Don Bosco Institute of Technology, Kurla Academic Year 2023-24

EXPERIMENT NO. 6

SEMESTER:III DATE OF PERFORMANCE:01/09/23

SUBJECT: Skill based Lab Course: Object Oriented Programming with Java
DATE OF SUBMISSION: 7/09/23
NAME OF THE STUDENT: Bhanudas Patil ROLL NO.: 40

AIM	Write a program to check whether matrix is symmetric or not(Use BufferedReader class for accep ng data from user)
LEARNING OBJECTIVE	Students will be able to write programs to check whether the matrix is symmetric or not.
LEARNING OUTCOME	Implement programs using strings/string buffer in java
COURSE OUTCOME	Students will be able to explain the various Java constructs and will be able to compare classes, objects, packages, arrays and strings.
PROGRAM OUTCOME	PO1,PO2,PO3,PO4,PO5,PO9,PO10,PSO1,PSO2,PSO3.
BLOOM'S TAXONOMY LEVEL	Apply.
THEORY	1.Explain following terms ● InputStreamReader. ● BufferedReder

Don Bosco Institute of Technology, Kurla Academic Year 2023-24

LAB EXERCISE

- Every student will execute the Program as per the instruc on and a ach output as per the requirements.

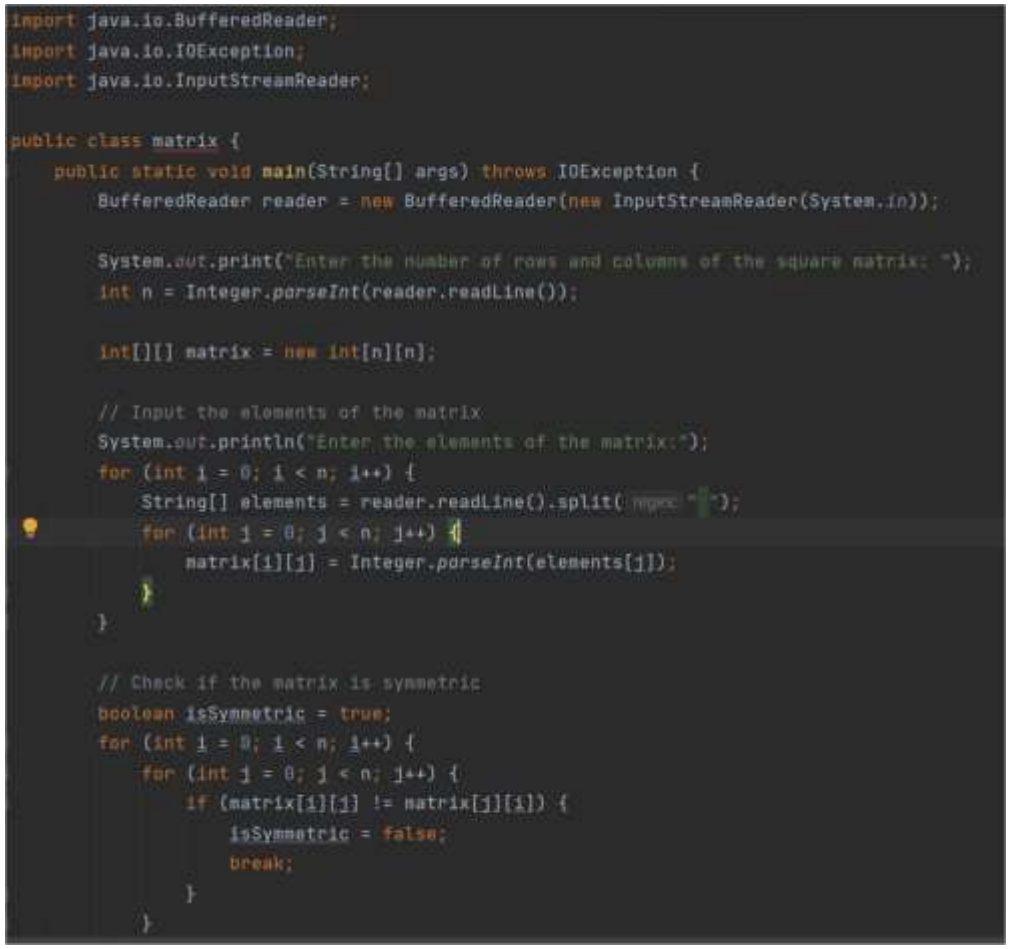
Code:

```
import java.io.BufferedReader; import
java.io.IOException; import
java.io.InputStreamReader;
 public class matrix {
   public static void main(String[] args) throws IOException {
      BufferedReader reader = new BufferedReader(new
InputStreamReader(System.in));

      System.out.print("Enter the number of rows and columns of the square matrix: ");
int n = Integer.parseInt(reader.readLine());
       int[][] matrix = new int[n][n];

      // Input the elements of the matrix
      System.out.println("Enter the elements of the matrix:");        for (int i = 0; i < n;
i++) {
          String[] elements = reader.readLine().split(" ");           for (int j = 0; j <
n; j++) {               matrix[i][j] = Integer.parseInt(elements[j]);
          }
      }

      // Check if the matrix is symmetric        boolean isSymmetric
= true;        for (int i = 0; i < n; i++) {            for (int j = 0; j < n;
j++) {               if (matrix[i][j] != matrix[j][i]) {
isSymmetric = false;                   break;
            }          }           if
(!isSymmetric) {               break;
          }
      }

      // Display the matrix
      System.out.println("Matrix:");        for (int i = 0;
i < n; i++) {           for (int j = 0; j < n; j++) {
            System.out.print(matrix[i][j] + " ");
          }
          System.out.println();
      }
```

Don Bosco Institute of Technology, Kurla Academic Year 2023-24

```
    // Display the result                    if
(isSymmetric) {
        System.out.println("The matrix is symmetric.");
    } else {
        System.out.println("The matrix is not symmetric.");
    }
  }
}
```

PICTURE OF THE CODE:



```java
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;

public class matrix {
    public static void main(String[] args) throws IOException {
        BufferedReader reader = new BufferedReader(new InputStreamReader(System.in));

        System.out.print("Enter the number of rows and columns of the square matrix: ");
        int n = Integer.parseInt(reader.readLine());

        int[][] matrix = new int[n][n];

        // Input the elements of the matrix
        System.out.println("Enter the elements of the matrix:");
        for (int i = 0; i < n; i++) {
            String[] elements = reader.readLine().split( regex: " ");
            for (int j = 0; j < n; j++) {
                matrix[i][j] = Integer.parseInt(elements[j]);
            }
        }

        // Check if the matrix is symmetric
        boolean isSymmetric = true;
        for (int i = 0; i < n; i++) {
            for (int j = 0; j < n; j++) {
                if (matrix[i][j] != matrix[j][i]) {
                    isSymmetric = false;
                    break;
                }
            }
```

```
                    isSymmetric = false;
                    break;
                }
            }
            if (!isSymmetric) {
                break;
            }
        }

        // Display the matrix
        System.out.println("Matrix:");
        for (int i = 0; i < n; i++) {
            for (int j = 0; j < n; j++) {
                System.out.print(matrix[i][j] + " ");
            }
            System.out.println();
        }

        // Display the result
        if (isSymmetric) {
            System.out.println("The matrix is symmetric.");
        } else {
            System.out.println("The matrix is not symmetric.");
        }
    }
}
```

Output:

```
Enter the number of rows and columns of the square matrix: 3
Enter the elements of the matrix:
1 2 3
2 4 5
3 5 6
Matrix:
1 2 3
2 4 5
3 5 6
The matrix is symmetric.
```

Don Bosco Institute of Technology, Kurla Academic Year 2023-24

REFERENCES	1.h ps://www.javatpoint.com/java-oops-concepts 2.h ps://www.geeksforgeeks.org/object-oriented-programming-oops -concept-in-java/ 3.h ps://www.javatpoint.com/java-oops-concepts

Don Bosco Institute of Technology, Kurla Academic Year 2023-24

THEORY QUESTIONS

1. InputStreamReader:

InputStreamReader is a class in Java that bridges the conversion between byte streams and character streams. It is o en used for reading text from an InputStream, such as reading from a file or reading data from the standard input (e.g., the keyboard).

Key points about InputStreamReader:

- It is part of the java.io package.
- It wraps an InputStream and converts the bytes read from the stream into characters based on a specified character encoding (e.g., UTF-8, ISO-8859-1).
- It provides methods for reading characters, making it easier to work with textual data.
- It is commonly used with other higher-level classes like BufferedReader or `Scanner` to efficiently read and process text data.

2. BufferedReader:

BufferedReader is another class in Java, also part of the java.io package, that provides efficient reading of text from character input streams. It is o en used for reading text from sources like files, network sockets, or input streams.

Key points about BufferedReader:

- It can improve reading performance by buffering data, meaning it reads larger chunks of data at once from the underlying input stream.
- It provides methods like readLine() for reading lines of text and `read()` for reading individual characters.
- It is commonly used for reading user input or reading text from files because of its efficient buffering mechanism.