class numc {
    public static void main(String[] args) {
        String x = "10";
        int y = 20;
        String z = x + y;
        System.out.println(z);
    }
}
```
- Terminal:** PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '--showCodeDetailsInExceptionMessages' '-cp' 'C:/Users/VIP/AppData/Roaming/Code/User/workspaceStorage/d67226fb1cf7feacc971bc3fd8d85e77/redhat.java/jdt_ws/OBJ Z lab task 4_62a9105/bin' 'number'
- Suggested Actions:** A sidebar on the right lists actions such as Run: boolex, Run: boolex, Run: charlen, Run: compa..., Run: concat, Run: concatex, Run: concet..., Run: Concret..., Run: finding..., Run: helooo, Run: itsalright, Run: mathabs, Run: mathm..., Run: mathm..., Run: mathra..., Run: maths..., Run: Run: num, Run: number.
- Bottom Status:** Ln 1, Col 1 | Spaces: 2 | UTF-8 | CRLF | () Java | Go Live

The screenshot shows the VS Code interface with the following details:

- File Bar:** File, Edit, Selection, View, ...
- Editor:** random.java (highlighted in the Explorer)
- Code:**

```
public class random {
    public static void main(String[] args) {
        int randomnum = (int)(Math.random() * 101); // 0 to 100
        System.out.println(randomnum);
    }
}
```
- Terminal:** PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '--showCodeDetailsInExceptionMessages' '-cp' 'C:/Users/VIP/AppData/Roaming/Code/User/workspaceStorage/d67226fb1cf7feacc971bc3fd8d85e77/redhat.java/jdt_ws/OBJ Z lab task 4_62a9105/bin' 'number'
- Suggested Actions:** A sidebar on the right lists actions such as Run: boolex, Run: boolex, Run: charlen, Run: compa..., Run: concat, Run: concatex, Run: concet..., Run: Concret..., Run: finding..., Run: helooo, Run: itsalright, Run: mathabs, Run: mathm..., Run: mathm..., Run: mathra..., Run: maths..., Run: Run: num, Run: number.
- Bottom Status:** Ln 1, Col 1 | Spaces: 2 | UTF-8 | CRLF | () Java | Go Live

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

```

1 public class rmwwhite {
2     public static void main(String[] args) {
3         String txt = "Hello world ";
4         System.out.println("Before: [" + txt + "]");
5         System.out.println("After: [" + txt.trim() + "]");
6     }
7 }

```

PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '--XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\VHP\AppData\Roaming\Code\User\workspaceStorage\d67226fb1c7f0eac971bc3fd885877\redhat.java\jdt-ws\OBJ Z lab task 4.62a910b5\bin' 'rmwwhite'
before: [Hello world]
After: [Hello world]
PS E:\OBJ Z lab task 4>

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.

```

1 public class rounding {
2     public static void main(String[] args) {
3         System.out.println(Math.round(4.6));
4         System.out.println(Math.ceil(4.1));
5         System.out.println(Math.floor(4.9));
6     }
7 }

```

PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '--XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\VHP\AppData\Roaming\Code\User\workspaceStorage\d67226fb1c7f0eac971bc3fd885877\redhat.java\jdt-ws\OBJ Z lab task 4.62a910b5\bin' 'rounding'
5
5.0
4.0
PS E:\OBJ Z lab task 4>

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.

```

public class stlen {
    public static void main(String[] args) {
        String txt = "ABCDEFGHIJKLMNPQRSTUVWXYZ";
        System.out.println("The length of the txt string is: " + txt.length());
    }
}

```

Build with Agent

SUGGESTED ACTIONS

- Run: compa...
- Run: concat...
- Run: concate...
- Run: concet...
- Run: Conct...
- Run: finding...
- Run: helloo...
- Run: itsalright...
- Run: mathabs...
- Run: mathm...
- Run: mathm...
- Run: mathp...
- Run: mathra...
- Run: maths...
- Run: num...
- Run: number...
- Run: rmwhe...
- Run: roundi...
- Run: stlen...

Using `length()` we know every sentence word length .

```

public class uplow {
    public static void main(String[] args) {
        String txt = "Hello World";
        System.out.println(txt.toUpperCase());
        System.out.println(txt.toLowerCase());
    }
}

```

Build with Agent

SUGGESTED ACTIONS

- Run: concat...
- Run: concate...
- Run: concet...
- Run: Conct...
- Run: finding...
- Run: helloo...
- Run: itsalright...
- Run: mathabs...
- Run: mathm...
- Run: mathm...
- Run: mathp...
- Run: mathra...
- Run: maths...
- Run: num...
- Run: number...
- Run: rmwhe...
- Run: roundi...
- Run: stlen...
- Run: uplow...

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).

The screenshot shows the Visual Studio Code interface with the following details:

- File Bar:** File, Edit, Selection, View, ...
- Search Bar:** Q OBJ Z lab task 4
- Explorer:**
 - OPEN EDITORS: bolee.java, charlen.java, mathabs.java, wearevikings.java
 - OBJ Z LAB TASK 4: mathrandom.class, mathrandom.java, mathsqt.class, mathsqt.java, num.class, num.java, number.class, number.java, numc.class, numc.java, random.class, random.java, rmwwhite.class, rmwwhite.java, rounding.class, rounding.java, sten.class, sten.java, uplow.class, uplow.java, wearevikings.class, wearevikings.java
- Terminal:**

```
PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '--showCodeDetailsInExceptionMessages' '-cp' 'C:\Users\VHP\AppData\Roaming\Code\User\workspaceStorage\d67226fb1c7f0eac7971bc3fd885877\redhat\java\jdt_ws\OBJ Z lab task 4_62a910b5\bin' 'wearevikings'
We are the so-called "Vikings" from the north.
PS E:\OBJ Z lab task 4>
```
- Suggested Actions:**
 - Run: concat...
 - Run: concet...
 - Run: Concret...
 - Run: finding...
 - Run: helloo...
 - Run: itsalright...
 - Run: mathabs...
 - Run: mathm...
 - Run: mathp...
 - Run: mathr...
 - Run: mathst...
 - Run: mathu...
 - Run: num...
 - Run: Run: number...
 - Run: rmwh...
 - Run: roundi...
 - Run: sten...
 - Run: uplow...
 - Run: wearev...
- Bottom Status:** Ln 7, Col 1 Spaces: 2 UTF-8 CRLF () Java ⚡ Go Live

Print string txt variable sentence in print line .

The screenshot shows the Visual Studio Code interface with the following details:

- File Bar:** File, Edit, Selection, View, ...
- Search Bar:** Q OBJ Z lab task 4
- Explorer:**
 - OPEN EDITORS: bolee.java, charlen.java, mathabs.java, wearevikings.java, ifconduc.java
 - OBJ Z LAB TASK 4: concat.java, concatex.class, concatex.java, concatenation.class, concatenation.java, Concatenaration.class, Concatenaration.java, findingcc.class, findingcr.java, helloo.class, helloo.java, ifconduc.java, itsalright.class, itsalright.java, mathabs.class, mathabs.java, mathmax.class, mathmax.java, mathmin.class, mathmin.java, mathpow.class, mathpow.java
- Terminal:**

```
PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '--showCodeDetailsInExceptionMessages' '-cp' 'C:\Users\VHP\AppData\Roaming\Code\User\workspaceStorage\d67226fb1c7f0eac7971bc3fd885877\redhat\java\jdt_ws\OBJ Z lab task 4_62a910b5\bin' 'ifconduc'
Bring an umbrella!
PS E:\OBJ Z lab task 4>
```
- Suggested Actions:**
 - Run: concat...
 - Run: concet...
 - Run: Concret...
 - Run: finding...
 - Run: helloo...
 - Run: itsalright...
 - Run: mathabs...
 - Run: mathm...
 - Run: mathp...
 - Run: mathr...
 - Run: mathst...
 - Run: mathu...
 - Run: num...
 - Run: Run: number...
 - Run: rmwh...
 - Run: roundi...
 - Run: sten...
 - Run: uplow...
 - Run: wearev...
 - Run: ifconduc...
- Bottom Status:** Ln 3, Col 30 Spaces: 4 UTF-8 CRLF () Java ⚡ Go Live

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

```
File Edit Selection View ... < > Q OBJ Z lab task 4 File Explorer OPEN EDITORS ifcompair.java charlen.java mathbs.java wearevikings.java ifconduc.java ifcompair.java Run Debug public static void main(String[] args) { if (20 > 18) { System.out.println("20 is greater than 18"); // obviously } } PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS CHAT Build with Agent AI responses may be inaccurate. Generate Agent Instructions to onboard AI onto your codebase. SUGGESTED ACTIONS Build Workspace Show Config 0 + J ifcompair.java Describe what to build next Agent Auto Go Live Ln 1, Col 18 Spaces: 4 UTF-8 CRLF Java Go Live
```

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

```
File Edit Selection View ... < > Q OBJ Z lab task 4 File Explorer OPEN EDITORS ifvericom.java mathbs.java wearevikings.java ifconduc.java ifcompair.java ifvericom.java Run Debug public class ifvericom { public static void main(String[] args) { int x = 20; int y = 18; if (x > y) { System.out.println("x is greater than y"); } } PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS CHAT Build with Agent AI responses may be inaccurate. Generate Agent Instructions to onboard AI onto your codebase. SUGGESTED ACTIONS Build Workspace Show Config 0 + J ifvericom.java Describe what to build next Agent Auto Go Live Ln 2, Col 25 Spaces: 4 UTF-8 CRLF Java Go Live
```

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

The screenshot shows the VS Code interface with the following details:

- File Bar:** File, Edit, Selection, View, ...
- Search Bar:** Q OBJ Z lab task 4
- Explorer:** Shows files like bolee.java, charlen.java, mathabs.java, wearevikings.java, ifconduc.java, ifcompair.java, ifvericom.java, and ifex.java.
- Editor:** The ifex.java file is open, containing the following code:

```
1 public class ifex {  
2     Run Debug  
3     public static void main(String[] args) {  
4         int x = 20;  
5         int y = 20;  
6         if (x == y) {  
7             System.out.println(x: "x is equal to y");  
8         }  
9     }  
10 }
```
- Terminal:** Shows the command PS E:\OBJ Z lab task 4 & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' 'xx:+showCodeDetailsInExceptionMessages' '-cp' 'C:/Users/VIP/AppData/Roaming/Code/User/workspaceStorage/d67226fb1c7f0eaecc971bc3fd885877\redhat.java\jdt_ws\OBJ Z lab task 4_62a910b5\bin' 'ifex'
x is equal to y
PS E:\OBJ Z lab task 4>
- Suggested Actions:** A dropdown menu with various run configurations for the ifex.java file.
- Bottom Status Bar:** Ln 10, Col 1 | Spaces: 4 | UTF-8 | CRLF | () Java | Go Live

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

The screenshot shows the VS Code interface with the following details:

- File Bar:** File, Edit, Selection, View, ...
- Search Bar:** Q OBJ Z lab task 4
- Explorer:** Shows files like bolee.java, charlen.java, mathabs.java, wearevikings.java, ifconduc.java, ifcompair.java, ifvericom.java, ifex.java, and ifbovar.java.
- Editor:** The ifbovar.java file is open, containing the following code:

```
1 public class ifbovar {  
2     Run Debug  
3     public static void main(String[] args) {  
4         boolean isLightOn = true;  
5         if (isLightOn) {  
6             System.out.println(x: "The light is on.");  
7         }  
8     }  
9 }
```
- Terminal:** Shows the command PS E:\OBJ Z lab task 4 & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' 'xx:+showCodeDetailsInExceptionMessages' '-cp' 'C:/Users/VIP/AppData/Roaming/Code/User/workspaceStorage/d67226fb1c7f0eaecc971bc3fd885877\redhat.java\jdt_ws\OBJ Z lab task 4_62a910b5\bin' 'ifbovar'
The light is on.
PS E:\OBJ Z lab task 4>
- Suggested Actions:** A dropdown menu with various run configurations for the ifbovar.java file.
- Bottom Status Bar:** Ln 1, Col 18 | Spaces: 4 | UTF-8 | CRLF | () Java | Go Live

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

```

File Edit Selection View ...
EXPLORER OPEN EDITORS
J Ifconduc.java J Ifcompair.java J Ifvericom.java J Ifex.java J Ifbovar.java J Ifbrc.java
J charlen.java J mathabs.java J wearavikings.java J Ifconduc.java J Ifcompair.java J Ifvericom.java
J Ifex.java J Ifbovar.java J Ifbrc.java
OBJ Z LAB TASK 4
concatenation.class
concatenation.java
J Concatenaration.class
J Concatenaration.java
J findingcrc.class
J findingcrc.java
J hellos.class
J hellos.java
J Ifbovar.java
J Ifbrc.java
J Ifcompair.java
J Ifconduc.java
J Ifex.java
J Ifvericom.java
J itsalright.class
J itsalright.java
J mathabs.class
J mathabs.java
> OUTLINE
> TIMELINE
> JAVA PROJECTS
Java: Ready

```

```

ifbrc.java > Ifbrc > main(String[])
1 public class Ifbrc {
2     public static void main(String[] args) {
3         if (20 > 18)
4             System.out.println("20 is greater than 18");
5     }
6 }
7

PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '--XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\V\P\AppData\Roaming\Code\User\workspaceStorage\d67226fb1c7f0eac9791bc3fd885877\redhat.java\jdt-ws\OBJ Z lab task 4_62a9105\bin' 'Ifbrc'
20 is greater than 18
PS E:\OBJ Z lab task 4>

```

Build with Agent

AI responses may be inaccurate.
Generate Agent Instructions to onboard AI onto your codebase.

SUGGESTED ACTIONS

Build Workspace Show Config

0 + J Ifbrc.java
Describe what to build next
Agent Auto

Ln 3, Col 17 Spaces: 4 UTF-8 CRLF (Java Go Live

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

```

File Edit Selection View ...
EXPLORER OPEN EDITORS
J mathabs.java J wearavikings.java J Ifconduc.java J Ifcompair.java J Ifvericom.java J Ifex.java J Ifbovar.java J Ifbrc.java
J Ifpotpob.java
OBJ Z LAB TASK 4
findingcrc.class
J findingcrc.java
J hellos.class
J hellos.java
J Ifbovar.java
J Ifbrc.java
J Ifcompair.java
J Ifconduc.java
J Ifex.java
J Ifpotpob.java
J Ifvericom.java
J itsalright.class
J itsalright.java
J mathabs.class
J mathabs.java
J mathmax.class
J mathmax.java
J mathmin.class
> OUTLINE
> TIMELINE
> JAVA PROJECTS
Java: Ready

```

```

ifpotpob.java > Ifpotpob
1 public class Ifpotpob {
2     Run Debug
3     public static void main(String[] args) {
4         int x = 20;
5         int y = 18;
6
7         if (x > y)
8             System.out.println("x is greater than y");
9         System.out.println("this line runs no matter what (not part of the if statement)");
10    }
11

PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '--XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\V\P\AppData\Roaming\Code\User\workspaceStorage\d67226fb1c7f0eac9791bc3fd885877\redhat.java\jdt-ws\OBJ Z lab task 4_62a9105\bin' 'Ifpotpob'
x is greater than y
This line runs no matter what (not part of the if statement)
PS E:\OBJ Z lab task 4>

```

Build with Agent

AI responses may be inaccurate.
Generate Agent Instructions to onboard AI onto your codebase.

SUGGESTED ACTIONS

Build Workspace Show Config

0 + J Ifpotpob.java
Describe what to build next
Agent Auto

Ln 1, Col 18 Spaces: 4 UTF-8 CRLF (Java Go Live

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.

```

1 public class Ifavmis {
2     public static void main(String[] args) {
3         int x = 20;
4         int y = 18;
5
6         if (x > y) {
7             System.out.println(x: "x is greater than y");
8             System.out.println(x: "Both lines are part of the if");
9         }
10
11     System.out.println(x: "I am outside if, not part of if!");
}

```

The terminal output shows:

```

PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '.\objzlabtask4\ifavmis'
x is greater than y
both lines are part of the if
I am outside if, not part of if!

```

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.

```

1 public class Elsesex {
2     public static void main(String[] args) {
3         boolean isRaining = false;
4
5         if (isRaining) {
6             System.out.println(x: "Bring an umbrella!");
7         } else {
8             System.out.println(x: "No rain today, no need for an umbrella!");
9         }
10    }
}

```

The terminal output shows:

```

PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '.\objzlabtask4\elsesex'
No rain today, no need for an umbrella!

```

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.

The screenshot shows a Java code editor with the following code:

```
public class elseifex {
    public static void main(String[] args) {
        int weather = 2; // 1 = raining, 2 = sunny, 3 = cloudy
        if (weather == 1) {
            System.out.println("Bring an umbrella.");
        } else if (weather == 2) {
            System.out.println("Wear sunglasses.");
        } else {
            System.out.println("Just go outside normally.");
        }
    }
}
```

The code is run in a terminal window, and the output is:

```
PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-pr
odeDetailsInExceptionMessages' '-cp' 'C:\Users\HP\AppData\Roaming\Code\User\workspa
1c7f0eaec971bc3fd885877\redhat.java\jdt_ws\OBJ Z lab task 4_62a910b5\bin' 'elseif
Wear sunglasses.
PS E:\OBJ Z lab task 4>
```

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."

The screenshot shows a Java code editor interface with the title bar "Q, OBJ Z lab task 4". The left sidebar has sections for "EXPLORER", "OPEN EDITORS", and "OBJ Z LAB TASK 4". In the "OPEN EDITORS" section, there are multiple tabs for files like ifvericom.java, ifex.java, ifbovar.java, ifrc.java, ifpotpob.java, ifavmis.java, elsesex.java, elseifex.java, and elseift.java. The "OBJ Z LAB TASK 4" section contains files such as concat.class, concat.java, concatex.class, concatex.java, concatenation.class, Concatenation.java, Concatenation.class, findingorc.class, findingorc.java, helloc.class, helloc.java, ifavmis.java, ifbovar.java, ifrc.java, and ifvericom.java. The main editor area displays the "elseift.java" code:

```
1 public class elseift {
2     Run | Debug
3     public static void main(String[] args) {
4         int time = 22;
5         if (time < 10) {
6             System.out.println("Good morning.");
7         } else if (time < 18) {
8             System.out.println("Good day.");
9         } else {
10             System.out.println("Good evening.");
11         }
12     }
13 }
```

The terminal below shows the output of running the program: "Good evening."

The right side of the interface includes a "CHAT" section with a speech bubble icon, a "Build with Agent" section with a message "Build workspace", and a "SUGGESTED ACTIONS" panel with various build options.

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."

The screenshot shows a Java code editor interface with the title bar "Q, OBJ Z lab task 4". The left sidebar has sections for "EXPLORER", "OPEN EDITORS", and "OBJ Z LAB TASK 4". In the "OPEN EDITORS" section, there are multiple tabs for files like ifex.java, ifbovar.java, ifrc.java, ifpotpob.java, ifavmis.java, elsesex.java, elseifex.java, elseift.java, and shifelse.java. The "OBJ Z LAB TASK 4" section contains files such as number.class, number.java, num.class, num.java, random.class, random.java, rmwhite.class, rmwhite.java, rounding.class, rounding.java, shifelse.java, stlen.class, stlen.java, uplow.class, uplow.java, wearevikings.class, and wearevikings.java. The main editor area displays the "shifelse.java" code:

```
1 public class shifelse {
2     Run | Debug
3     public static void main(String[] args) {
4         int time = 20;
5         if (time < 18) {
6             System.out.println("Good day.");
7         } else {
8             System.out.println("Good evening.");
9         }
10    }
11 }
```

The terminal below shows the output of running the program: "Good evening."

The right side of the interface includes a "CHAT" section with a speech bubble icon, a "Build with Agent" section with a message "Build workspace", and a "SUGGESTED ACTIONS" panel with various build options.

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."

```
public class tmpelij {
    public static void main(String[] args) {
        int time = 20;
        System.out.println((time < 18) ? "Good day." : "Good evening.");
    }
}
```

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

```
public class naster {
    public static void main(String[] args) {
        int time = 22;
        String message = (time < 12) ? "Good morning."
                                     : (time < 18) ? "Good afternoon."
                                     : "Good evening.";
        System.out.println(message);
    }
}
```

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

The screenshot shows a Java code editor interface with the following details:

- File Path:** nasif.java
- Code Content:**

```
public class nasif {
    public static void main(String[] args) {
        int x = 15;
        int y = 25;

        if (x > 10) {
            System.out.println(x: "x is greater than 10");
            // Nested if
            if (y > 20) {
                System.out.println(x: "y is also greater than 20");
            }
        }
    }
}
```

- Terminal Output:**

```
PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '--showCodeDetailsInExceptionMessages' '-cp' 'C:\Users\VH\appData\Roaming\Code\User\workspaceStorage\d67226fb1c\feaecc971bc3fd8d5877\redhat.java\jdt_ws\OBJ Z lab task 4_62a910b5\bin' 'nasif'
x is greater than 10
y is also greater than 20
PS E:\OBJ Z lab task 4>
```

- Suggested Actions:** A sidebar on the right lists various run configurations for Java code.
- Build with Agent:** A feature panel on the right allows building the workspace with AI assistance.

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.

The screenshot shows a Java development environment with the following details:

- File Menu:** File, Edit, Selection, View, ...
- Search Bar:** Q OBJ Z lab task 4
- Explorer:** Shows a list of Java files under "OPEN EDITORS" and "OBJ Z LAB TASK 4".
- Editor:** Displays the content of `relnas.java`. The code checks if a user is old enough (≥ 18) and a citizen to vote.
- Terminal:** Shows the output of running the Java code, indicating the user is old enough to vote and is a citizen.

```
public class relnas {
    public static void main(String[] args) {
        int age = 20;
        boolean iscitizen = true;

        if (age >= 18) {
            System.out.println("Old enough to vote.");

            if (iscitizen) {
                System.out.println("And you are a citizen, so you can vote!");
            } else {
                System.out.println("But you are not a citizen.");
            }
        }
    }
}
```

```
PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enableassertionDetailsInExceptionMessages' '-cp' 'C:\Users\HP\AppData\Roaming\Code\User\workspaceStorage\1c7f0eaec7971bc3fd885877\redhat.java\jdt_ws\OBJ Z lab task 4_62a910b5\bin' 'relnas'
Old enough to vote.
And you are a citizen, so you can vote!
PS E:\OBJ Z lab task 4>
```

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

The screenshot shows the VS Code interface with the following details:

- File Bar:** File, Edit, Selection, View, ...
- Title Bar:** Q OBJ Z lab task 4
- Explorer:** Shows files like ifavmis.java, elseif.java, etc., under OPEN EDITORS and OBJ Z LAB TASK 4.
- Terminal:** Displays the command PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '>objzlabtask4'. It also shows the output of the program: "old enough to vote." and "And you are a citizen, so you can vote!"
- Suggested Actions:** A dropdown menu with options like Run: stlen, Run: uplow, etc.
- Build with Agent:** A section with a message about AI responses being inaccurate and a link to Generate Agent Instructions.
- Bottom Status Bar:** Ln 1, Col 18, Spaces: 4, UTF-8, CRLF, Java, Go Live, Java Ready.

- 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.

The screenshot shows the VS Code interface with the following details:

- File Bar:** File, Edit, Selection, View, ...
- Title Bar:** Q OBJ Z lab task 4
- Explorer:** Shows files like elseif.java, shifelse.java, etc., under OPEN EDITORS and OBJ Z LAB TASK 4.
- Terminal:** Displays the command PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '>objzlabtask4'. It shows the output: "Both conditions are true".
- Suggested Actions:** A dropdown menu with options like Run: uplow, Run: wearev..., etc.
- Build with Agent:** A section with a message about AI responses being inaccurate and a link to Generate Agent Instructions.
- Bottom Status Bar:** Ln 1, Col 18, Spaces: 4, UTF-8, CRLF, Java, Go Live, Java Ready.

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

The screenshot shows the Visual Studio Code interface with the following details:

- File Bar:** File, Edit, Selection, View, ...
- Title Bar:** Q. OBJ Z lab task 4
- Explorer:** Shows a list of Java files under "OBJ Z LAB TASK 4".
- Editor:** Displays the content of `or.java`. The code is:

```
public class or {
    public static void main(String[] args) {
        int a = 200;
        int b = 33;
        int c = 500;

        if (a > b || a > c) {
            System.out.println("At least one condition is true");
        }
    }
}
```
- Terminal:** Shows the command run and its output:

```
PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '.\xx;showCodeDetailsInExceptionMessages' '-cp' 'C:\Users\HP\AppData\Roaming\Code\User\workspaceStorage\d67226fb1c\feaeac971bc3fd885877\redhat.java\jdt_ws\OBJ Z lab task 4_62a910b5\bin' 'or'
At least one condition is true
PS E:\OBJ Z lab task 4>
```
- Suggested Actions:** A sidebar on the right lists various Java operations like Run: wearever, Run: ifcondic, etc.
- Bottom Status Bar:** Ln 1, Col 18, Spaces: 4, UTF-8, CRLF, Java, Go Live, etc.

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

```

1 public class not {
2     public static void main(String[] args) {
3         int a = 33;
4         int b = 269;
5
6         if (!(a > b)) {
7             System.out.println("a is NOT greater than b");
8         }
9     }
10 }
11

```

PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '--showCodeDetailsInExceptionMessages' '-cp' 'C:\Users\HP\AppData\Roaming\Code\User\workspaceStorage\d67226fb1c7f0eac9797bc3fd885877\redhat.java\jdt_ws\OBJ Z lab task 4.62a910b5\bin' 'not'
a is NOT greater than b
PS E:\OBJ Z lab task 4>

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.

```

1 public class relli {
2     public static void main(String[] args) {
3         boolean isLoggedIn = true;
4         boolean isAdmin = false;
5         int securityLevel = 3; // 1 = highest
6
7         if (isLoggedIn && (isAdmin || securityLevel <= 2)) {
8             System.out.println("Access granted");
9         } else {
10             System.out.println("Access denied");
11         }
12     }
13 }
14

```

PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '--showCodeDetailsInExceptionMessages' '-cp' 'C:\Users\HP\AppData\Roaming\Code\User\workspaceStorage\d67226fb1c7f0eac9797bc3fd885877\redhat.java\jdt_ws\OBJ Z lab task 4.62a910b5\bin' 'relli'
Access denied
PS E:\OBJ Z lab task 4>

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

The screenshot shows an IDE interface with the title bar "OBJ Z lab task 4". The left sidebar lists files under "EXPLORER" and "OBJ Z LAB TASK 4". The main editor window displays the following Java code:

```
public class real {
    public static void main(String[] args) {
        int doorCode = 1337;

        if (doorCode == 1337) {
            System.out.println("Correct code. The door is now open.");
        } else {
            System.out.println("Wrong code. The door remains closed.");
        }
    }
}
```

The terminal window at the bottom shows the output of running the code:

```
PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' 'xx:showCodeDetailsInExceptionMessages' '-cp' 'C:\Users\VHP\AppData\Roaming\Code\User\workspaceStorage\d67226fb1c7f0eaec7971bc3fd885877\redhat.java\jdt_ws\OBJ Z lab task 4_62e910b5\bin'\real'
Correct code. The door is now open.
```

The status bar at the bottom right indicates "Java: Ready".

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

```

File Edit Selection View ...
File Explorer Problems Output Debug Console Terminal Ports
OPEN EDITORS posneg.java
OBJ Z LAB TASK 4
posneg.java > posneg
1 public class posneg {
2     Run | Debug
3     public static void main(String[] args) {
4         int myNum = 10; // Is this a positive or negative number?
5
6         if (myNum > 0) {
7             System.out.println("The value is a positive number.");
8         } else if (myNum < 0) {
9             System.out.println("The value is a negative number.");
10        } else {
11            System.out.println("The value is 0.");
12        }
13    }
14
15 PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:/Users/VIP/AppData/Roaming/Code/User/workspaceStorage/d67226fb1c7f0eac9791bc3fd885877/redhat.java/jdt_ws/OBJ Z lab task 4_62a9105/bin' 'posneg'
16 The value is a positive number.
17 PS E:\OBJ Z lab task 4>

```

Build with Agent

AI responses may be inaccurate.
Generate Agent Instructions to onboard AI onto your codebase.

SUGGESTED ACTIONS

Build Workspace Show Config

Run: ifex Run: ifbovar Run: ifrc Run: ifpotpob Run: ifavmis Run: elsesex Run: elseifex Run: elseift Run: shifelse Run: tmplif Run: naster Run: nafis Run: relnas Run: and Run: or Run: not Run: reli Run: real Run: posneg

Describe what to build next

Agent Auto Go Live

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

```

File Edit Selection View ...
File Explorer Problems Output Debug Console Terminal Ports
OPEN EDITORS old.java
OBJ Z LAB TASK 4
old.java > old
1 public class old {
2     Run | Debug
3     public static void main(String[] args) {
4         int myAge = 25;
5         int votingAge = 18;
6
7         if (myAge >= votingAge) {
8             System.out.println("Old enough to vote!");
9         } else {
10            System.out.println("Not old enough to vote.");
11        }
12    }
13
14 PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:/Users/VIP/AppData/Roaming/Code/User/workspaceStorage/d67226fb1c7f0eac9791bc3fd885877/redhat.java/jdt_ws/OBJ Z lab task 4_62a9105/bin' 'old'
15 Old enough to vote!
16 PS E:\OBJ Z lab task 4>

```

Build with Agent

AI responses may be inaccurate.
Generate Agent Instructions to onboard AI onto your codebase.

SUGGESTED ACTIONS

Build Workspace Show Config

Run: ifbovar Run: ifrc Run: ifpotpob Run: ifavmis Run: elsesex Run: elseifex Run: elseift Run: shifelse Run: tmplif Run: naster Run: nafis Run: relnas Run: and Run: or Run: not Run: reli Run: real Run: posneg Run: old

Describe what to build next

Agent Auto Go Live

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

The screenshot shows a Java development environment with the following details:

- File Menu:** File, Edit, Selection, View, ...
- Toolbar:** Back, Forward, Search, Chat, etc.
- Explorer:** Shows a list of Java files in the workspace, including `citi.java`.
- Editor:** Displays the code for `citi.java`:


```
1 public class citi {
2     public static void main(String[] args) {
3         int age = 20;
4         boolean isCitizen = true;
5
6         if (age >= 18) {
7             System.out.println("Old enough to vote.");
8
9             if (isCitizen) {
10                 System.out.println("And you are a citizen, so you can vote!");
11             } else {
12             }
13         }
14     }
15 }
```
- Terminal:** Shows the command-line output of running the program:


```
PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:/Users/VIP/AppData/Roaming/Code/User/workspaceStorage/d67226fb1c7f0eac971bc3fd885877/redhat.java/jdt_ws/OBJ Z lab task 4.62a9105/bin' 'citi'
Old enough to vote.
And you are a citizen, so you can vote!
```
- Suggested Actions:** A dropdown menu with various run configurations like `Run: ifrc`, `Run: ifpotpb`, etc.
- Build with Agent:** A feature panel for generating AI responses.
- Bottom Bar:** Includes tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, PORTS, and other status indicators.

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

The screenshot shows a Java development environment with the following details:

- File Menu:** File, Edit, Selection, View, ...
- Toolbar:** Back, Forward, Search, Chat, etc.
- Explorer:** Shows a list of Java files in the workspace, including `evenodd.java`.
- Editor:** Displays the code for `evenodd.java`:


```
1 public class evenodd {
2     public static void main(String[] args) {
3         int myNum = 5;
4
5         if (myNum % 2 == 0) {
6             System.out.println(myNum + " is even");
7         } else {
8             System.out.println(myNum + " is odd");
9         }
10    }
11 }
```
- Terminal:** Shows the command-line output of running the program:


```
PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:/Users/VIP/AppData/Roaming/Code/User/workspaceStorage/d67226fb1c7f0eac971bc3fd885877/redhat.java/jdt_ws/OBJ Z lab task 4.62a9105/bin' 'evenodd'
5 is odd
```
- Suggested Actions:** A dropdown menu with various run configurations like `Run: ifrc`, `Run: ifpotpb`, etc.
- Build with Agent:** A feature panel for generating AI responses.
- Bottom Bar:** Includes tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, PORTS, and other status indicators.

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 screenshot of a Java code editor interface. The central area shows a code editor with the following Java code:

```
public class temp {
    public static void main(String[] args) {
        int temperature = 30;
        if (temperature < 0) {
            System.out.println("It's freezing!");
        } else if (temperature < 20) {
            System.out.println("It's cool.");
        } else {
            System.out.println("It's warm.");
        }
    }
}
```

The code is run in a terminal window, producing the output:

```
PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '--XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\VHP\AppData\Roaming\Code\User\workspaceStorage\d67226fb1cf0eaec971bc3fd885877\redhat.java\jdt_ws\OBJ Z lab task 4.62a910b5\bin' 'temp'
It's warm.
```

The terminal window also shows the path: PS E:\OBJ Z lab task 4>. The status bar at the bottom indicates "Java: Ready".

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

A screenshot of a Java code editor interface. The central area shows a code editor with the following Java code:

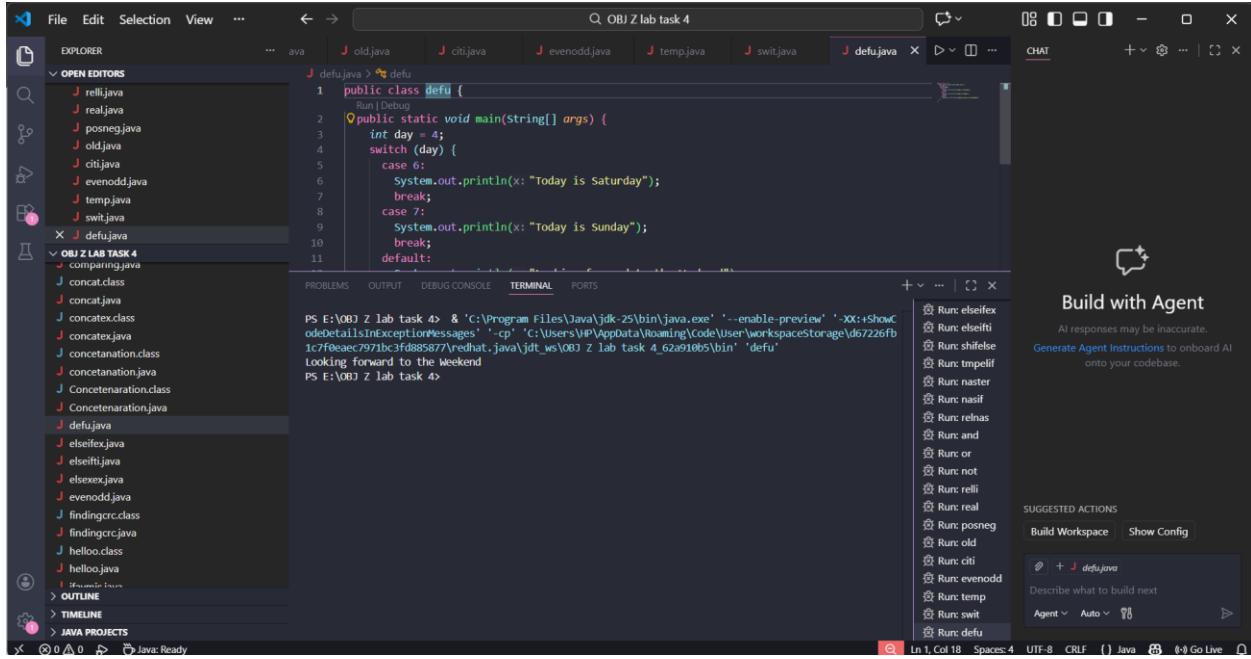
```
public class swit {
    public static void main(String[] args) {
        int day = 4;
        switch (day) {
            case 1:
                System.out.println("Monday");
                break;
            case 2:
                System.out.println("Tuesday");
                break;
            case 3:
                System.out.println("Wednesday");
                break;
        }
    }
}
```

The code is run in a terminal window, producing the output:

```
PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '--XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\VHP\AppData\Roaming\Code\User\workspaceStorage\d67226fb1cf0eaec971bc3fd885877\redhat.java\jdt_ws\OBJ Z lab task 4.62a910b5\bin' 'swit'
Thursday
```

The terminal window also shows the path: PS E:\OBJ Z lab task 4>. The status bar at the bottom indicates "Java: Ready".

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



The screenshot shows a Java application named "OBJ Z lab task 4" in the title bar. The "OPEN EDITORS" sidebar lists several Java files: relli.java, real.java, posneg.java, old.java, citi.java, evenodd.java, temp.java, swit.java, and defu.java. The main editor window displays the "defu.java" file:

```
defu.java > defu
1 public class defu {
2     public static void main(String[] args) {
3         int day = 4;
4         switch (day) {
5             case 6:
6                 System.out.println("Today is Saturday");
7                 break;
8             case 7:
9                 System.out.println("Today is Sunday");
10                break;
11         }
12     }
13 }
```

The terminal below the editor shows the command run and its output:

```
PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' 'xx:showodeDetailsInExceptionMessages' '--cp' 'C:\Users\VIP\AppData\Roaming\Code\User\workspaces\storage\d67226fb1c7f0eaec7971bc3fd885877\redhat.java\jdt_ws\OBJ Z lab task 4_62a910b5\bin' 'defu'
Looking forward to the Weekend
PS E:\OBJ Z lab task 4>
```

The bottom right corner of the interface features a "Build with Agent" feature, which includes a list of suggested actions and a "Build Workspace" button.

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

A screenshot of the Visual Studio Code (VS Code) interface. The title bar says "OBJ Z lab task 4". The left sidebar shows an "EXPLORER" view with files like real.java, posneg.java, old.java, etc., and an "OBJ Z LAB TASK 4" folder containing files such as relljava, relnas.java, rmwwhite.class, rmwwhite.java, rounding.class, rounding.java, shifelse.java, stlen.class, stlen.java, swit.java, temp.java, tmpeif.java, uplow.class, uplow.java, wearevikings.class, wearevikings.java, and whi.java. The main editor window displays a Java code snippet:

```
1 public class whi{  
2     public static void main(String[] args) {  
3         int i = 0;  
4         while (i < 5) {  
5             System.out.println(i);  
6             i++;  
7         }  
8     }  
9 }
```

The terminal below the editor shows the command PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '.\whi'. The output of the program is displayed in the terminal.

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

A screenshot of the Visual Studio Code (VS Code) interface. The title bar says "OBJ Z lab task 4". The left sidebar shows an "EXPLORER" view with files like posneg.java, old.java, etc., and an "OBJ Z LAB TASK 4" folder containing files such as comparing.java, concat.class, concat.java, concatex.class, concatex.java, concatenation.class, concatenation.java, Concatenaration.class, Concatenaration.java, couwhi.java, defu.java, elseif.java, elseif.java, elsesex.java, evenodd.java, findingrc.class, findingrc.java, helio.class, and hallo.java. The main editor window displays a Java code snippet:

```
1 public class couwhi {  
2     public static void main(String[] args) {  
3         int countdown = 3;  
4         while (countdown > 0) {  
5             System.out.println(countdown);  
6             countdown--;  
7         }  
8         System.out.println("Happy New Year!!");  
9     }  
10 }  
11 }
```

The terminal below the editor shows the command PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '.\couwhi'. The output of the program is displayed in the terminal.

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

The screenshot shows an IDE interface with the title bar "Q. OBJ Z lab task 4". The left sidebar lists several Java files under "OPEN EDITORS" and "OBJ Z LAB TASK 4". The main editor window displays the following Java code:

```
public class fals {
    public static void main(String[] args) {
        int i = 10;
        while (i < 5) {
            System.out.println("This will never be printed");
            i++;
        }
    }
}
```

The terminal tab at the bottom shows the command line output:

```
PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '--XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\HP\\AppData\Roaming\Code\User\workspaceStorage\67226fb1cf0eac7971bc3fd885877\redhat.java\jdt_ws\OBJ Z lab task 4_62a910b5\bin' 'fals'
PS E:\OBJ Z lab task 4>
```

A context menu is open over the code, listing various run configurations such as "Run: tempelf", "Run: mster", "Run: nasif", etc.

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.

The screenshot shows the Visual Studio Code interface with the following details:

- File Bar:** File, Edit, Selection, View, ...
- Title Bar:** Q. OBJ Z lab task 4
- Explorer:** Shows files in the current workspace, including `dow.java` which is open in the editor.
- Editor:** Displays the following Java code:


```
public class dow {
    public static void main(String[] args) {
        int i = 0;
        do {
            System.out.println(i);
            i++;
        } while (i < 5);
    }
}
```
- Terminal:** Shows the command PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' -cp 'C:/Users/HP/AppData/Roaming/Code/User/workspaceStorage/d67226fb1c7f0eaec971bc3fd885877/redhat.java/jdt_ws/OBJ Z lab task 4_62a910b5\bin' 'dow'
 and its output:


```
0
1
2
3
4
```
- Right Panel:**
 - Build with Agent:** A button to generate agent instructions.
 - Suggested Actions:** A list of build options for `dow.java`, such as Run: master, Run: nasif, Run: relias, etc.
 - Build Workspace:** A button to build the entire workspace.
- Bottom Status Bar:** Ln 1, Col 18, Spaces: 4, UTF-8, CRLF, Java, Go Live, Java Ready.

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

The screenshot shows the Visual Studio Code interface with the following details:

- File Bar:** File, Edit, Selection, View, ...
- Title Bar:** Q. OBJ Z lab task 4
- Explorer:** Shows files in the current workspace, including `conf.java` which is open in the editor.
- Editor:** Displays the following Java code:


```
public class conf {
    public static void main(String[] args) {
        int i = 10;
        do {
            System.out.println("i is " + i);
            i++;
        } while (i < 5);
    }
}
```
- Terminal:** Shows the command PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' -cp 'C:/Users/HP/AppData/Roaming/Code/User/workspaceStorage/d67226fb1c7f0eaec971bc3fd885877/redhat.java/jdt_ws/OBJ Z lab task 4_62a910b5\bin' 'conf'
 and its output:


```
i is 10
```
- Right Panel:**
 - Build with Agent:** A button to generate agent instructions.
 - Suggested Actions:** A list of build options for `conf.java`, such as Run: nasif, Run: relias, etc.
 - Build Workspace:** A button to build the entire workspace.
- Bottom Status Bar:** Ln 1, Col 18, Spaces: 4, UTF-8, CRLF, Java, Go Live, Java Ready.

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

```

public class realwhi {
    public static void main(String[] args) {
        int countdown = 3;
        while (countdown > 0) {
            System.out.println(countdown);
            countdown--;
        }
        System.out.println("Happy New Year!!!");
    }
}

```

PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '--showCodeDetailsInExceptionMessages' '-cp' 'C:\Users\HP\AppData\Roaming\Code\User\workspaceStorage\d67226fb1c7f0eaecc971bc3fd885877\redhat.java\jdt-ws\OBJ Z lab task 4.62a910b5\bin' 'realwhi'
3
2
1
Happy New Year!!
PS E:\OBJ Z lab task 4>

Build with Agent

AI responses may be inaccurate.
Generate Agent Instructions to onboard AI onto your codebase.

SUGGESTED ACTIONS

Build Workspace Show Config

+ J realwhi.java
Describe what to build next
Agent Auto

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

The screenshot shows the VS Code interface with the title bar "Q_OBJ Z lab task 4". The Explorer sidebar on the left lists files under "OBJ Z LAB TASK 4" and "OPEN EDITORS". The "realloop.java" file is open in the editor, displaying the following code:

```

public class realloop {
    public static void main(String[] args) {
        int dice = 1;

        while (dice <= 6) {
            if (dice < 6) {
                System.out.println("No Yatzy.");
            } else {
                System.out.println("Yatzy!");
            }
            dice = dice + 1;
        }
    }
}

```

The terminal at the bottom shows the output of running the program:

```

PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '--showCodeDetailsInExceptionMessages' '-cp' 'C:\Users\HP\AppData\Roaming\Code\User\workspaceStorage\d67226fb1c7f0eaec7971bc3fd885877\redhat.java\jdt_ws\OBJ Z lab task 4_62a910b5\bin' 'realloop'
No Yatzy.
No Yatzy.
No Yatzy.
No Yatzy.
No Yatzy.
Yatzy!

```

The right side of the interface includes a "Build with Agent" panel and a "SUGGESTED ACTIONS" sidebar.

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

The screenshot shows the VS Code interface with the title bar "Q_OBJ Z lab task 4". The Explorer sidebar on the left lists files under "OBJ Z LAB TASK 4" and "OPEN EDITORS". The "loopnum.java" file is open in the editor, displaying the following code:

```

public class loopnum {
    public static void main(String[] args) {
        for (int i = 0; i < 5; i++) {
            System.out.println(i);
        }
    }
}

```

The terminal at the bottom shows the output of running the program:

```

PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '--showCodeDetailsInExceptionMessages' '-cp' 'C:\Users\HP\AppData\Roaming\Code\User\workspaceStorage\d67226fb1c7f0eaec7971bc3fd885877\redhat.java\jdt_ws\OBJ Z lab task 4_62a910b5\bin' 'loopnum'
0
1
2
3
4

```

The right side of the interface includes a "Build with Agent" panel and a "SUGGESTED ACTIONS" sidebar.

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

The screenshot shows a Java application named "OBJ Z lab task 4" running in VS Code. The code in the editor is as follows:

```
public class loopeven {
    public static void main(String[] args) {
        for (int i = 0; i <= 10; i = i + 2) {
            System.out.println(i);
        }
    }
}
```

The terminal output shows the execution of the program:

```
PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\HP\AppData\Roaming\Code\User\workspaceStorage\d67226fb1c7f0eaec7971bc3fd885877\redhat.java\jdt_ws\OBJ Z lab task 4_62a910b5\bin' 'loopeven'
0
2
4
6
8
10
```

The sidebar on the right is titled "Build with Agent" and lists various run configurations. The "SUGGESTED ACTIONS" section includes "Build Workspace" and "Show Config".

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

The screenshot shows an IDE interface with the following details:

- File Menu:** File, Edit, Selection, View, ...
- Search Bar:** Q OBJ Z lab task 4
- Left Sidebar (OPEN EDITORS):** A list of Java files including couwhijava, fals.java, dow.java, conf.java, realwhijava, realloop.java, loopnum.java, loopeven.java, and loopsum.java.
- Current Editor:** The file "loopsum.java" is open, containing the following code:

```
public class loopsum {
    public static void main(String[] args) {
        int sum = 0;
        for (int i = 1; i <= 5; i++) {
            sum = sum + i;
        }
        System.out.println("Sum is " + sum);
    }
}
```
- Terminal:** The terminal window shows the command PS E:\OBJ Z lab task 4 & 'c:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\HP\AppData\Roaming\Code\User\workspaceStorage\d67226fb1c7f0eac9797bc3fd858577\redhat\java\jdt_ws\OBJ Z lab task 4_62a910b5\bin' 'loopsum'. The output shows "Sum is 15".
- Right Sidebar:**
 - Build with Agent:** A button to onboard the codebase.
 - AI Responses:** A note stating "AI responses may be inaccurate."
 - Generate Agent Instructions:** A link to generate instructions for the agent.
 - Run Options:** A list of run configurations including reli, real, posneg, old, cli, evenodd, temp, swift, defu, whil, couwhi, fals, dow, conf, realwhi, realloop, loopnu..., loopeven, and loopsum.
 - Suggested Actions:** Buttons for "Build Workspace" and "Show Config".
 - Build Log:** A section describing what to build next, with a "Describe what to build next" input field and "Agent" dropdown.

- 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.

The screenshot shows a Java development environment with multiple files in the Explorer pane and a terminal window in the foreground. The terminal window displays the output of a Java application named 'loopcount'. The application prints the numbers 5, 4, 3, 2, and 1 sequentially. The terminal window also shows the command used to run the application: 'PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\VHP\AppData\Roaming\Code\User\workspaceStorage\d67226fb1c7f0eaec7971bc3fd885877\redhat.java\jdt_ws\OBJ Z lab task 4.62a910b5\bin' 'loopcount'

```

public class loopcount {
    public static void main(String[] args) {
        for (int i = 5; i > 0; i--) {
            System.out.println(i);
        }
    }
}

PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\VHP\AppData\Roaming\Code\User\workspaceStorage\d67226fb1c7f0eaec7971bc3fd885877\redhat.java\jdt_ws\OBJ Z lab task 4.62a910b5\bin' 'loopcount'
5
4
3
2
1

```

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

The screenshot shows a Java development environment with the following details:

- File Bar:** File, Edit, Selection, View, ...
- Toolbar:** Back, Forward, Home, Refresh, Chat.
- Explorer:** Shows a tree view of files under "OBJ Z LAB TASK 4".
- Open Editors:** Shows multiple Java files: dow.java, conf.java, realhi.java, realloop.java, loopnum.java, loopeven.java, loopsum.java, loopcount.java, and falseconloop.java (the active file).
- Code Editor:** Displays the code for "falseconloop.java":public class falseconloop {
 public static void main(String[] args) {
 for (int i = 10; i < 5; i++) {
 System.out.println("This will never be printed");
 }
 }
}
- Terminal:** Shows the command line output:

```
PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '--showCodeDetailsInExceptionMessages' '-cp' 'C:\Users\VHP\AppData\Roaming\Code\User\workspaceStorage\d67226fb1c7f0eac797bc3fd885877\redhat.java\jdt-ws\OBJ Z lab task 4_62a9105\bin' 'falseconloop'  
PS E:\OBJ Z lab task 4>
```
- Suggested Actions:** A dropdown menu with various run configurations.
- Build with Agent:** A feature panel for building the workspace.
- Bottom Status:** Ln 1, Col 18, Spaces: 4, UTF-8, CRLF, Java, Go Live, etc.

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

The screenshot shows a Microsoft Visual Studio Code (VS Code) interface. The left sidebar displays a file tree with several Java files under 'OPEN EDITORS' and 'OBJ Z LAB TASK 4'. The main area shows the code for 'nastloop.java':

```
1 < public class nastloop {
2   public static void main(String[] args) {
3     // Outer loop
4     for (int i = 1; i <= 2; i++) {
5       System.out.println("Outer: " + i); // Executes 2 times
6       // Inner loop
7       for (int j = 1; j <= 3; j++) {
8         System.out.println("Inner: " + j); // Executes 6 times (2 * 3)
9       }
10    }
11 }
```

The terminal at the bottom shows the output of running the code:

```
PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '.\objzshowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\HP\AppData\Roaming\CodeUser\workspaceStorage\d67226fb1c7f0eac971bc3fd885877\redhat.java\jdt-ws\OBJ Z lab task 4\62a9105\bin'\ 'nastloop'
Outer: 1
Inner: 1
Inner: 2
Inner: 3
Outer: 2
Inner: 1
Inner: 2
Inner: 3
```

The status bar at the bottom right indicates the date and time: 1/30/2026 5:19 PM.

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

The screenshot shows the Microsoft Visual Studio Code interface with the following details:

- File Bar:** File, Edit, Selection, View, ...
- Search Bar:** Q OBJ Z lab task 4
- Explorer:** Shows a list of Java files in the workspace, including `realwhi.java`, `realloop.java`, `loopnum.java`, `loopenv.java`, `loopsum.java`, `loopcount.java`, `falseconloop.java`, `nastloop.java`, and `multnasloop.java`.
- Open Editors:** Multinasloop.java is the active editor, displaying the following Java code:

```
1 public class multnasloop {  
2     public static void main(String[] args) {  
3         for (int i = 1; i <= 3; i++) {  
4             for (int j = 1; j <= 3; j++) {  
5                 System.out.print(i * j + " ");  
6             }  
7             System.out.println();  
8         }  
9     }  
10 }  
11
```

- Terminal:** Shows the command line output of running the Java code:

```
PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '--showCodeDetailsInExceptionMessages' '-cp' 'C:\Users\VP\AppData\Roaming\Code\User\workspaceStorage\d67226fb1c7f0eaecc971bc3fd885877\r\nhat\java\jdt_ws\OBJ Z lab task 4_62a9105\bin' 'multnasloop'  
1 2 3  
2 4 6  
3 6 9
```
- Suggested Actions:** A dropdown menu lists various run configurations for the Java file.
- Bottom Status Bar:** Shows the current line (Ln 1, Col 21), spaces (Spaces: 4), encoding (UTF-8), and date/time (5:20 PM, 1/30/2026).

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

The screenshot shows the Microsoft Visual Studio Code interface with the following details:

- File Bar:** File, Edit, Selection, View, ...
- Search Bar:** Q OBJ Z lab task 4
- Explorer:** Shows a tree view of files under "OBJ Z LAB TASK 4" and "OPEN EDITORS".
- Editor:** The main editor window displays the following Java code:

```
foreach.java > ...
1 public class foreach {
2     public static void main(String[] args) {
3         String[] cars = {"Volvo", "BMW", "Ford", "Mazda"};
4
5         for (String car : cars) {
6             System.out.println(car);
7         }
8     }
9 }
```
- Terminal:** Shows the command line output of running the Java code:

```
PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '>xc+showCodeDetailsInExceptionMessages' '-cp' 'C:/Users/VIP/AppData/Roaming/Code/User/workspaceStorage/d67226fb1c7f0eac7971bc3fd885877/reddhat.java/jdt_ws/OBJ Z lab task 4_62a9105\bin' 'foreach'
Volvo
BMW
Ford
Mazda
```
- Suggested Actions:** A dropdown menu lists various run configurations for the current file.
- Bottom Status Bar:** Shows the line number (Ln 10, Col 1), spaces (Spaces: 4), encoding (UTF-8), and date (1/30/2026).

- 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.

The screenshot shows the Visual Studio Code interface with the following details:

- File Bar:** File, Edit, Selection, View, ...
- Search Bar:** Q OBJ Z lab task 4
- Explorer:** Shows files in the workspace, including `OBJ Z LAB TASK 4` and `OPEN EDITORS` (which lists several Java files).
- Editor:** Displays the `valunumforeach.java` file with the following code:


```
public class valunumforeach {
    public static void main(String[] args) {
        int[] numbers = {10, 20, 30, 40};
        for (int num : numbers) {
            System.out.println(num);
        }
    }
}
```
- Terminal:** Shows the command line output:


```
PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '--showCodeDetailsInExceptionMessages' '-cp' 'C:\Users\V\P\AppData\Roaming\Code\User\workspaceStorage\d67226fb1c7f0eaec9791bc3fd885877\redhat.java\jdt_ws\OBJ Z lab task 4_62a910b5\bin'\ 'valunumforeach'
10
20
30
40
```
- Suggestions:** A dropdown menu shows suggestions for the current word, such as `Run: temp`, `Run: swit`, etc.
- Build with Agent:** A panel on the right provides options to build the workspace or a specific file.
- Bottom Status Bar:** Shows the current file (`valunumforeach.java`), line (Ln 1, Col 18), spaces (Spaces: 4), encoding (UTF-8), and date (1/30/2026).

- 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.

The screenshot shows the Visual Studio Code interface with the following details:

- File Bar:** File, Edit, Selection, View, ...
- Search Bar:** Q OBJ Z lab task 4
- Explorer:** Shows files in the workspace, including `OBJ Z LAB TASK 4` and `OPEN EDITORS` (which lists several Java files).
- Editor:** Displays the `countloop.java` file with the following code:


```
public class countloop {
    public static void main(String[] args) {
        for (int i = 0; i <= 100; i += 10) {
            System.out.println(i);
        }
    }
}
```
- Terminal:** Shows the command line output:


```
PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '--showCodeDetailsInExceptionMessages' '-cp' 'C:\Users\V\P\AppData\Roaming\Code\User\workspaceStorage\d67226fb1c7f0eaec9791bc3fd885877\redhat.java\jdt_ws\OBJ Z lab task 4_62a910b5\bin'\ 'countloop'
0
10
20
30
40
50
60
70
80
90
100
```
- Suggestions:** A dropdown menu shows suggestions for the current word, such as `Run: swit`, `Run: defu`, etc.
- Build with Agent:** A panel on the right provides options to build the workspace or a specific file.
- Bottom Status Bar:** Shows the current file (`countloop.java`), line (Ln 1, Col 18), spaces (Spaces: 4), encoding (UTF-8), and date (1/30/2026).

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

The screenshot shows the Microsoft Visual Studio Code interface. The title bar reads "Q. OBJ Z lab task 4". The left sidebar has sections for "OPEN EDITORS" and "OBJ Z LAB TASK 4". The "OPEN EDITORS" section lists several Java files: "loopsum.java", "loopcount.java", "falseconloop.java", "nastloop.java", "multloop.java", "foreach.java", "valumnumforeach.java", "countloop.java", and "zerototen.java". The "OBJ Z LAB TASK 4" section lists various Java classes and files. The main editor area displays the "zerototen.java" file:

```
public class zerototen {
    public static void main(String[] args) {
        for (int i = 0; i <= 10; i = i + 2) {
            System.out.println(i);
        }
    }
}
```

The terminal below the editor shows the command "PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '--showCodeDetailsInExceptionMessages' '-cp' 'C:\Users\VHP\AppData\Roaming\Code\User\workspaceStorage\d67226fb1c7f0eaecc971bc3fd885877\redhat_java\jdt_ws\OBJ Z lab task 4_62a910b5\bin'" 'zerototen'" followed by the output "0", "2", "4", "6", "8", "10". The status bar at the bottom right shows "Ln 1, Col 18 Spaces: 4", "UTF-8", "CRLF", "Java", "5:27 PM", "1/30/2026".

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

The screenshot shows the Visual Studio Code interface with the following details:

- File Bar:** File, Edit, Selection, View, ...
- Search Bar:** Q OBJ Z lab task 4
- Explorer:** Shows a list of Java files in the workspace, including `foreach.java`, `valunumforeach.java`, `countloop.java`, `zerototen.java`, `multiforloop.java`, `OBJ Z LAB TASK 4`, `matmax.java`, `mathmin.class`, `mathmin.java`, `mathpow.class`, `mathpow.java`, `mathrandom.java`, `mathrandom.class`, `mathsqrt.class`, `mathsqrt.java`, `multiforloop.java`, `multinasloop.java`, `nasif.java`, `naster.java`, `nastiloop.java`, `not.java`, `num.class`, `num.java`, `number.class`.
- Terminal:** Displays the command PS E:\OBJ Z lab task 4 & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '--showCodeDetailsInExceptionMessages' '-cp' 'C:\Users\VIP\AppData\Roaming\Code\User\workspaceStorage\d67226fb1c7f0eaec9791bc3fd885877\redhat.java\jdt_ws\OBJ Z lab task 4_62a91005\bin' 'multiforloop'. The output shows the multiplication table for 2:

```
2 x 1 = 2
2 x 2 = 4
2 x 3 = 6
2 x 4 = 8
2 x 5 = 10
2 x 6 = 12
2 x 7 = 14
2 x 8 = 16
2 x 9 = 18
2 x 10 = 20
PS E:\OBJ Z lab task 4
```

- Suggested Actions:** A dropdown menu with various run configurations like Run: while, Run: couwhi, etc.
- Build with Agent:** A section for generating agent instructions.
- Bottom Bar:** Shows the status bar with Ln 1, Col 18, Spaces: 4, UTF-8, CRLF, Java, Go Live, and a date/time stamp of 5:28 PM 1/30/2026.

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

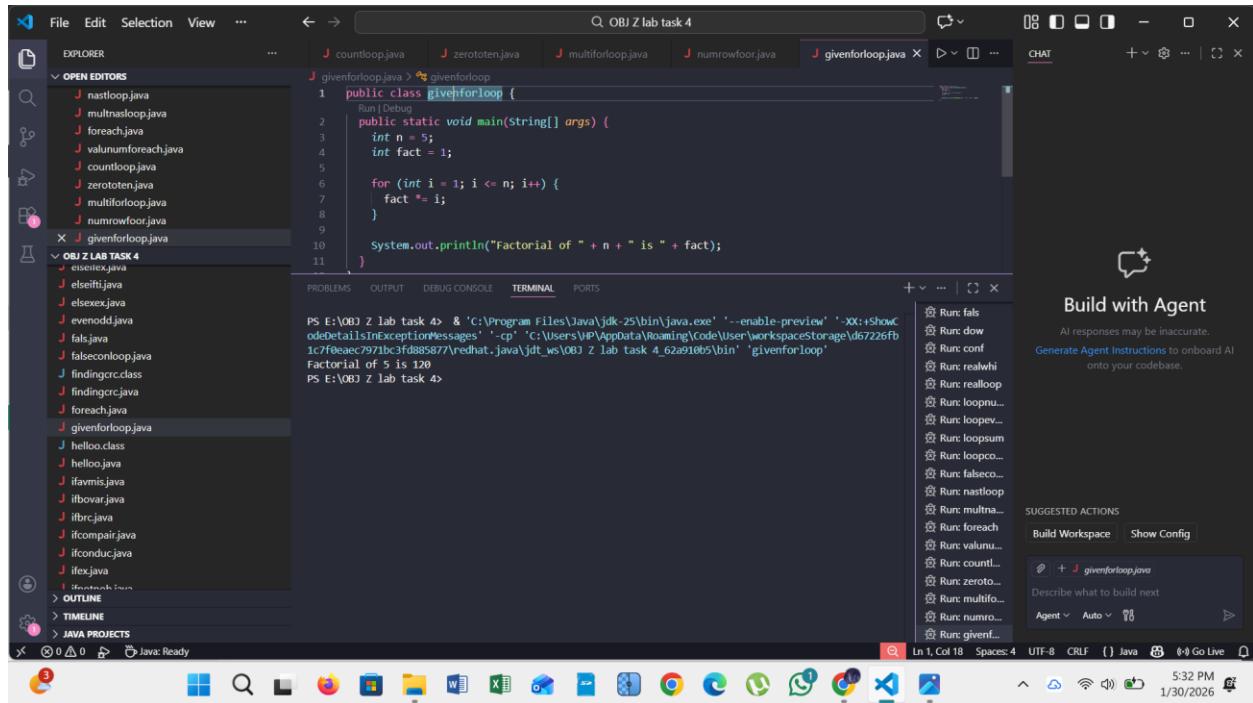
The screenshot shows the Visual Studio Code interface with the following details:

- File Bar:** File, Edit, Selection, View, ...
- Search Bar:** Q OBJ Z lab task 4
- Explorer:** Shows a list of Java files in the workspace, including `foreach.java`, `valunumforeach.java`, `countloop.java`, `zerototen.java`, `multiforloop.java`, `OBJ Z LAB TASK 4`, `naster.java`, `nastiloop.java`, `not.java`, `num.class`, `num.java`, `number.class`, `numc.java`, `numrowfor.java`, `old.java`, `or.java`, `posneg.java`, `random.class`, `random.java`, `realjava`, `realloop.java`, `realwhi.java`.
- Terminal:** Displays the command PS E:\OBJ Z lab task 4 & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '--showCodeDetailsInExceptionMessages' '-cp' 'C:\Users\VIP\AppData\Roaming\Code\User\workspaceStorage\d67226fb1c7f0eaec9791bc3fd885877\redhat.java\jdt_ws\OBJ Z lab task 4_62a91005\bin' 'numrowfor'. The output shows seat numbers from 1 to 5:

```
Seat number: 1
Seat number: 2
Seat number: 3
Seat number: 4
Seat number: 5
PS E:\OBJ Z lab task 4
```

- Suggested Actions:** A dropdown menu with various run configurations like Run: couwhi, etc.
- Build with Agent:** A section for generating agent instructions.
- Bottom Bar:** Shows the status bar with Ln 1, Col 16, Spaces: 4, UTF-8, CRLF, Java, Go Live, and a date/time stamp of 5:30 PM 1/30/2026.

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



The screenshot shows the Visual Studio Code (VS Code) interface with the following details:

- File Bar:** File, Edit, Selection, View, ...
- Title Bar:** OBJ Z lab task 4
- Explorer:** Shows a list of Java files in the workspace, including `nastloop.java`, `multisloop.java`, `foreach.java`, `valumunforeach.java`, `countloop.java`, `zerototen.java`, `multiforloop.java`, `numrowfoor.java`, and `givenforloop.java`.
- Editor:** The `givenforloop.java` file is open, displaying the following Java code:

```

1 public class givenforloop {
2     public static void main(String[] args) {
3         int n = 5;
4         int fact = 1;
5
6         for (int i = 1; i <= n; i++) {
7             fact *= i;
8         }
9
10        System.out.println("Factorial of " + n + " is " + fact);
11    }

```
- Terminal:** Shows the output of running the code, which prints "Factorial of 5 is 120".
- Suggested Actions:** A sidebar on the right lists various run configurations for the Java code.
- Bottom Status Bar:** Shows the current line (Ln 1, Col 18), spaces (Spaces: 4), encoding (UTF-8), and date/time (5:32 PM, 1/30/2026).

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

The screenshot shows the Microsoft Visual Studio Code interface with the following details:

- File Bar:** File, Edit, Selection, View, ...
- Search Bar:** Q OBJ Z lab task 4
- Explorer:** Shows a list of Java files in the workspace, including `multisloop.java`, `foreach.java`, `valumoreach.java`, `countloop.java`, `zerototen.java`, `multiforloop.java`, `numrowforloop.java`, `givenforloop.java`, and `brkcoun.java`.
- Terminal:** Displays the command-line output of running the `brkcoun.java` file. The output shows the numbers 0, 1, 2, and 3 printed to the console.
- Suggested Actions:** A dropdown menu lists various run configurations for the Java file, such as `Run: dow`, `Run: conf`, `Run: realwhi`, etc.
- Suggested Actions Bar:** Includes buttons for "Build Workspace" and "Show Config".
- Bottom Status Bar:** Shows Ln 1, Col 18, Spaces: 4, UTF-8, CRLF, Java, 5:35 PM, 1/30/2026.

```
public class brkcoun {  
    public static void main(String[] args) {  
        for (int i = 0; i < 10; i++) {  
            if (i == 4) {  
                break;  
            }  
            System.out.println(i);  
        }  
    }  
}
```

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

The screenshot shows the Microsoft Visual Studio Code interface with the following details:

- File Bar:** File, Edit, Selection, View, ...
- Search Bar:** Q OBJ Z lab task 4
- Explorer:** Shows a list of Java files in the workspace, including foreach.java, valumoreforeach.java, countloop.java, zerototen.java, multiforloop.java, numrowforloop.java, givenforloop.java, brkcoun.java, cont.java, concat.java, concatex.class, concatex.java, concatenation.class, concatenation.java, Conceternation.class, Conceternation.java, conf.java, cont.java, countloop.java, couwhi.java, defjava, dow.java, elseifex.java, elseif.java, elseex.java, evenodd.java.
- Terminal:** Shows the command PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '-Djava.awt.d&showCodeDetailsInExceptionMessages' '-cp' 'C:/Users/VIP/AppData/Roaming/Code/User/workspaceStorage/d67226fb1c7f0eaec7971bc3fd0885877/reddhat.java/jdt_ws/OBJ Z lab task 4_62a9105/bin' 'cont'. The output shows the numbers 0 through 9, with the number 4 omitted due to the continue statement in the code.
- Code Editor:** Displays the Java code for cont.java:

```
1 public class cont {
2     public static void main(String[] args) {
3         for (int i = 0; i < 10; i++) {
4             if (i == 4) {
5                 continue;
6             }
7             System.out.println(i);
8         }
9     }
10 }
```
- Suggestions:** A dropdown menu titled "Build with Agent" lists various run configurations: Run: conf, Run: realwhi..., Run: loopal..., Run: loopnu..., Run: loopov..., Run: loopsum..., Run: loopco..., Run: falseco..., Run: nastloop..., Run: multna..., Run: foreach..., Run: valum..., Run: countl..., Run: zero..., Run: multifor..., Run: numrow..., Run: givenfor..., Run: brkcoun..., Run: conti... .
- Bottom Bar:** Shows the status bar with Ln 1, Col 18, Spaces: 4, UTF-8, CRLF, Java, 5:36 PM, 1/30/2026.

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

The screenshot shows the Microsoft Visual Studio Code interface with the following details:

- File Bar:** File, Edit, Selection, View, ...
- Search Bar:** Q OBJ Z lab task 4
- Explorer:** Shows a tree view of files under "OBJ Z LAB TASK 4" and "OPEN EDITORS".
- Editor:** Displays the "cbc.java" file content:

```
1 public class cbc {  
2     public static void main(String[] args) {  
3         for (int i = 0; i < 6; i++) {  
4             if (i == 2) {  
5                 continue;  
6             }  
7             if (i == 4) {  
8                 break;  
9             }  
10            System.out.println(i);  
11        }  
12    }  
13}
```
- Terminal:** Shows the command PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '--showCodeDetailsInExceptionMessages' '-cp' 'C:/Users/VIP/AppData/Roaming/Code/User/workspaceStorage/d67226fb1c7f0eaecc971bc3fd885877/reddhat.java/jdt_ws/OBJ Z lab task 4_62a9105/bin' 'cbc'. The output is:

```
0  
1  
3
```
- Run Context Menu:** A context menu is open over the code editor, listing various run configurations.
- Suggested Actions:** A panel on the right lists "Build Workspace" and "Show Config".
- Bottom Status Bar:** Shows Ln 8, Col 15, Spaces: 4, UTF-8, CRLF, Java, 5:37 PM, 1/30/2026.

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

The screenshot shows the Microsoft Visual Studio Code interface with the following details:

- File Bar:** File, Edit, Selection, View, ...
- Title Bar:** Q OBJ Z lab task 4
- Explorer:** Shows files in the workspace, including `bcwl.java` which is currently selected.
- Terminal:** Displays the command-line output of running the `bcwl` Java class. The output shows the numbers 0 through 3 being printed, followed by the prompt `PS E:\OBJ Z lab task 4>`.
- Context Menu (opened over bcwl.java):** A list of suggested actions for running the file, such as `Run: reloop`, `Run: loopnu...`, etc.
- Suggested Actions:** A panel on the right showing options like `Build Workspace` and `Show Config`.
- Bottom Status Bar:** Shows Ln 3, Col 15, Spaces: 4, UTF-8, CRLF, Java, 5:38 PM, 1/30/2026.

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

The screenshot shows a Java development environment with the following details:

- File Bar:** File, Edit, Selection, View, ...
- Toolbar:** Standard icons for file operations.
- EXPLORER:** Shows a tree view of files under "OBJ Z LAB TASK 4".
- OPEN EDITORS:** Shows multiple Java files: givenforloop.java, brkcoun.java, contijava, cbcjava, bcw.java, and cex.java.
- Code Editor:** The "cex.java" file is open, containing the following code:

```
public class cex {
    public static void main(String[] args) {
        int i = 0;
        while (i < 10) {
            if (i == 4) {
                i++;
                continue;
            }
            System.out.println(i);
            i++;
        }
    }
}
```
- TERMINAL:** The terminal window shows the command PS E:\OBJ Z lab task 4 & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' 'objz+showCodeDetailsInExceptionMessages' '-cp' 'C:\Users\VP\AppData\Roaming\Code\User\workspaceStorage\d67226fb1c7f0eaecc971bc3fd0885877\redhat.java\jdt_ws\OBJ Z lab task 4_62a9105\bin' 'cex'
- SUGGESTED ACTIONS:** A dropdown menu with various run configurations like Run: loopnu..., Run: loopev..., Run: loopsum..., Run: loopco..., Run: falseco..., Run: nastloop..., Run: multima..., Run: foreach..., Run: valnumu..., Run: countl..., Run: zerofo..., Run: multifo..., Run: numro..., Run: givenf..., Run: brkcoun..., Run: conti..., Run: cbc..., Run: bcw..., and Run: cex.
- STATUS BAR:** Shows Ln 1, Col 18, Spaces: 4, UTF-8, CRLF, Java, 5:40 PM, 1/30/2026.

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

The screenshot shows the Microsoft Visual Studio Code interface with the following details:

- File Bar:** File, Edit, Selection, View, ...
- Search Bar:** Q OBJ Z lab task 4
- Explorer:** Shows a list of Java files in the workspace, including `multiforloop.java`, `numrowforloop.java`, `givenforloop.java`, `brkcoun.java`, `contijava`, `cbcjava`, `bcwjava`, `cexjava`, `rle.java`, `or.java`, `posneg.java`, `random.class`, `random.java`, `realjava`, `realoop.java`, `realwhi.java`, `relijava`, `relas.java`, `rle.java`, `rmrwwhite.class`, `rmrwwhite.java`, `rounding.class`, `rounding.java`, `shifeise.java`, `stien.class`, `stien.java`, `swt.java`.
- Terminal:** Displays the command-line output of running the `rle.java` file. The output shows the Java command being run and the resulting sequence of numbers: 3, 7.
- Context Menu:** A context menu is open over the `rle.java` file in the Explorer. It includes options like Run: loop..., Run: loopsum..., Run: loopco..., Run: falseco..., Run: nastloop..., Run: multna..., Run: foreach..., Run: valumu..., Run: countl..., Run: zero...o..., Run: multifo..., Run: numro..., Run: givenf..., Run: brkcoun..., Run: conti..., Run: cbc..., Run: bcw..., Run: cex..., and Run: rle... .
- Suggested Actions:** A sidebar titled "Build with Agent" lists suggested actions for building the workspace, such as "Build Workspace" and "Show Config".
- Bottom Status Bar:** Shows the current line (Ln 1), column (Col 19), spaces (Spaces: 4), encoding (UTF-8), and date/time (5:41 PM, 1/30/2026).

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

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows files like numrowfor.java, givenforloop.java, brkcoun.java, conti.java, cbc.java, bndl.java, rle.java, and arrex.java.
- Code Editor:** Displays the content of `arrex.java`:

```
public class arrex { public static void main(String[] args) { String[] cars = {"volvo", "BMW", "Ford", "Mazda"}; System.out.println(cars[0]); } }
```
- Terminal:** Shows the command-line output of running the Java code:

```
PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '--XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\V\P\AppData\Roaming\Code\User\workspaceStorage\d67226fb1cf0eaec7971bc3fd885877\redhat.java\jdt_ws\OBJ Z lab task 4_62a91005\bin' 'arrex'  
Volvo
```
- Suggested Actions:** A dropdown menu lists various run configurations for the current file, such as Run: loopsum, Run: loopco..., Run: falseco..., Run: nastloop, Run: multna..., Run: foreach, Run: foreach, Run: valnum..., Run: countl..., Run: zeroit..., Run: multifo..., Run: numro..., Run: givenf..., Run: brkcoun, Run: conti, Run: cbc, Run: bndl, Run: rle, and Run: arrex.
- Build with Agent:** A panel on the right provides AI integration options.
- Bottom Status Bar:** Shows the line number (Ln 1, Col 18), spaces (Spaces: 4), encoding (UTF-8 CRLF), Java, Go Live, and the date/time (5:45 PM 1/30/2026).

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

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows files like givenforloop.java, brkcoun.java, conti.java, cbc.java, bndl.java, rle.java, arrex.java, and cngar.java.
- Code Editor:** Displays the content of `cngar.java`:

```
public class cngar { public static void main(String[] args) { String[] cars = {"volvo", "BMW", "Ford", "Mazda"}; cars[0] = "opel"; System.out.println(cars[0]); } }
```
- Terminal:** Shows the command-line output of running the Java code:

```
PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '--XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\V\P\AppData\Roaming\Code\User\workspaceStorage\d67226fb1cf0eaec7971bc3fd885877\redhat.java\jdt_ws\OBJ Z lab task 4_62a91005\bin' 'cngar'  
Opel
```
- Suggested Actions:** A dropdown menu lists various run configurations for the current file, such as Run: loopco..., Run: falseco..., Run: nastloop, Run: multna..., Run: foreach, Run: foreach, Run: valnum..., Run: countl..., Run: zeroit..., Run: multifo..., Run: numro..., Run: givenf..., Run: brkcoun, Run: conti, Run: cbc, Run: bndl, Run: rle, and Run: arrex.
- Build with Agent:** A panel on the right provides AI integration options.
- Bottom Status Bar:** Shows the line number (Ln 1, Col 18), spaces (Spaces: 4), encoding (UTF-8 CRLF), Java, Go Live, and the date/time (5:47 PM 1/30/2026).

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

The screenshot shows the Visual Studio Code (VS Code) interface with the following details:

- File Structure (EXPLORER):** Shows various Java files like arr.java, arrcoun.java, conti.java, etc., under the 'OBJ Z LAB TASK 4' project.
- Code Editor:** Displays the content of arr.java:


```
arr.java > arr
1 public class arr {
2     Run | Debug
3     public static void main(String[] args) {
4         String[] cars = {"volvo", "BMW", "Ford", "Mazda"};
5         System.out.println(cars.length);
6     }
7 }
```
- Terminal:** Shows the command run in the terminal:


```
PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\VHP\AppData\Roaming\Code\User\workspaceStorage\d67226fb1cf0eac797bc3fd885877\redhat.java\jdt_ws\OBJ Z lab task 4.62a910b5\bin' 'arr'
4
```
- Context Menu (bottom right):** A context menu is open for the word 'arr' in the code editor, listing various run configurations such as 'Run: falseco...', 'Run: nastloop...', 'Run: multifa...', 'Run: foreach...', 'Run: volumu...', 'Run: count...', 'Run: zero...', 'Run: multifo...', 'Run: numro...', 'Run: given...', 'Run: brkoun...', 'Run: conti...', 'Run: cbc...', 'Run: bndl...', 'Run: cex...', 'Run: fe...', 'Run: arr...', 'Run: engar...', and 'Run: arr!'. It also includes 'SUGGESTED ACTIONS' for 'Build Workspace' and 'Show Config'.
- Bottom Status Bar:** Shows the current file is arr.java, line 1, column 18, spaces: 4, encoding: UTF-8, CRLF, Java, Go Live, and the date/time: 5:49 PM, 1/30/2026.

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

The screenshot shows an IDE interface with the following details:

- File Menu:** File, Edit, Selection, View, ...
- Toolbar:** Back, Forward, Search, Chat.
- Sidebar:** EXPLORER (OPEN EDITORS: conti.java, cbcjava, bocl.java, cex.java, rle.java, arrex.java, cngar.java, arl.java, nkar.java), OUTLINE, TIMELINE, JAVA PROJECTS.
- Editor:** nkar.java (containing a main method that prints "Volvo").
- Terminal:** PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '-enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' 'cp' 'C:\Users\VP\AppData\Roaming\Code\User\workspaceStorage\d67226fb1c2f6eac5971bc3fd885677\redhat\java\jdt_ws\OBJ Z lab task 4_62a91805\bin'\nkar
Volvo
PS E:\OBJ Z lab task 4>
- Suggested Actions:** Run: nastloop, Run: multna..., Run: foreach, Run: valun..., Run: count..., Run: zerofo..., Run: multifo..., Run: numro..., Run: givent..., Run: brkcoun, Run: conti, Run: cbc, Run: bocl, Run: cex, Run: rle, Run: arrex, Run: cngar, Run: arl, Run: nkar.
- Status Bar:** Ln 1, Col 18, Spaces: 4, UTF-8, CRLF, Java, 5:51 PM, 1/30/2026.

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

The screenshot shows the Visual Studio Code interface with the following details:

- File Bar:** File, Edit, Selection, View, ...
- Search Bar:** Q OBJ Z lab task 4
- Explorer:** Shows files like `java`, `rle.java`, `arrex.java`, `cngar.java`, `arr.java`, `nkar.java`, `Ita.java`, and `OBJ Z LAB TASK 4` containing files such as `itex.java`, `ifpotpob.java`, `jvericom.java`, `jtsalright.java`, `jtsalright.class`, `loopcount.java`, `loopeven.java`, `loopnum.java`, `loopsum.java`, `J.java`, `mathabs.class`, `mathabs.java`, `mathmax.class`, `mathmax.java`, `mathmin.class`, `mathmin.java`, `mathpow.class`, `mathpow.java`.
- Terminal:** Displays the output of running the `Ita.java` file. The terminal shows the Java command being run and the resulting output: "Volvo", "BMW", "Ford", "Mazda".
- Suggested Actions:** A dropdown menu with options like Run: multithreaded, Run: foreach, Run: valuenumerate, Run: count..., Run: zero..., Run: multithreaded, Run: numro..., Run: given..., Run: brkcoun, Run: Run: conti, Run: cbc, Run: bocl, Run: cex, Run: rle, Run: arrex, Run: cngar, Run: arfl, Run: nkar, Run: Ita.
- Build with Agent:** A panel on the right with the message "Build with Agent" and "Al responses may be inaccurate. Generate Agent Instructions to onboard AI onto your codebase."
- Status Bar:** Ln 8, Col 2, Spaces: 4, UTF-8, CRLF, Java, Go Live, 5:54 PM, 1/30/2026

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

The screenshot shows the Visual Studio Code interface with the following details:

- File Bar:** File, Edit, Selection, View, ...
- Search Bar:** Q OBJ Z lab task 4
- Explorer:** Shows files like `java`, `arrex.java`, `cngar.java`, `arr.java`, `nkar.java`, `Ita.java`, and `OBJ Z LAB TASK 4` containing files such as `arex.java`, `and.java`, `arrex.java`, `arr.java`, `backslash.class`, `backslash.java`, `bcwl.java`, `bole.class`, `bole.java`, `bool.java`, `bool.class`, `booleanex.java`, `booleanex.class`, `boolex.java`, `boole.class`, `boole.java`.
- Terminal:** Displays the output of running the `arex.java` file. The terminal shows the Java command being run and the resulting output: "10", "20", "30", "40".
- Suggested Actions:** A dropdown menu with options like Run: foreach, Run: valuenumerate, Run: count..., Run: zero..., Run: multithreaded, Run: numro..., Run: given..., Run: brkcoun, Run: Run: conti, Run: cbc, Run: bocl, Run: cex, Run: rle, Run: arrex, Run: cngar, Run: arfl, Run: nkar, Run: Ita.
- Build with Agent:** A panel on the right with the message "Build with Agent" and "Al responses may be inaccurate. Generate Agent Instructions to onboard AI onto your codebase."
- Status Bar:** Ln 1, Col 18, Spaces: 4, UTF-8, CRLF, Java, Go Live, 5:56 PM, 1/30/2026

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

```

public class sl {
    public static void main(String[] args) {
        int[] numbers = {1, 5, 10, 25};
        int sum = 0;

        // Loop through the array and add each element to sum
        for (int i = 0; i < numbers.length; i++) {
            sum += numbers[i];
        }

        System.out.println("The sum is: " + sum);
    }
}

```

The sum is: 41

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

The screenshot shows the Microsoft Visual Studio Code interface. In the center, a terminal window displays Java code and its execution output. The code is a simple for-each loop that prints each item in an array of car brands. The terminal output shows the words 'Volvo', 'BMW', 'Ford', and 'Mazda' printed to the console. The status bar at the bottom right indicates the date and time as 1/30/2026 5:59 PM.

```
public class foreac {
    public static void main(String[] args) {
        String[] cars = {"Volvo", "BMW", "Ford", "Mazda"};
        for (String car : cars) {
            System.out.println(car);
        }
    }
}
```

```
PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '--showCodeDetailsInExceptionMessages' '-cp' 'C:/Users/VIP/AppData/Roaming/Code/User/workspaceStorage/d67226fb1c7f0eaecc971bc3fd885877/reddhat.java/jdt_ws/OBJ Z lab task 4_62a9105/bin' 'foreac'
Volvo
BMW
Ford
Mazda
```

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

The screenshot shows the Microsoft Visual Studio Code interface with the following details:

- File Menu:** File, Edit, Selection, View, ...
- Editor Bar:** File, Edit, Selection, View, ...
- Search Bar:** Q OBJ Z lab task 4
- Terminal:** elar1.java > elar1 > main(String[])
- Code Editor:** elar1.java (Content shown below)
- Output Panel:** PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '--showCodeDetailsInExceptionMessages' '-cp' 'C:/Users/VIP/AppData/Roaming/Code/User/workspaceStorage/d67226fb1c7f0aecc971bc3fd885877/redhat.java/jdt_ws/OBJ Z lab task 4.62a9105/bin' 'elar1'
The average age is: 46.75
PS E:\OBJ Z lab task 4>
- Suggested Actions:** Run: zero... (and many others listed)
- Bottom Status Bar:** Ln 7, Col 1 Spaces: 4 UTF-8 CRLF (Java) Go Live 6:01 PM 1/30/2026

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

The screenshot shows the Microsoft Visual Studio Code interface with the following details:

- File Bar:** File, Edit, Selection, View, ...
- Editor:** The main editor window displays a Java file named `rllif2.java` with the following code:

```
public class rllif2 {
    public static void main(String[] args) {
        int lowestAge = args[0];
        for (int i = 1; i < args.length; i++) {
            if (lowestAge > args[i]) {
                lowestAge = args[i];
            }
        }
        System.out.println("The lowest age in the array is: " + lowestAge);
    }
}
```
- Terminal:** The terminal window shows the output of running the Java code:

```
PS E:\OBJ Z lab task 4> java rllif2
The lowest age in the array is: 18
PS E:\OBJ Z lab task 4>
```
- Suggestions:** A dropdown menu titled "Build with Agent" lists various run configurations.
- Status Bar:** Shows the current file is `rllif2.java`, the line count is 17, the character count is 409, the encoding is UTF-8, and the date is 1/30/2026.

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

The screenshot shows the Visual Studio Code interface with the following details:

- File Bar:** File, Edit, Selection, View, ...
- Title Bar:** Q OBJ Z lab task 4
- Explorer:** Shows a list of Java files in the workspace, including arrex.java, sl.java, foreac.java, elar1.java, rllif2.java, gokgok.java, and several files under OBJ Z LAB TASK 4.
- Code Editor:** Displays the following Java code:

```
public class gokgok {    public static void main(String[] args) {        int[][] myNumbers = { { 1, 4, 2 }, { 3, 6, 8 } };        System.out.println(myNumbers[1][2]);    }}
```
- Terminal:** Shows the command PS E:\OBJ Z lab task 4 & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\V\P\AppData\Roaming\Code\User\workspaceStorage\d67226fb1c7f0eaec797bc3fd885877\redhat.java\jdt_ws\OBJ Z lab task 4.62a910b5\bin' 'gokgok' followed by the output 8.
- Suggested Actions:** A sidebar on the right lists various run configurations such as Run: numro..., Run: givenf..., Run: brkcoun, etc.
- Bottom Status Bar:** Ln 1, Col 18, Spaces: 4, UTF-8, CRLF, Java, Go Live, 6:06 PM, 1/30/2026.

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

This screenshot is nearly identical to the first one, showing the same Java code and terminal output. The difference is in the code editor's status bar, which now shows Ln 8, Col 1, indicating a cursor has moved to the eighth line of the code.

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

```

J rac.java > raSc
1 public class rac {
2     public static void main(String[] args) {
3         int[][] myNumbers = { {1, 4, 2}, {3, 5, 8, 5, 2} };
4
5         System.out.println("Rows: " + myNumbers.length); // 2
6         System.out.println("Cols in row 0: " + myNumbers[0].length); // 3
7         System.out.println("Cols in row 1: " + myNumbers[1].length); // 5
8     }
9 }
10
PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '--showCodeDetailsInExceptionMessages' '-cp' 'C:\Users\VP\AppData\Roaming\CodeUser\workspaceStorage\d67226f1cf0eac97bcb3fd885877\redhat.java\dt_ws\OBJ Z lab task 4.62a9105\bin' 'rac'
ROWS: 2
Cols in row 0: 3
Cols in row 1: 5
PS E:\OBJ Z lab task 4>
  
```

Build with Agent

AI responses may be inaccurate.
Generate Agent Instructions to onboard AI onto your codebase.

SUGGESTED ACTIONS

Build Workspace Show Config

Agent Auto

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

```
public class ltm {
    public static void main(String[] args) {
        int[][] myNumbers = { {1, 4, 2}, {3, 6, 8, 5, 2} };

        for (int row = 0; row < myNumbers.length; row++) {
            for (int col = 0; col < myNumbers[row].length; col++) {
                System.out.println("myNumbers[" + row + "][" + col + "] = " + myNumbers[row][col]);
            }
        }
    }
}
```

PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' 'ltm'
myNumbers[0][0] = 1
myNumbers[0][1] = 4
myNumbers[0][2] = 2
myNumbers[1][0] = 3
myNumbers[1][1] = 6
myNumbers[1][2] = 8
myNumbers[1][3] = 5
myNumbers[1][4] = 2

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

```
public class wifu {
    public static void main(String[] args) {
        int[][] myNumbers = { {1, 4, 2}, {3, 6, 8, 5, 2} };

        for (int[] row : myNumbers) {
            for (int num : row) {
                System.out.println(num);
            }
        }
    }
}
```

PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' 'wifu'
1
4
2
3
6
8
5
2

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

The screenshot shows a Java code editor interface with the following details:

- File Explorer:** Shows a tree view of files under "OBJ Z LAB TASK 4". The "backslash.java" file is selected.
- Editor:** Displays the code for "backslash.java":

```
public class backslash {
    public static void main(String[] args) {
        String txt = "The character \\ is called backslash.";
        System.out.println(txt);
    }
}
```

- Terminal:** Shows the command line output of running the program:

```
PS E:\OBJ Z lab task 4> & 'C:\Program Files\Java\jdk-25\bin\java.exe' '--enable-preview' '--showCodeDetailsInExceptionMessages' '-cp' 'C:\Users\V\P\AppData\Roaming\Code\User\workspaceStorage\d67226fb1c7f0eaec7971bc3fd885877\redhat\java\jdt_ws\OBJ Z lab task 4\62a910b5\bin' 'backslash'
The character \ is called backslash.
```

- Suggested Actions:** A sidebar on the right suggests building the workspace or configuration.

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