


Name: Ganesh N Moroliya

Level 2 output:

Level 2



Task 1



Task: Tic-Tac-Toe Game

Description: Implement a two-player tic-tac-toe game. Display the game board and prompt each player to enter their moves. Check for a winning condition or a draw after each move, and display the result accordingly. Allow the players to play multiple rounds if desired.

Skills: Arrays or matrices, loops, conditional statements.



Program:

```
import java.util.*;
public class Task2 {
    private static char[][] board = new char[3][3];
    private static char currentPlayer = 'X';
    private static boolean gameOn = true;

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        String playAgain;
        do {
            resetBoard();
            gameOn = true;
            currentPlayer = 'X';
            while (gameOn) {
                printBoard();
                playerMove(scanner);
                checkGameState();
                currentPlayer = (currentPlayer == 'X') ? 'O' : 'X';
            }
            System.out.print("Do you want to play again? (yes/no): ");
            playAgain = scanner.nextLine().trim().toLowerCase();
        }
```

```

        } while (playAgain.equals("yes"));
        scanner.close();
    }

    private static void resetBoard() {
        for (int i = 0; i < 3; i++) {
            for (int j = 0; j < 3; j++) {
                board[i][j] = '-';
            }
        }
    }

    private static void printBoard() {
        System.out.println("Current board:");
        for (int i = 0; i < 3; i++) {
            for (int j = 0; j < 3; j++) {
                System.out.print(board[i][j] + " ");
            }
            System.out.println();
        }
    }

    private static void playerMove(Scanner scanner) {
        int row, col;
        while (true) {
            System.out.println("Player " + currentPlayer + ", enter
your move (row and column: 0, 1, or 2): ");
            row = scanner.nextInt();
            col = scanner.nextInt();
            scanner.nextLine(); // Consume newline
            if (row >= 0 && row < 3 && col >= 0 && col < 3 &&
board[row][col] == '-') {
                board[row][col] = currentPlayer;
                break;
            } else {
                System.out.println("This move is not valid");
            }
        }
    }

    private static void checkGameState() {
        if (checkWinner()) {
            printBoard();
            System.out.println("Player " + currentPlayer + " wins!");
            gameOn = false;
        } else if (checkDraw()) {
            printBoard();
            System.out.println("The game is a draw!");
            gameOn = false;
        }
    }

    private static boolean checkWinner() {
        // Check rows and columns
        for (int i = 0; i < 3; i++) {
            if (board[i][0] == currentPlayer && board[i][1] ==
currentPlayer && board[i][2] == currentPlayer) {
                return true;
            }
            if (board[0][i] == currentPlayer && board[1][i] ==
currentPlayer && board[2][i] == currentPlayer) {

```

```

        return true;
    }
}
// Check diagonals
if (board[0][0] == currentPlayer && board[1][1] ==
currentPlayer && board[2][2] == currentPlayer) {
    return true;
}
if (board[0][2] == currentPlayer && board[1][1] ==
currentPlayer && board[2][0] == currentPlayer) {
    return true;
}
return false;
}

private static boolean checkDraw() {
    for (int i = 0; i < 3; i++) {
        for (int j = 0; j < 3; j++) {
            if (board[i][j] == '-') {
                return false;
            }
        }
    }
    return true;
}
}

```

output

The screenshot displays the IntelliJ IDEA interface. The main editor shows the source code for `Task2.java`, which implements a Tic-Tac-Toe game. The code includes a `main` method that handles user input, board updates, and game state checks. The `Run` window on the right shows the execution output, which includes the current board state, prompts for player moves, and the game's progress.

```

import java.util.*;

public class Task2 {

    private static char[][] board = new char[3][3];

    private static char currentPlayer = 'X';

    private static boolean gameOn = true;

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        String playAgain;
        do {
            resetBoard();
            gameOn = true;
            currentPlayer = 'X';
            while (gameOn) {
                printBoard();
                playerMove(scanner);
                checkGameState();
                currentPlayer = (currentPlayer == 'X') ? 'O' : 'X';
            }
            System.out.print("Do you want to play again? (yes/no): ");
            playAgain = scanner.nextLine().trim().toLowerCase();
        } while (playAgain.equals("yes"));
        scanner.close();
    }

    private static void resetBoard() {
        for (int i = 0; i < 3; i++) {
            for (int j = 0; j < 3; j++) {

```

Run Window Output:

```

Current board:
- - -
- - -
- - -
Player X, enter your move (row and column: 0, 1, or 2):
0
Current board:
- - -
- X -
- - -
Player O, enter your move (row and column: 0, 1, or 2):
0
Current board:
X O -
- X -
- - -
Player O, enter your move (row and column: 0, 1, or 2):
0
Current board:
X O O
- X -
- - -
Player X, enter your move (row and column: 0, 1, or 2):
2
Current board:
X O O
- X -
- X -
Player X wins!
Do you want to play again? (yes/no):

```

Level 2

Task 2



Task: Password Strength Checker

Description: Create a program that checks the strength of a password. Prompt the user to input a password and analyze its strength based on certain criteria, such as length, presence of uppercase letters, lowercase letters, numbers, and special characters. Provide feedback on the password strength.

Skills: String manipulation, conditional statements.



The screenshot shows an IDE with the following components:

- Project Explorer:** Shows a project structure with folders like 'Level2', 'Level3', and 'Level4'. The 'Level2' folder is expanded, showing files like 'Task2', 'Task3', and 'Task4'. 'Task3' is selected.
- Editor:** Displays the code for 'Task3.java'. The code is as follows:

```
1 package Level2;
2 import java.util.*;
3 public class Task3 {
4
5     public static void main(String[] args) {
6         Scanner scanner = new Scanner(System.in);
7
8         System.out.print("Enter your password: ");
9         String password = scanner.nextLine();
10
11         String strength = checkPasswordStrength(password);
12         System.out.println("Password strength: " + strength);
13     }
14
15     1 usage
16     public static String checkPasswordStrength(String password) {
17         int length = password.length();
18         boolean hasUpper = false;
19         boolean hasLower = false;
20         boolean hasDigit = false;
21         boolean hasSpecial = false;
22
23         for (char c : password.toCharArray()) {
24             if (Character.isUpperCase(c)) {
25                 hasUpper = true;
26             } else if (Character.isLowerCase(c)) {
27                 hasLower = true;
28             } else if (Character.isDigit(c)) {
29                 hasDigit = true;
30             } else if (isSpecialCharacter(c)) {
31                 hasSpecial = true;
```
- Run Console:** Shows the output of the program. It displays the command prompt, the input 'Pass@123#', and the output 'Password strength: Strong'. It also shows 'Process finished with exit code 0'.

The screenshot shows an IDE with a project named 'Level2'. The file explorer on the left shows a directory structure including 'Level2' and 'LevelTask3'. The main editor displays the code for 'Level2\Task3.java'. The code implements a password strength checker. It defines a method 'isSpecialCharacter' and a main method that prompts the user for a password and returns its strength based on length and character composition.

```

29         hasDigit = true;
30     } else if (isSpecialCharacter(c)) {
31         hasSpecial = true;
32     }
33 }
34
35 if (length >= 12 && hasUpper && hasLower && hasDigit && hasSpecial) {
36     return "Very Strong";
37 } else if (length >= 8 && hasUpper && hasLower && hasDigit) {
38     return "Strong";
39 } else if (length >= 8 && (hasUpper || hasLower) && hasDigit) {
40     return "Medium";
41 } else if (length >= 6) {
42     return "Weak";
43 } else {
44     return "Very Weak";
45 }
46
47
48 1 usage
49 public static boolean isSpecialCharacter(char c) {
50     String specialCharacters = "!@#$%^&*()-+";
51     return specialCharacters.indexOf(c) != -1;
52 }
53
54

```

The terminal window shows the execution of the program. It prompts the user to enter a password, and the output indicates that the password 'Pass@123#' is 'Strong'.

```

Run Task3 x
"C:\Program Files\Java\jdk-18.0.2.1\bin\java.exe" "-javaagent:C:\Progr
Enter your password: Pass@123#
Password strength: Strong
Process finished with exit code 0

```

Level 2

Task 3



Task: File Encryption/Decryption

Description: Create a program that encrypts or decrypts the contents of a text file using a simple encryption algorithm. Prompt the user to choose between encryption or decryption, and input the file name or path. Encrypt or decrypt the file accordingly and save the result to a new file.

Skills: File handling, string manipulation, basic input/output operations.



Program:

```
import java.io.*;
import java.util.Scanner;

public class Task4{

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.println("Do you want to (E)ncrypt or (D)ecrypt a
file?");
        String choice = scanner.nextLine().trim().toUpperCase();

        if (!choice.equals("E") && !choice.equals("D")) {
            System.out.println("Invalid choice. Please enter E for
Encryption or D for Decryption.");
            return;
        }

        System.out.println("Enter the file path: ");
        String filePath = scanner.nextLine().trim();

        System.out.println("Enter the shift key (integer): ");
        int shiftKey = scanner.nextInt();

        if (choice.equals("D")) {
            shiftKey = -shiftKey;
        }

        try {
            String content = readFile(filePath);
            String processedContent = processContent(content, shiftKey);
            String outputFilePath = choice.equals("E") ? "encrypted.txt" :
"decrypted.txt";
            writeFile(outputFilePath, processedContent);
            System.out.println("The file has been processed. Output file: "
+ outputFilePath);
        } catch (IOException e) {
            System.out.println("Error processing the file: " +
e.getMessage());
        }
    }

    private static String readFile(String filePath) throws IOException {
        StringBuilder content = new StringBuilder();
        try (BufferedReader br = new BufferedReader(new
FileReader(filePath))) {
            String line;
            while ((line = br.readLine()) != null) {
                content.append(line).append(System.lineSeparator());
            }
        }
        return content.toString();
    }
}
```

```

    private static void writeFile(String filePath, String content) throws
IOException {
        try (BufferedWriter bw = new BufferedWriter(new
FileWriter(filePath))) {
            bw.write(content);
        }
    }

    private static String processContent(String content, int shiftKey) {
        StringBuilder result = new StringBuilder();
        for (char c : content.toCharArray()) {
            if (Character.isLetter(c)) {
                char base = Character.isUpperCase(c) ? 'A' : 'a';
                char shifted = (char) (((c - base + shiftKey) % 26 + 26) %
26 + base);
                result.append(shifted);
            } else {
                result.append(c);
            }
        }
        return result.toString();
    }
}

```

Output:

The screenshot shows an IDE with a Java file named `Task4.java` and a Run window displaying the program's execution.

Code in `Task4.java`:

```

1  package Level2;
2  import java.io.*;
3  import java.util.Scanner;
4
5  public class Task4{
6
7      public static void main(String[] args) {
8          Scanner scanner = new Scanner(System.in);
9
10         System.out.println("Do you want to (E)ncrypt or (D)ecrypt a file?");
11         String choice = scanner.nextLine().trim().toUpperCase();
12
13         if (!choice.equals("E") && !choice.equals("D")) {
14             System.out.println("Invalid choice. Please enter E for Encryption or D for Decryption.");
15             return;
16         }
17
18         System.out.println("Enter the file path: ");
19         String filePath = scanner.nextLine().trim();
20
21         System.out.println("Enter the shift key (integer): ");
22         int shiftKey = scanner.nextInt();
23
24         if (choice.equals("D")) {
25             shiftKey = -shiftKey;
26         }
27
28         try {
29             String content = readFile(filePath);
30             String processedContent = processContent(content, shiftKey);
31             String outputFilePath = choice.equals("E") ? "encrypted.txt" : "decrypted.txt";
32             writeFile(outputFilePath, processedContent);

```

Run Window Output:

```

C:\Program Files\Java\jdk-18.0.2.1\bin\java.exe "-javaagent:C:\Program File
Do you want to (E)ncrypt or (D)ecrypt a file?
E
Enter the file path:
GANESHMOROLIYA.txt
Enter the shift key (integer):
5
Error processing the file: GANESHMOROLIYA.txt (The system cannot find the fi
Process finished with exit code 0

```