PALINDROME CHECKER USING STACK AND QUEUE

COMPUTER SCIENCE & ENGINEERING

Course Code: PETV89L

Submitted by:

NAME: PRATEEK JHA (12310132)

Submitted to:

Dr. Preetjot Kaur

Index

Summary of Project

I developed the **Palindrome Checker** as a console-based Java application using core Java concepts. The aim of this project was to determine whether a given string is a palindrome, i.e., it reads the same forward and backward. I implemented the logic using two fundamental data structures — **Stack** and **Queue** — from the **Java Collections Framework**.

To make the checker accurate, I incorporated **input preprocessing** techniques such as converting to lowercase and removing all non-alphanumeric characters using **regular expressions**. The cleaned string was then stored in both a stack and a queue to allow comparison from both directions.

This project not only strengthened my understanding of linear data structures but also helped me apply them in a real-world use case. I made sure to test it with multiple edge cases to ensure correctness. Overall, the experience of designing, coding, debugging, and refining the project has enhanced my skills in **Java programming, string handling, and problem-solving**.

Technologies used

1. **Java** – Core programming language used to implement the application logic.
2. **Java Collections Framework** – Used Stack and Queue interfaces for data structure operations.
3. **Regular Expressions (Regex)** – For sanitizing and preprocessing user input.
4. **Command Line / Terminal** – For running and testing the application in a console environment.
5. **Visual Studio Code (VS Code)** – Used as the primary IDE for writing, compiling, and debugging the Java code.
6. **JDK (Java Development Kit)** – Required for compiling and executing Java programs.

GitHub link

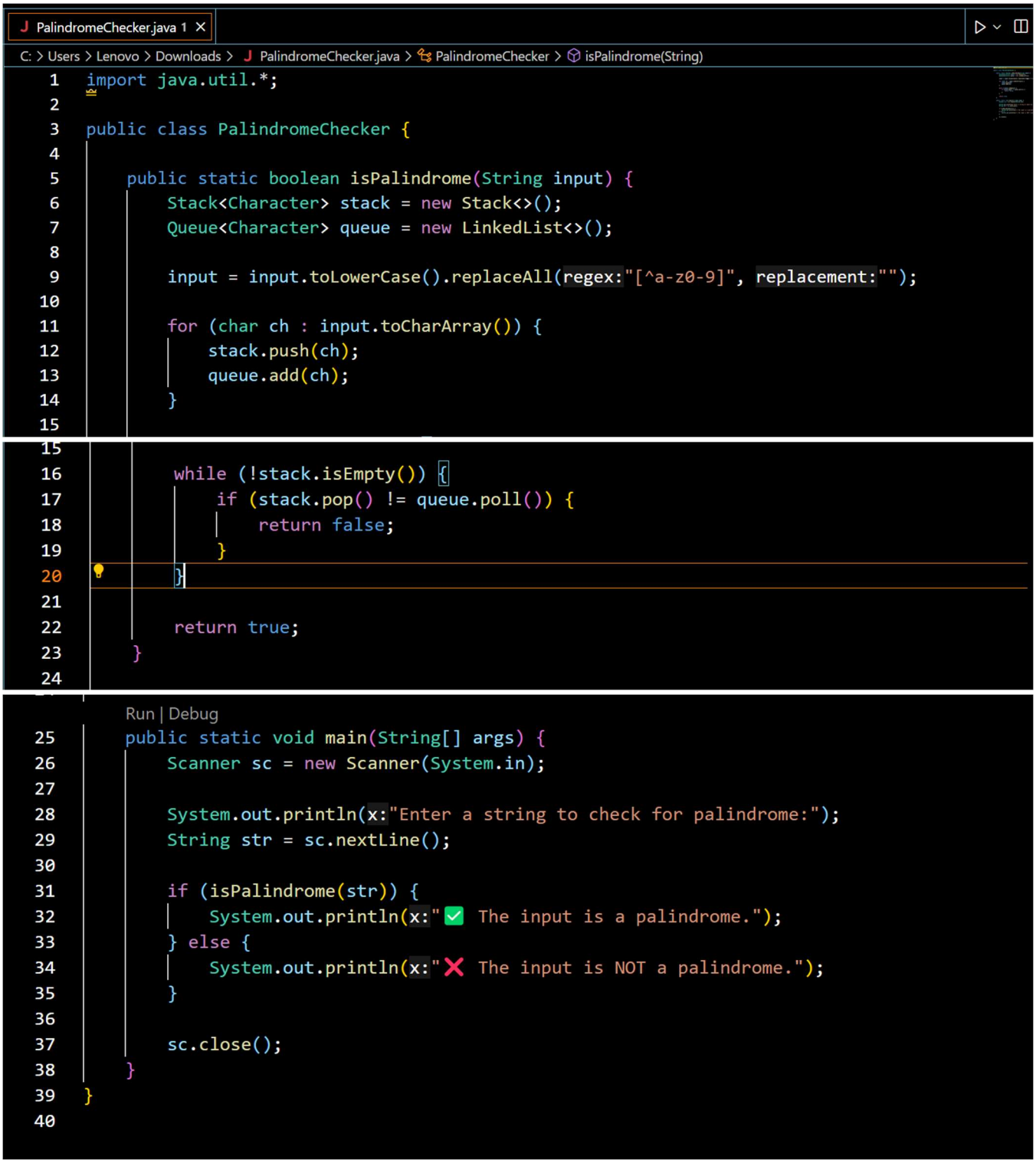
https://github.com/prateekjhaa369/Palindrome-Checker.git

Whole Code :

```
import java.util.*;
public class PalindromeChecker {
public static boolean isPalindrome(String input) {
    Stack<Character> stack = new Stack<>();
    Queue<Character> queue = new LinkedList<>();

    input = input.toLowerCase().replaceAll("[^a-z0-9]", "");

    for (char ch : input.toCharArray()) {
      stack.push(ch);
      queue.add(ch);
    }

    while (!stack.isEmpty()) {
      if (stack.pop() != queue.poll()) {
        return false;
      }
    }
```

```
        return true;
    }

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

        System.out.println("Enter a string to check for palindrome:");
        String str = sc.nextLine();

        if (isPalindrome(str)) {
            System.out.println("✅ The input is a palindrome.");
        } else {
            System.out.println("❌ The input is NOT a palindrome.");
        }

        sc.close();
    }
}
```

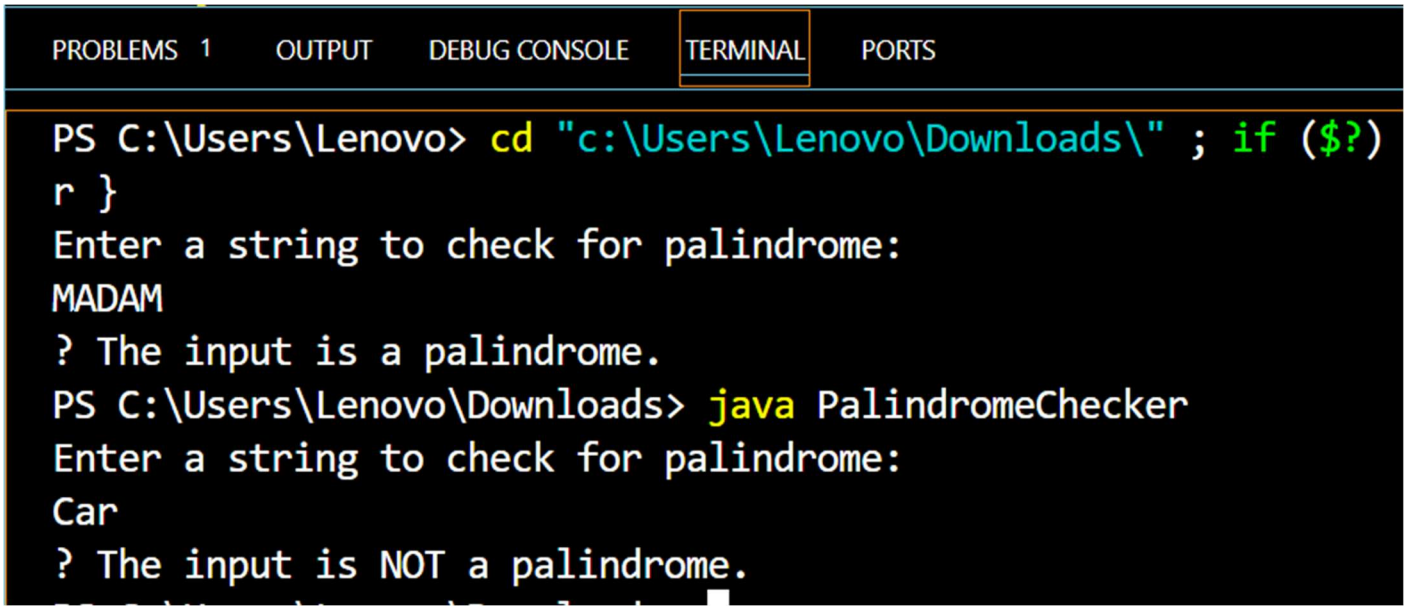
Screenshots



This screenshot shows the complete Java code for the Palindrome Checker program. It includes input handling, preprocessing, and the logic using Stack and Queue to determine whether a string is a palindrome.

In this screenshot, the user enters the input string "Madam". The program processes the string and correctly identifies it as a palindrome, printing

"✅ **The input is a palindrome."** in the terminal.



Here, the input "Hello" is entered. The program evaluates the string and rightly detects that it is **not a palindrome**, printing **"❌ The input is NOT a palindrome."** in the output.

Learning Outcome:

· Gained hands-on experience with core Java programming concepts.
· Understood and implemented linear data structures like Stack and Queue using the Java Collections Framework.
· Learned how to process and clean strings using regular expressions.
· Practiced writing efficient logic to compare characters from different ends.
· Improved skills in designing a user-interactive console-based application.

Limitations and Future Scope

Limitations:

- Only checks one input at a time (no batch processing of multiple inputs).
- Works only in a console environment; lacks a graphical user interface.
- No file saving or history tracking of checked inputs.
- Limited user feedback if input is completely invalid (e.g., blank or special characters only).

Future Scope:-

- Add support for multiple input checks in a single run.
- Develop a GUI using JavaFX or Swing for better user interaction.
- Store checked strings and results in a text file or database for future reference.
- Create a web-based version using JSP/Servlets or Spring Boot.
- Integrate additional features like palindrome detection for numbers and full sentences with punctuation.

Conclusion

The Palindrome Checker project helped me understand how Stack and Queue can be used together to solve real-world problems. By building this application from scratch, I strengthened my Java fundamentals, improved logical thinking, and learned how to manipulate user input effectively. This project has laid a solid foundation for tackling more advanced data structure-based applications in the future.

Greetings

I take this opportunity to extend my warm regards to all the respected faculty members and readers of this report.
It is with great enthusiasm and dedication that I present my project titled “Palindrome Checker Using Stack and Queue”, submitted as part of the course PETV89L under the guidance of Dr. Preetjot Kaur, Our beloved Professor, Department of Computer Science & Engineering, Lovely Professional University.

This project reflects the knowledge I have gained during the summer internship and the sincere effort I have put into understanding and implementing core DSA concepts.
Through this report, I aim to showcase not only the functionality of the application but also the learning journey I undertook while developing it.

I hope this project adds value to my academic progress and contributes meaningfully to the curriculum objectives.