```java
import java.util.ArrayList;
import java.util.List;
import java.util.Random;
import java.util.Scanner;


/**
 * Pre-Conditions
 * Returns a list of "length' which is random symbols and numbers.
 * "Sequences" is a non-empty list of symbols/numbers.
 * "miliseconds" is a non-negative integer.
 * Length matches the length of the sequence displayed.
 * "usersequence" and "sequence" are list of the same length.
 */


//display time is the 20 seconds that the user is given to memories the
symbols/numbers
//sequence length is the amount of symbols/numbers that will be
displayed
public class Memory { //Defines Memory class

    private static final int DISPLAY_TIME_MS = 20000; //Declares a
constant 'DISPLAY_TIME_MS' with value of 2000, (equal to 20 seconds).
    private static final int SEQUENCE_LENGTH = 5; //Declares another
constant 'SEQUENCE_LENGTH' with length of the sequence = 5.

    //Generating the random sequences
    //Showing the sequence to the user
    //Waiting for the display time (use 20 second timer)
    //Clearing the sequence
    //prompting the user to enter the sequence once its gone
    //Check the users input
    public static void minigameplayer() { //Defines method
minigameplayer.
        System.out.println("entered minigameplayer"); //Prints "entered
minigameplayer' to console.
        Scanner170 console = new Scanner170(System.in); //Creates
scanner object 'console' to read user input.
        List<String> sequence =
generateRandomSequence(SEQUENCE_LENGTH); //Generates a random sequence
with the length of 5.

        System.out.println("Memorize the following sequence: ");
//Prints user game instructions to console.
        displaySequence(sequence); //Call to display the generated
sequence to the user in the console.

        waitForSeconds(DISPLAY_TIME_MS); //Call to pause for 20 seconds
while the user memorizes the sequence.

        clearScreen(); //Clears the screen 20 seconds after displaying
the sequence.

        System.out.println("Enter the sequence you remember:");
//Prints user instructions for next step.
        List<String> userSequence = getUSerInput(SEQUENCE_LENGTH);
//Call to get the users input as a list of String.

//Compares the original sequence with the users input.
        if (sequence.equals(userSequence)) {
            System.out.println("WOW thats correct! You have great
memory!"); //If the sequence matches print 'correct message' to
console.
        } else {
            System.out.println("WOMP WOMP! That's worng! The correct
sequence was: " + sequence); //If the sequence does not match print
'wrong message' to console and print the correct answer.
        }
    }

    //Generating a random sequence of symbols/numbers
    //Displaying the sequence
    public static List<String> generateRandomSequence(int length) {
//Defines the method generateRandomSequence.
        Random random = new Random(); //Creates Random object random to
generate random numbers.
        String[] symbols = {"@", ")", "*", "1", "2", "3", "4", "5",
"6", "7", "8", "9", "10", "!", "#", "$", "%"}; //Defines array symbols
containing a set of symbols and numbers.
        List<String> sequence = new ArrayList<>(); //Creates an empty
list called 'sequence'.

        for (int i = 0; i < length; i++) { //Loop from 0 to 'length' (5
times)
            sequence.add(symbols[random.nextInt(symbols.length)]);
//Adds a random symbol from 'symbols' to sequence.
        }
        return sequence; //Returns the sequence list.
    }

    //Pausing the program
    public static void displaySequence(List<String> sequence) {
//Defines method 'displaySequence.
        for (String iteam : sequence) { //Loop through each symbol in
'sequence'.
            System.out.print(iteam + " "); //Print the symbol followed
by a space.
        }
        System.out.println(); //Print a new line in console after the
sequence is displayed.
    }

    //Getting the users input as a list of strings
    public static void waitForSeconds(int milliseconds) { //Defines
method waitForSeconds.
        try {
            Thread.sleep(milliseconds); //Use
Thread.sleep(milliseconds) to pause the program for a specified time.
        } catch (InterruptedException e) { //If an Exception occurs
catch it and print 'error message' to console.
            System.out.println("An error occurred during wait");
        }
    }

    public static List<String> getUSerInput(int length) { //Defines
method getUserInput.
        Scanner170 console = new Scanner170(System.in); //Creates
Scanner object to read user input.
        List<String> userInput = new ArrayList<>(); //Creates an empty
list to store the users input.

        for (int i = 0; i < length; i++) { //Loop 0 - 'length' (5
times).
            userInput.add(console.next()); //Add users next input
(string) to 'userInput'.
        }
        return userInput; //Return the 'userInput' list.
    }

    //Clearing the screen and replacing it with multiple blank lines
    public static void clearScreen() {
        for (int i = 0; i < 50; i++) { //Loop 50 times.
            System.out.println(); //Print an empty line to clear the
console screen.

        }
    }

}
/**
 * Post-conditions
 * Returns a list of "length" which is the set of random
symbols/numbers.
 * Displays the sequence in the console.
 * Program pauses for the whole twenty seconds, to give the user enough
time to memories the sequences.
 * Console is cleared from previous sequences.
 * Returns a list of symbols/numbers input by the user.
 * Final displays weather the user wrote the sequence correctly or not.
 */
```