



Experiment 2.2

StudentName:Suraj Sagar

UID:21BCS3388

Branch: BE-CSE

Section/Group:NTTP-601-A

Semester: 6

Date of Performance:20-02-2024

Subject Name: Project Based Learning in Java with Lab

Subject Code:21CSH-319

- 1. Aim:** Create a program to collect unique symbols from a set of cards using set interface.
- 2. Objective:** This cards game consists of N number of cards. Get N number of cards details from the user and store the values in Card object with the attributes symbol and number. Store all the cards in a map with symbol as its key and list of cards as its value. Map is used here to easily group all the cards based on their symbol.

Once all the details are captured print all the distinct symbols in alphabetical order from the Map. For each symbol print all the card details, number of cards and their sum respectively.

3. Algo. /Approach and output:

```
public class Card implements Comparable<Card> {  
    private char symbol;  
    private int number;  
    public Card() {}  
  
    public Card(char symbol, int number) {  
        super();  
        this.symbol = symbol;  
        this.number = number;  
    }  
  
    public char getSymbol() {  
        return symbol;  
    }  
  
    public void setSymbol(char symbol) {  
        this.symbol = symbol;  
    }  
  
    public int getNumber() {  
        return number;  
    }  
}
```



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
public void setNumber(int number) {
    this.number = number;
}

@Override
public String toString() {
    return "Card [symbol=" + symbol + ", number=" + number + "]";
}

@Override
public int compareTo(Card o) {
    if (this.symbol < o.symbol) return -1;
    else if (this.symbol > o.symbol) return 1;
    else return 0;
}

@Override
public int hashCode() {
    return String.valueOf(symbol).hashCode();
}

@Override
public boolean equals(Object obj){
    if (obj instanceof Card) {
        Card card = (Card) obj;
        return (card.symbol == this.symbol);
    } else {
        return false;
    }
}

import java.util.HashSet;
import java.util.Scanner;
import java.util.Set;

public class exp5{

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        Set<Card> set = new HashSet<>();

        for (int i = 0; i < 8; i++) {
            System.out.println("Enter a card:");
            Card card = new Card();

            card.setSymbol(sc.nextLine().charAt(0));
```



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
        card.setNumber(sc.nextInt());
        sc.nextLine();

        set.add(card);
    }
    System.out.println("Four symbols gathered in eight cards.");
    System.out.println("Cards in Set are:");

    for (Card card : set)
        System.out.println(card.getSymbol() + " " + card.getNumber());

    sc.close();
}

}
```

4. Output:

```
(base) PS D:\codes\javaCode> cd "d:\codes\javaCode\sem6Java\" ; if ($?) { javac exp5.java } ; if ($?) { java exp5 }
Enter a card:
a
1
Enter a card:
a
2
Enter a card:
d
6
Enter a card:
c
2
Enter a card:
d
1
Enter a card:
c
1
Enter a card:
b
2
Enter a card:
b
3
Four symbols gathered in eight cards.
Cards in Set are:
a 1
b 2
c 2
d 6
```



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

Experiment 2.3

Name: Suraj Sagar

UID: 21BCS3388

Branch: BE-CSE

Section/Group: NTPP-601-A

Semester: 6

Date of Performance: 27-02-2024

Subject Name: Project Based Learning in Java with Lab

Subject Code: 21CSH-319

Aim: Write a Program to perform the basic operations like insert, delete, display and search in list. List contains String object items where these operations are to be performed.

Code:

```
import java.util.*;
public class exp6 {
    private static void insert(ArrayList<String> list, String item){
        list.add(item);
        System.out.println(item + " inserted successfully.");
    }
    private static void delete(ArrayList<String> list, String item){
        if (list.remove(item)) System.out.println(item + " deleted successfully.");

        else System.out.println(item + " not found in the list.");
    }

    private static void display(ArrayList<String> list){
        if (list.isEmpty()){
            System.out.println("List is empty.");
        }
        else{
            System.out.println("List contents:");
            for (String item : list)
                System.out.println(item);
        }
    }
    private static void search(ArrayList<String> list, String item){

        if (list.contains(item)) System.out.println(item + " found in the list.");
        else System.out.println(item + " not found in the list.");
    }

    public static void main(String[] args){
        ArrayList<String> stringList = new ArrayList<>();
        Scanner scanner = new Scanner(System.in);
        boolean running = true;
        while (running){
```



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
System.out.println("\nSelect an option:");
System.out.println("1. Insert");
System.out.println("2. Delete");
System.out.println("3. Display");
System.out.println("4. Search");
System.out.println("5. Exit");
int choice = scanner.nextInt();
scanner.nextLine();
switch (choice){

    case 1:
        System.out.println("Enter the string to insert:"); String insertString = scanner.nextLine();
        insert(stringList, insertString);
        break;
    case 2:
        System.out.println("Enter the string to delete:"); String deleteString = scanner.nextLine();
        delete(stringList, deleteString);
        break;
    case 3:
        display(stringList); break;
    case 4:
        System.out.println("Enter the string to search:"); String searchString = scanner.nextLine();
        search(stringList, searchString);
        break;
    case 5:
        running = false; System.out.println("Exiting..."); break;
    default:
        System.out.println("Invalid choice!");
    }
}
scanner.close();
}
```

Output:

```
PROBLEMS 39 OUTPUT DEBUG CONSOLE TERMINAL PORTS
(base) PS D:\codes\javaCode> cd "d:\codes\javaCode\sem6Java\" ; if ($?) { javac exp6.java } ; if ($?) { java exp6 }

Select an option:
1. Insert
2. Delete
3. Display
4. Search
5. Exit
1
Enter the string to insert:
hello hi
hello hi inserted successfully.
```



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

Select an option:

1. Insert
2. Delete
3. Display
4. Search
5. Exit

3

List contents:

hello hi

Select an option:

1. Insert
2. Delete
3. Display
4. Search
5. Exit

1

Enter the string to insert:

my name is Yash

my name is Yash inserted successfully.

Select an option:

1. Insert
2. Delete
3. Display
4. Search
5. Exit

3

List contents:

hello hi

my name is Yash

Select an option:

1. Insert
2. Delete
3. Display
4. Search
5. Exit

4

Enter the string to search:

hello hi

hello hi found in the list.

Select an option:

1. Insert
2. Delete
3. Display
4. Search
5. Exit

2

Enter the string to delete:

hello hi

hello hi deleted successfully.



Experiment 2.4

StudentName:Suraj Sagar

UID:21BCS3388

Branch: CSE

Section/Group: NTPP-601-A

Semester: 6

Date of Performance:05/03/2024

Subject Name: Project based learning in Java with Lab

Subject Code: 21CSH-319

1. Aim: Write a menu-based program to create a Java Application 'Employee Management System'.

2. Objective: The objective of this program is to create an application that should gather details of the employee like employee name, employee id, designation and salary and store it in a file. The application should display all the employee details.

3. Code:

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

class Employee {
    private int id;
    private String name;
    private int age;
    private double salary;

    public Employee(int id, String name, int age, double salary) {
        this.id = id;
        this.name = name;
        this.age = age;
        this.salary = salary;
    }

    @Override
    public String toString() {
        return id + " " + name + " " + age + " " + salary;
    }
}

public class EmployeeManagementSystem {
    private static final String FILE_NAME = "employees.txt";
```



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
private static Scanner scanner = new Scanner(System.in);

public static void main(String[] args) {
    List<Employee> employees = new ArrayList<>();
    int choice;

    do {
        System.out.println("Main Menu");
        System.out.println("1. Add an Employee");
        System.out.println("2. Display All");
        System.out.println("3. Exit");
        System.out.print("Enter your choice: ");
        choice = scanner.nextInt();
        scanner.nextLine(); // Consume newline

        switch (choice) {
            case 1: addEmployee(employees);
                    break;
            case 2: displayAllEmployees(employees);
                    break;
            case 3: System.out.println("Exiting the System");
                    break;
            default: System.out.println("Invalid choice. Please enter again.");
        }
    } while (choice != 3);
}

private static void addEmployee(List<Employee> employees) {
    System.out.print("Enter Employee ID: ");
    int id = scanner.nextInt();
    scanner.nextLine(); // Consume newline

    System.out.print("Enter Employee Name: ");
    String name = scanner.nextLine();

    System.out.print("Enter Employee Age: ");
    int age = scanner.nextInt();
    scanner.nextLine(); // Consume newline

    System.out.print("Enter Employee Salary: ");
    double salary = scanner.nextDouble();
    scanner.nextLine(); // Consume newline

    Employee employee = new Employee(id, name, age, salary);
    employees.add(employee);

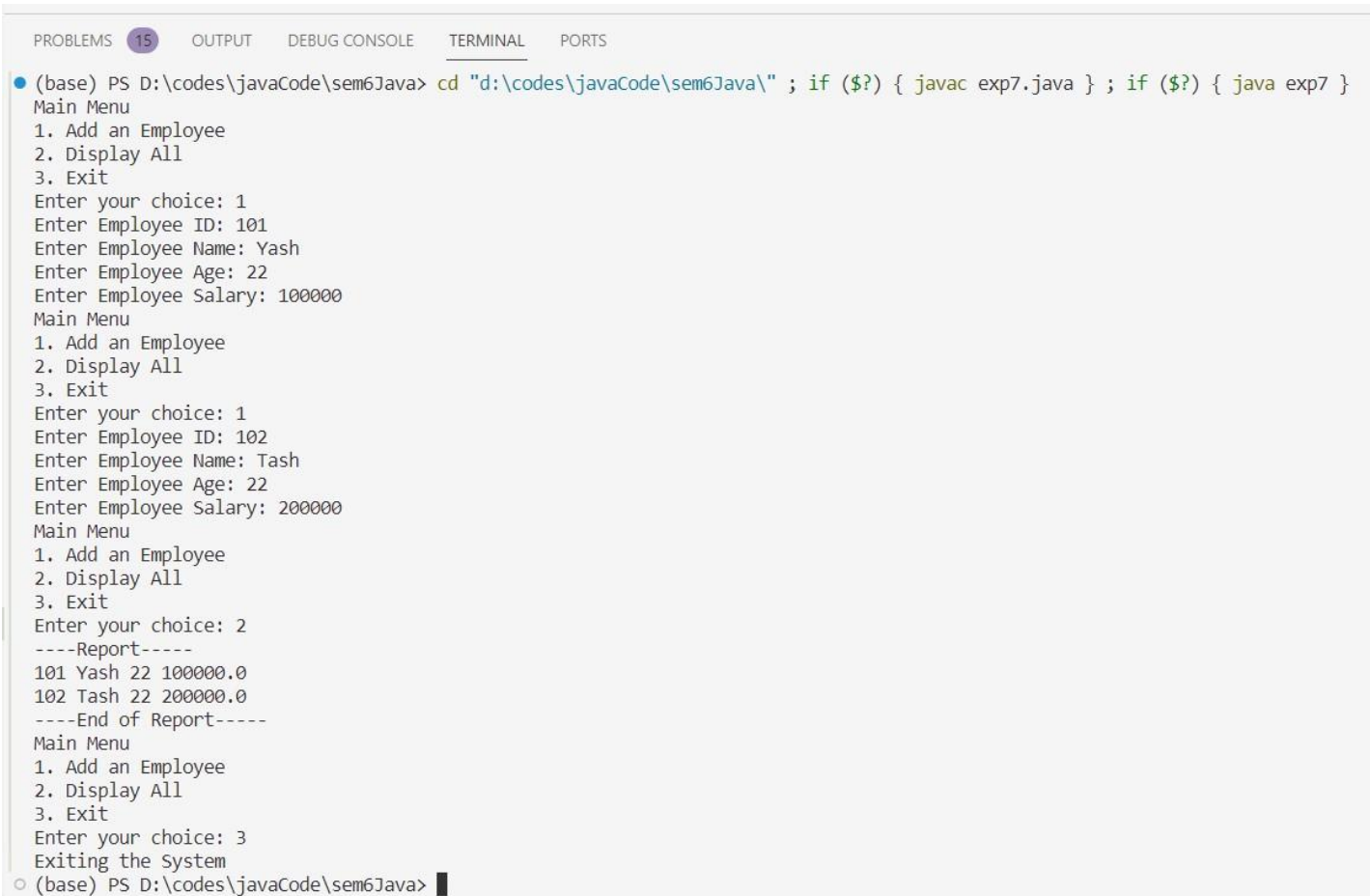
    // Save the employee to file
    try (PrintWriter writer = new PrintWriter(new FileWriter(FILE_NAME, true))) {
        writer.println(employee);
    } catch (IOException e) {
        System.out.println("Error occurred while writing to file: " + e.getMessage());
    }
}

private static void displayAllEmployees(List<Employee> employees) {
    System.out.println("----Report ----");
}
```



```
try (Scanner fileScanner = new Scanner(new File(FILE_NAME))) {  
    while (fileScanner.hasNextLine()) {  
        System.out.println(fileScanner.nextLine());  
    }  
} catch (FileNotFoundException e) {  
    System.out.println("File not found: " + e.getMessage());  
}  
System.out.println("----End of Report-----");  
}  
}
```

4. OUTPUT:



```
(base) PS D:\codes\javaCode\sem6Java> cd "d:\codes\javaCode\sem6Java\" ; if ($?) { javac exp7.java } ; if ($?) { java exp7 }  
Main Menu  
1. Add an Employee  
2. Display All  
3. Exit  
Enter your choice: 1  
Enter Employee ID: 101  
Enter Employee Name: Yash  
Enter Employee Age: 22  
Enter Employee Salary: 100000  
Main Menu  
1. Add an Employee  
2. Display All  
3. Exit  
Enter your choice: 1  
Enter Employee ID: 102  
Enter Employee Name: Tash  
Enter Employee Age: 22  
Enter Employee Salary: 200000  
Main Menu  
1. Add an Employee  
2. Display All  
3. Exit  
Enter your choice: 2  
----Report-----  
101 Yash 22 100000.0  
102 Tash 22 200000.0  
----End of Report-----  
Main Menu  
1. Add an Employee  
2. Display All  
3. Exit  
Enter your choice: 3  
Exiting the System  
(base) PS D:\codes\javaCode\sem6Java>
```

5. Learning Outcomes:

- Learnt about the implementation of classes and objects in java.
- Implemented an StringList using ArrayList in Java.
- Implemented the concepts of OOPS in JAVA.



Experiment No. 3.1

StudentName:Suraj Sagar

UID:21BCS3388

Branch: BE-CSE

Section/Group: FL_602-A

Semester: 6th

Date of Performance: 19/03/24

Subject Name: Project based learning in Java with Lab

Subject Code: 21CSH-319

1. **Aim:** Write a Program for palindrome creator application for making a longest possible palindrome out of given input string.
2. **Objective:** To develop a program which finds the largest palindromic substring out of a given string input by user.

3. Script and Output:

Code:

```
import java.util.Scanner;
```

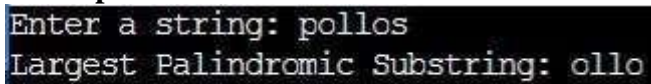
```
public class Main {  
    public static String longestPalindrome(String s) {  
        if (s == null || s.length() < 1) return "";  
  
        int start = 0;  
        int end = 0;  
  
        for (int i = 0; i < s.length(); i++) {  
            int len1 = expandAroundCenter(s, i, i);  
            int len2 = expandAroundCenter(s, i, i + 1);  
            int len = Math.max(len1, len2);  
            if (len > end - start) {  
                start = i - (len - 1) / 2;  
                end = i + len / 2;  
            }  
        }  
        return s.substring(start, end + 1);  
    }  
  
    private static int expandAroundCenter(String s, int left, int right) {  
        while (left >= 0 && right < s.length() && s.charAt(left) == s.charAt(right)) {  
            left--;  
            right++;  
        }  
    }  
}
```

```
        return right - left - 1;
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a string: ");
        String s = scanner.nextLine();
        scanner.close();

        System.out.println("Largest Palindromic Substring: " + longestPalindrome(s));
    }
```

4. Output:



```
Enter a string: pollos
Largest Palindromic Substring: ollo
```

Fig 1. Test Case

5. Learning Outcomes

- 1) Develop algorithms to find the largest palindromic substring, focusing on efficiency.
- 2) Practice Java string manipulation and explore the String class.
- 3) Improve Java programming by using standard libraries and coding practices.



Experiment 3.2

StudentName: Suraj Sagar

UID: 21BCS3388

Section/Group: NTPP-601-A

Date of Performance: 26/03/2024

Subject Name: Project based learning in Java with Lab

Subject Code: 21CSH-319

1. **Aim:** Create JSP application for addition, multiplication and division.

2. **Objective:** The objective of this program is to create an application that uses Java Servlet and takes input from user to add, multiply or divide numbers. The application then returns the output on the interface.

3. Code:

index.jsp:

```
<!DOCTYPE html>
<html>
<head>
  <title>Math Operations</title>
</head>
<body>
  <h2>Math Operations</h2>
  <form action="calculate.jsp" method="post">
    <input type="checkbox" name="operation" value="add" checked>Addition
    <input type="checkbox" name="operation" value="subtract">Subtraction
    <input type="checkbox" name="operation" value="multiply">Multiplication
    <input type="checkbox" name="operation" value="divide">Division<br><br>
    Enter first number: <input type="text" name="num1"><br><br>
    Enter second number: <input type="text" name="num2"><br><br>
    <input type="submit" value="Calculate">
  </form>
</body>
</html>
```

calculate.jsp:

```
<% @ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
    <title>Result</title>
</head>
<body>
    <h2>Result</h2>
    <%
        String operation = request.getParameter("operation");
        double num1 = Double.parseDouble(request.getParameter("num1"));
        double num2 = Double.parseDouble(request.getParameter("num2"));
        double result = 0;
        String errorMessage = null;

        if ("add".equals(operation)) {
            result = num1 + num2;
        } else if ("subtract".equals(operation)) {
            result = num1 - num2;
        } else if ("multiply".equals(operation)) {
            result = num1 * num2;
        } else if ("divide".equals(operation)) {
            if (num2 != 0) {
                result = num1 / num2;
            } else {
                errorMessage = "Error: Cannot divide by zero.";
                request.setAttribute("error", errorMessage);
                request.getRequestDispatcher("error.jsp").forward(request, response);
            }
        }
    %>

    <% if (errorMessage == null) { %>
        <p>Result of <%= operation %>: <%= result %></p>
    <% } %>
</body>
</html>
```

error.jsp:

```
<% @ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
    <title>Error</title>
</head>
<body>
    <h2>Error</h2>
    <p><%= request.getAttribute("error") %></p>
</body>
</html>
```

4. Output:

Math Operations

☒ Addition ☐ Subtraction ☐ Multiplication ☐ Division

Enter first number:

Enter second number:

Result

Result of add: 41.0

5. Learning Outcomes:

- Learned how to use Apache Tomcat to run JSP applications.
- Design and implement a dynamic JSP calculator application.
- Deploy and test the JSP calculator on a web server optimizing performance.