

Lab Report 4

Submitted to:

Shakib Mahmud Dipto
Faculty

Submitted by:

Sumaiya Akter
ID: 201014071

Department of CSE
Summer'24

Course code: CSE 2104

Course Title: Object Oriented Programming Lab

Section: 01

University of Liberal Arts Bangladesh

July 01, 2024

Problem 01

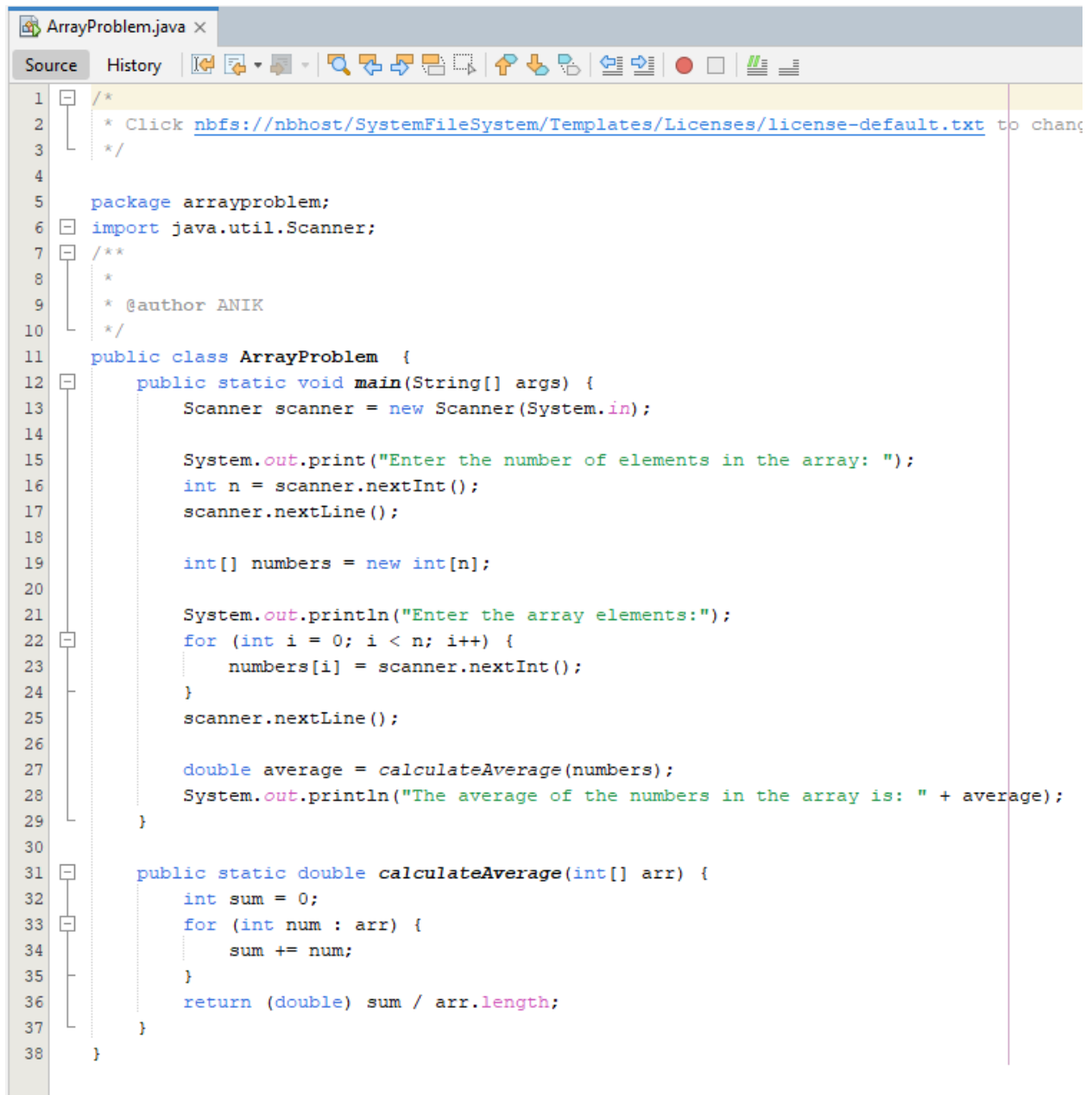
Code Explanation :

This Java program calculates the average of a set of numbers entered by the user. The program first prompts the user to enter the number of elements in the array, then it asks the user to enter the array elements. The `calculateAverage` method is used to compute the average of the numbers in the array, and the result is displayed to the user.

The program uses a `Scanner` object to read user input, and it creates an array of integers to store the numbers. The `for` loop is used to iterate through the array and add up all the numbers, and the `calculateAverage` method divides the sum by the length of the array to compute the average.

Code Screenshots:

Input:



```

1  /*
2   * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change
3   *
4   */
5   package arrayproblem;
6   import java.util.Scanner;
7   /**
8    *
9    * @author ANIK
10   */
11  public class ArrayProblem {
12      public static void main(String[] args) {
13          Scanner scanner = new Scanner(System.in);
14
15          System.out.print("Enter the number of elements in the array: ");
16          int n = scanner.nextInt();
17          scanner.nextLine();
18
19          int[] numbers = new int[n];
20
21          System.out.println("Enter the array elements:");
22          for (int i = 0; i < n; i++) {
23              numbers[i] = scanner.nextInt();
24          }
25          scanner.nextLine();
26
27          double average = calculateAverage(numbers);
28          System.out.println("The average of the numbers in the array is: " + average);
29      }
30
31      public static double calculateAverage(int[] arr) {
32          int sum = 0;
33          for (int num : arr) {
34              sum += num;
35          }
36          return (double) sum / arr.length;
37      }
38  }

```

Output:

```

Output - Run (ArrayProblem) x
cd F:\sumaiya the V.I.P\sem 14\OOP\assignments\lab 4\ArrayProblem; "JAVA_HOME=C:\\Program Files\\Java\\jdk-2
Scanning for projects...

-----< com.mycompany:ArrayProblem >-----
Building ArrayProblem 1.0-SNAPSHOT
from pom.xml
-----[ jar ]-----

--- resources:3.3.1:resources (default-resources) @ ArrayProblem ---
skip non existing resourceDirectory F:\sumaiya the V.I.P\sem 14\OOP\assignments\lab 4\ArrayProblem\src\main\

--- compiler:3.11.0:compile (default-compile) @ ArrayProblem ---
Changes detected - recompiling the module! :source
Compiling 1 source file with javac [debug target 21] to target\classes

--- exec:3.1.0:exec (default-cli) @ ArrayProblem ---
Enter the number of elements in the array: 3
Enter the array elements:
5
8
9
The average of the numbers in the array is: 7.333333333333333

BUILD SUCCESS

Total time: 17.541 s
Finished at: 2024-07-01T23:29:44+06:00
|

```

Problem 02

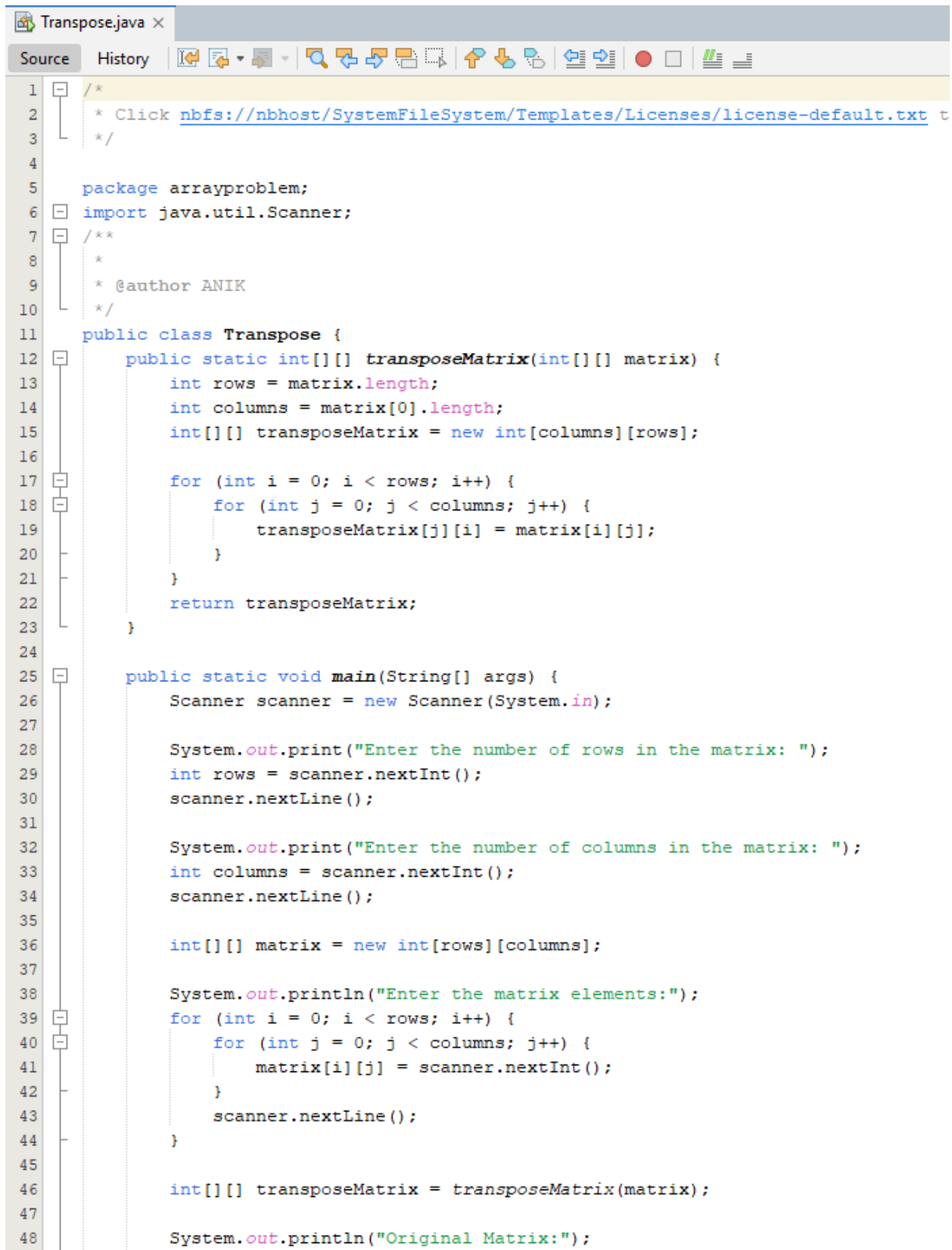
Code Explanation:

This Java program transposes a matrix entered by the user. The `transposeMatrix` method takes a 2D integer array as input and returns its transpose. The `printMatrix` method is used to display the original and transposed matrices.

The program prompts the user to enter the number of rows and columns in the matrix, then it asks the user to enter the matrix elements. The `transposeMatrix` method is called with the user-entered matrix, and the resulting transposed matrix is displayed using the `printMatrix` method.

Code Screenshot:

Input:



```

1  /*
2   * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to
3   */
4
5  package arrayproblem;
6  import java.util.Scanner;
7  /**
8   *
9   * @author ANIK
10  */
11 public class Transpose {
12     public static int[][] transposeMatrix(int[][] matrix) {
13         int rows = matrix.length;
14         int columns = matrix[0].length;
15         int[][] transposeMatrix = new int[columns][rows];
16
17         for (int i = 0; i < rows; i++) {
18             for (int j = 0; j < columns; j++) {
19                 transposeMatrix[j][i] = matrix[i][j];
20             }
21         }
22         return transposeMatrix;
23     }
24
25     public static void main(String[] args) {
26         Scanner scanner = new Scanner(System.in);
27
28         System.out.print("Enter the number of rows in the matrix: ");
29         int rows = scanner.nextInt();
30         scanner.nextLine();
31
32         System.out.print("Enter the number of columns in the matrix: ");
33         int columns = scanner.nextInt();
34         scanner.nextLine();
35
36         int[][] matrix = new int[rows][columns];
37
38         System.out.println("Enter the matrix elements:");
39         for (int i = 0; i < rows; i++) {
40             for (int j = 0; j < columns; j++) {
41                 matrix[i][j] = scanner.nextInt();
42             }
43             scanner.nextLine();
44         }
45
46         int[][] transposeMatrix = transposeMatrix(matrix);
47
48         System.out.println("Original Matrix:");

```

```
49     printMatrix(matrix);  
50     System.out.println("\nTransposed Matrix:");  
51     printMatrix(transposeMatrix);  
52 }  
53  
54 public static void printMatrix(int[][] matrix) {  
55     for (int[] row : matrix) {  
56         for (int num : row) {  
57             System.out.print(num + " ");  
58         }  
59         System.out.println();  
60     }  
61 }  
62 }
```

Output:

Department of Computer Science & Engineering

UNIVERSITY OF LIBERAL ARTS BANGLADESH

```

Output - Run (Transpose) ×
cd F:\sumaiya the V.I.P\sem 14\OOP\assignments\lab 4\Transpose; "J
Scanning for projects...

-----< com.mycompany:Transpose >-----
Building Transpose 1.0-SNAPSHOT
from pom.xml
-----[ jar ]-----

--- resources:3.3.1:resources (default-resources) @ Transpose ---
skip non existing resourceDirectory F:\sumaiya the V.I.P\sem 14\OO

--- compiler:3.11.0:compile (default-compile) @ Transpose ---
Nothing to compile - all classes are up to date

--- exec:3.1.0:exec (default-cli) @ Transpose ---
Enter the number of rows in the matrix: 6
Enter the number of columns in the matrix: 4
Enter the matrix elements:
6
9
0
0
4
3
1
9
8
0
0
4
2
5
7
9
0
7
4
3
2
4
5
5

Original Matrix:
6 9 0 0
4 3 1 9
8 0 0 4
2 5 7 9
0 7 4 3
2 4 5 5

Transposed Matrix:
6 4 8 2 0 2
9 3 0 5 7 4

```

```

0 1 0 7 4 5
0 9 4 9 3 5

```

```

-----
BUILD SUCCESS
-----

```

```

Total time: 53.091 s
Finished at: 2024-07-01T23:34:45+06:00
-----

```

Problem 03

Code Explanation:

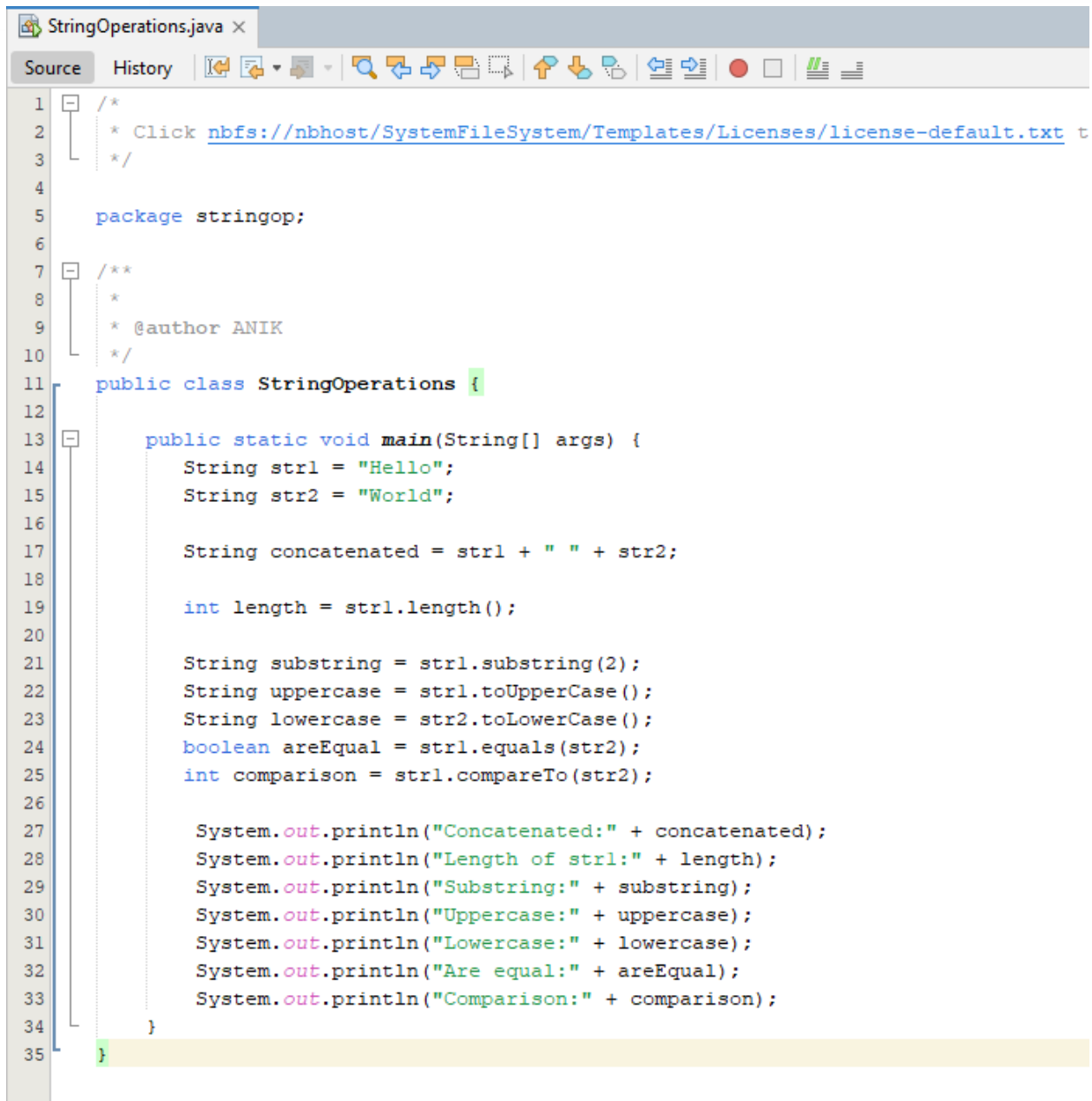
The provided Java program demonstrates various string operations using common string methods. The program starts by creating two string variables, 'str1' and 'str2', with sample values. It then performs a series of operations, including concatenating the two strings using the '+' operator and the 'concat()' method, calculating the length of 'str1' using the 'length()' method, extracting a substring of 'str1' starting from index 2 using the 'substring()' method, converting 'str1' to uppercase using the 'toUpperCase()' method and 'str2' to lowercase using the 'toLowerCase()' method, comparing 'str1' and 'str2' using the 'equals()' method to check for equality, and comparing them lexicographically using the 'compareTo()' method. Finally, the program outputs the results of these string operations.

Code Screenshot:

Input:

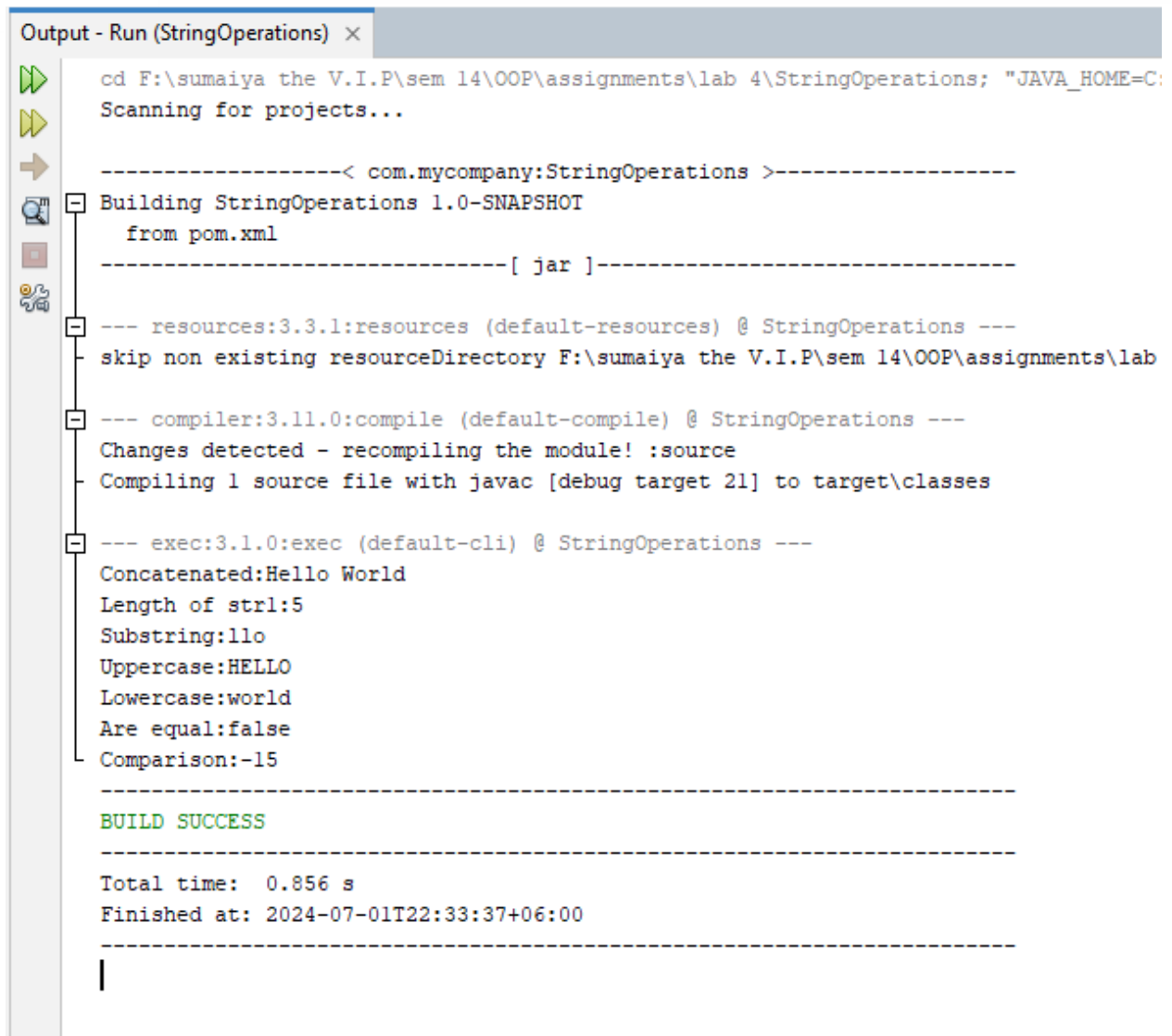
Department of Computer Science & Engineering

UNIVERSITY OF LIBERAL ARTS BANGLADESH



```
StringOperations.java x
Source History
1  /**
2   * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt t
3   */
4
5  package stringop;
6
7  /**
8   *
9   * @author ANIK
10  */
11  public class StringOperations {
12
13      public static void main(String[] args) {
14          String str1 = "Hello";
15          String str2 = "World";
16
17          String concatenated = str1 + " " + str2;
18
19          int length = str1.length();
20
21          String substring = str1.substring(2);
22          String uppercase = str1.toUpperCase();
23          String lowercase = str2.toLowerCase();
24          boolean areEqual = str1.equals(str2);
25          int comparison = str1.compareTo(str2);
26
27          System.out.println("Concatenated:" + concatenated);
28          System.out.println("Length of str1:" + length);
29          System.out.println("Substring:" + substring);
30          System.out.println("Uppercase:" + uppercase);
31          System.out.println("Lowercase:" + lowercase);
32          System.out.println("Are equal:" + areEqual);
33          System.out.println("Comparison:" + comparison);
34      }
35  }
```

Output:



```

Output - Run (StringOperations) x
cd F:\sumaiya the V.I.P\sem 14\OOP\assignments\lab 4\StringOperations; "JAVA_HOME=C:
Scanning for projects...

-----< com.mycompany:StringOperations >-----
Building StringOperations 1.0-SNAPSHOT
from pom.xml
-----[ jar ]-----

--- resources:3.3.1:resources (default-resources) @ StringOperations ---
skip non existing resourceDirectory F:\sumaiya the V.I.P\sem 14\OOP\assignments\lab

--- compiler:3.11.0:compile (default-compile) @ StringOperations ---
Changes detected - recompiling the module! :source
Compiling 1 source file with javac [debug target 21] to target\classes

--- exec:3.1.0:exec (default-cli) @ StringOperations ---
Concatenated:Hello World
Length of str1:5
Substring:llo
Uppercase:HELLO
Lowercase:world
Are equal:false
Comparison:-15

BUILD SUCCESS

Total time: 0.856 s
Finished at: 2024-07-01T22:33:37+06:00
|

```

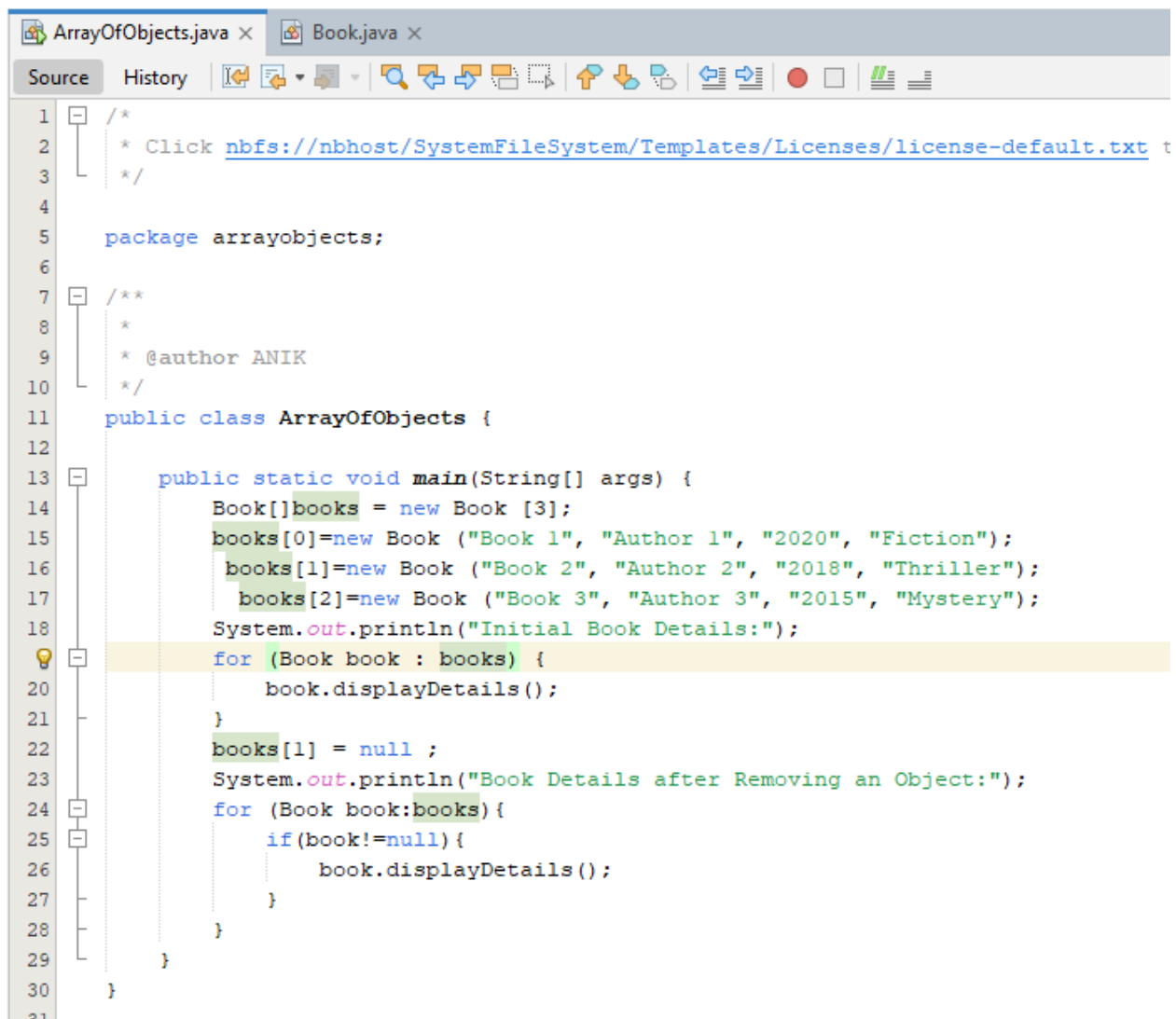
Problem 04

Code Explanation:

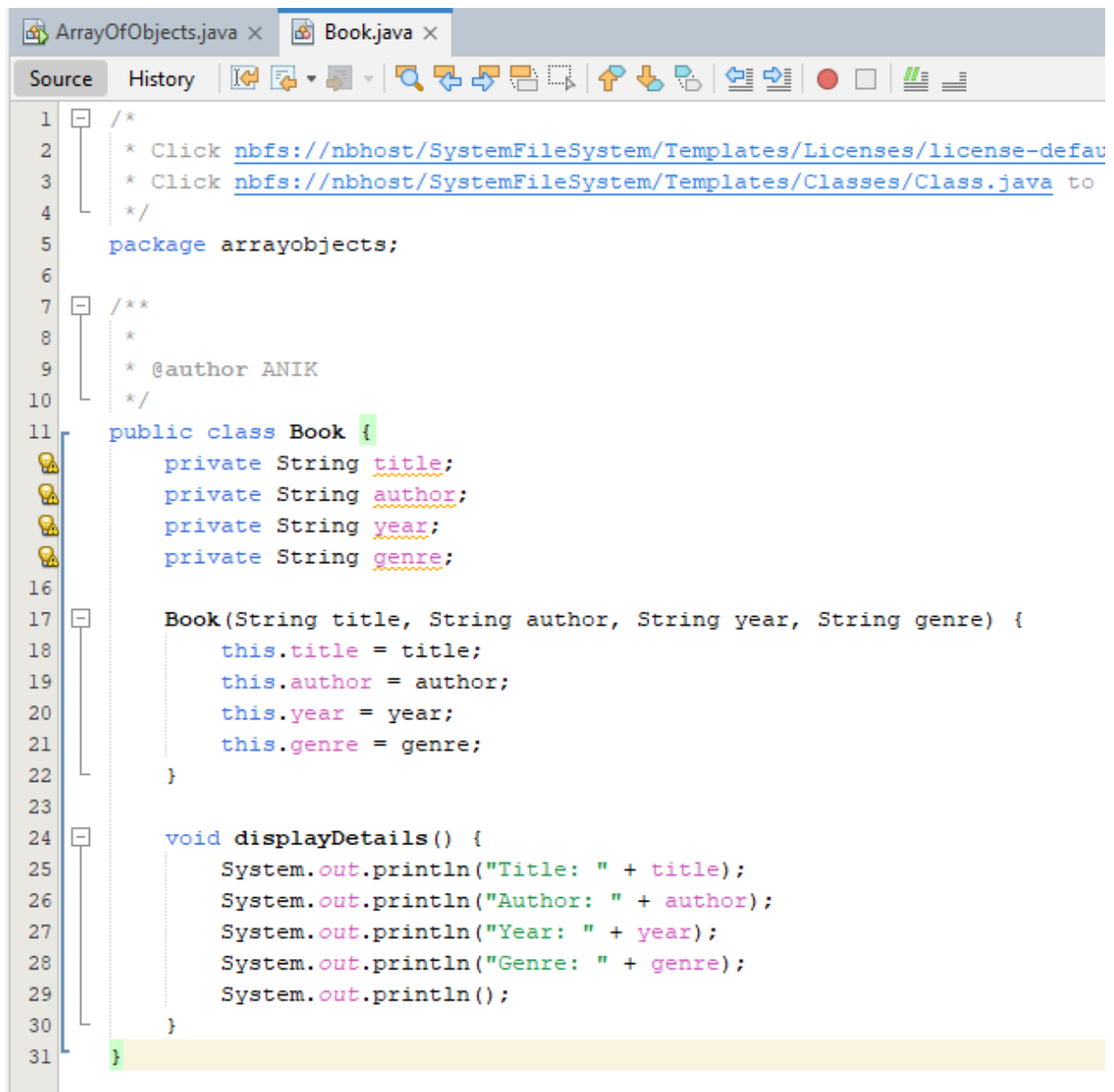
The provided Java program creates a 'Book' class with properties like 'title', 'author', 'year', and 'genre', and a parameterized constructor to initialize objects. It stores 3 'Book' objects in an array, displays their details using an object method, removes an object by assigning 'null' to an index, and displays the remaining objects. The program showcases the creation of objects, their storage in an array, and the removal of an object by assigning 'null'.

Code Screenshot:

Input:



```
1  /*
2   * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to
3   */
4
5  package arrayobjects;
6
7  /**
8   *
9   * @author ANIK
10   */
11  public class ArrayOfObjects {
12
13      public static void main(String[] args) {
14          Book[] books = new Book [3];
15          books[0]=new Book ("Book 1", "Author 1", "2020", "Fiction");
16          books[1]=new Book ("Book 2", "Author 2", "2018", "Thriller");
17          books[2]=new Book ("Book 3", "Author 3", "2015", "Mystery");
18          System.out.println("Initial Book Details:");
19          for (Book book : books) {
20              book.displayDetails();
21          }
22          books[1] = null ;
23          System.out.println("Book Details after Removing an Object:");
24          for (Book book:books) {
25              if(book!=null){
26                  book.displayDetails();
27              }
28          }
29      }
30  }
```



```
1  /*
2   * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to
3   * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to
4   */
5   package arrayobjects;
6
7   /**
8    *
9    * @author ANIK
10   */
11  public class Book {
12      private String title;
13      private String author;
14      private String year;
15      private String genre;
16
17      Book(String title, String author, String year, String genre) {
18          this.title = title;
19          this.author = author;
20          this.year = year;
21          this.genre = genre;
22      }
23
24      void displayDetails() {
25          System.out.println("Title: " + title);
26          System.out.println("Author: " + author);
27          System.out.println("Year: " + year);
28          System.out.println("Genre: " + genre);
29          System.out.println();
30      }
31  }
```

Output:

```

Output - Run (ArrayOfObjects) x
cd F:\sumaiya the V.I.P\sem 14\OOP\assignments\lab 4\ArrayOfObjects; "JAVA_HOME=C:\\Program Files
Scanning for projects...

-----< com.mycompany:ArrayOfObjects >-----
Building ArrayOfObjects 1.0-SNAPSHOT
  from pom.xml
-----[ jar ]-----

--- resources:3.3.1:resources (default-resources) @ ArrayOfObjects ---
skip non existing resourceDirectory F:\sumaiya the V.I.P\sem 14\OOP\assignments\lab 4\ArrayOfObje

--- compiler:3.11.0:compile (default-compile) @ ArrayOfObjects ---
Changes detected - recompiling the module! :source
Compiling 2 source files with javac [debug target 21] to target\classes

--- exec:3.1.0:exec (default-cli) @ ArrayOfObjects ---
Initial Book Details:
Title: Book 1
Author: Author 1
Year: 2020
Genre: Fiction

Title: Book 2
Author: Author 2
Year: 2018
Genre: Thriller

Title: Book 3
Author: Author 3
Year: 2015
Genre: Mystery

Book Details after Removing an Object:
Title: Book 1
Author: Author 1
Year: 2020
Genre: Fiction

Title: Book 3
Author: Author 3
Year: 2015
Genre: Mystery

-----
BUILD SUCCESS
-----

Total time: 0.849 s
Finished at: 2024-07-01T22:55:44+06:00
-----
|

```

Problem 05:

Code Explanation:

The provided Java program demonstrates the usage of 'ArrayList' and 'LinkedList' data structures to store and manipulate integers. It creates an 'ArrayList' and a 'LinkedList', performs various operations on them, such as adding elements, getting the size, accessing elements, modifying elements, removing elements, and sorting the elements. The program showcases the differences between the two data structures and how to leverage their respective strengths, such as fast random access for 'ArrayList' and efficient insertion and deletion for 'LinkedList'. The program then prints the contents of both data structures.

Code Screenshot:

Input:

```

ListOperations.java x
Source History
4
5 package listoperations;
6 import java.util.*;
7 /**
8  *
9  * @author ANIK
10 */
11 public class ListOperations {
12
13     public static void main(String[] args) {
14         List<Integer> arrayList = new ArrayList<>();
15         List<Integer> linkedList = new LinkedList<>();
16
17         arrayList.add(10);
18         arrayList.add(20);
19         arrayList.add(30);
20
21         linkedList.add(40);
22         linkedList.add(50);
23         linkedList.add(60);
24
25         int arrayListSize = arrayList.size();
26         int linkedListSize = linkedList.size();
27
28         int arrayListElement = arrayList.get(1);
29         int linkedListElement = linkedList.get(2);
30
31         arrayList.set(1, 25);
32         linkedList.set(2, 55);
33
34         arrayList.remove(0);
35         linkedList.remove(1);
36
37         Collections.sort(arrayList);
38         Collections.sort(linkedList);
39         System.out.println("ArrayList:");
40         for(int num: arrayList){
41             System.out.println(num);
42         }
43         System.out.println("linkedList:");
44         for (int num: linkedList){
45             System.out.println(num);
46         }
47     }
48 }
49

```

Output:

Output - Run (ListOperations) ×

```
cd F:\sumaiya the V.I.P\sem 14\OOP\assignments\lab 4\ListOperations; "JAVA_HOME=C:\\Progr
Scanning for projects...

-----< com.mycompany:ListOperations >-----
Building ListOperations 1.0-SNAPSHOT
  from pom.xml
-----[ jar ]-----

--- resources:3.3.1:resources (default-resources) @ ListOperations ---
skip non existing resourceDirectory F:\sumaiya the V.I.P\sem 14\OOP\assignments\lab 4\Li

--- compiler:3.11.0:compile (default-compile) @ ListOperations ---
Changes detected - recompiling the module! :source
Compiling 1 source file with javac [debug target 21] to target\classes

--- exec:3.1.0:exec (default-cli) @ ListOperations ---
ArrayList:
25
30
linkedList:
40
55

-----
BUILD SUCCESS
-----

Total time: 0.885 s
Finished at: 2024-07-01T23:23:13+06:00
-----
|
```