THE COLLECTIONS FRAMEWORK

Exercise 1:

```
import java.util.ArrayList;
import java.util.LinkedList;
import java.util.ListIterator;

public class CollectionExercise {
    public static void main(String[] args) {
        ArrayList<String> arrayList = new ArrayList<>();
        arrayList.add("Alice");
        arrayList.add("Bob");
        arrayList.add("Charlie");
        arrayList.add("David");
        System.out.println("ArrayList traversal in forward direction:");
        ListIterator<String> arrayListIterator = arrayList.listIterator();
        while (arrayListIterator.hasNext()) {
            System.out.println(arrayListIterator.next());
        }
        System.out.println("\nArrayList traversal in reverse direction:");
        while (arrayListIterator.hasPrevious()) {
            System.out.println(arrayListIterator.previous());
        }
        LinkedList<String> linkedList = new LinkedList<>();
        linkedList.add("Emma");
        linkedList.add("Frank");
        linkedList.add("Grace");
        linkedList.add("Henry");
        System.out.println("\nLinkedList traversal using for loop:");
        for (String name : linkedList) {
            System.out.println(name);
        }
        linkedList.removeFirst();
        linkedList.removeLast();
        System.out.println("\nLinkedList after removing first and last
elements:");
        for (String name : linkedList) {
            System.out.println(name);
        }
        linkedList.addFirst("Adam");
        linkedList.addLast("Isabella");
```

```
        System.out.println("\nLinkedList after adding new names at first
and last positions:");
        ListIterator<String> linkedListIterator =
linkedList.listIterator();
        while (linkedListIterator.hasNext()) {
            System.out.println(linkedListIterator.next());
        }
    }
}
```

Output:

```
LinkedList traversal using for loop:
Emma
Frank
Grace
Henry

LinkedList after removing first and last elements:
Frank
Grace

LinkedList after adding new names at first and last positions:
Adam
Frank
Grace
Isabella
```

Exercise 2:

```
import java.util.HashSet;

public class Student {
    private String name;
    private int rollNumber;

    public Student(String name, int rollNumber) {
        this.name = name;
        this.rollNumber = rollNumber;
    }

    public String getName() {
        return name;
    }
```

```
    public void setName(String name) {
        this.name = name;
    }

    public int getRollNumber() {
        return rollNumber;
    }

    public void setRollNumber(int rollNumber) {
        this.rollNumber = rollNumber;
    }

    @Override
    public String toString() {
        return "Student{" +
                "name='" + name + '\'' +
                ", rollNumber=" + rollNumber +
                '}';
    }

    @Override
    public int hashCode() {
        return rollNumber; // Using rollNumber as the hash code for
Student objects
    }

    @Override
    public boolean equals(Object obj) {
        if (this == obj) {
            return true;
        }
        if (obj == null || getClass() != obj.getClass()) {
            return false;
        }
        Student student = (Student) obj;
        return rollNumber == student.rollNumber;
    }

    public static void main(String[] args) {
```

```
        HashSet<Student> studentsSet = new HashSet<>();

        Student student1 = new Student("Alice", 1);
        Student student2 = new Student("Bob", 2);
        Student student3 = new Student("Alice", 1); // Duplicate of
student1

        studentsSet.add(student1);
        studentsSet.add(student2);
        studentsSet.add(student3); // This should not be added due to the
same rollNumber as student1

        System.out.println("Students in the HashSet:");
        for (Student student : studentsSet) {
            System.out.println(student);
        }
    }
}
```

Output:

```
Students in the HashSet:
Student{name='Alice', rollNumber=1}
Student{name='Bob', rollNumber=2}
```

Exercise 3:

```
import java.util.TreeMap;

public class Student {
    private String name;
    private int rollNumber;
    private int totalMarks;

    public Student(String name, int rollNumber, int totalMarks) {
        this.name = name;
        this.rollNumber = rollNumber;
        this.totalMarks = totalMarks;
    }

    public String getName() {
        return name;
```

```
    }

    public void setName(String name) {
        this.name = name;
    }

    public int getRollNumber() {
        return rollNumber;
    }

    public void setRollNumber(int rollNumber) {
        this.rollNumber = rollNumber;
    }

    public int getTotalMarks() {
        return totalMarks;
    }

    public void setTotalMarks(int totalMarks) {
        this.totalMarks = totalMarks;
    }

    public String getGrade() {
        if (totalMarks >= 60) {
            return "A";
        } else if (totalMarks > 40) {
            return "B";
        } else {
            return "C";
        }
    }

    public static void main(String[] args) {
        TreeMap<Integer, String> gradesMap = new TreeMap<>();

        Student student1 = new Student("Alice", 1, 75);
        Student student2 = new Student("Bob", 2, 50);
        Student student3 = new Student("Charlie", 3, 35);

        gradesMap.put(student1.getRollNumber(), student1.getGrade());
```

```
        gradesMap.put(student2.getRollNumber(), student2.getGrade());
        gradesMap.put(student3.getRollNumber(), student3.getGrade());

        System.out.println("Roll Number    Grade");
        for (Integer rollNumber : gradesMap.keySet()) {
            System.out.println(rollNumber + "\t\t\t" +
gradesMap.get(rollNumber));
        }
    }
}
```

Output:

```
Roll Number    Grade
1                    A
2                    B
3                    C
```

DATE/ TIME API

Exercise 1:

```
import java.time.LocalDate;
import java.time.Period;

public class AgeFinder {

    public int getAge(LocalDate dateOfBirth) {
        LocalDate currentDate = LocalDate.now();
        Period period = Period.between(dateOfBirth, currentDate);
        return period.getYears();
    }

    public LocalDate getDateAfterNDays(int noOfDays) {
        return LocalDate.now().plusDays(noOfDays);
    }

    public LocalDate getDateAfterXYearYMonthZDays(int year, int month, int
day) {
        return
LocalDate.now().plusYears(year).plusMonths(month).plusDays(day);
    }
```

```
    public static void main(String[] args) {
        AgeFinder ageFinder = new AgeFinder();

        // Get Age
        LocalDate dateOfBirth = LocalDate.of(1989, 10, 26);
        int age = ageFinder.getAge(dateOfBirth);
        System.out.println("Your age is: " + age + " years");

        // Get Date after N days
        int days = 15;
        LocalDate dateAfterNDays = ageFinder.getDateAfterNDays(days);
        System.out.println("The date after " + days + " days is: " +
dateAfterNDays);

        // Get Date after X years, Y months, and Z days
        int year = 1, month = 2, day = 3;
        LocalDate dateAfterXYZDays =
ageFinder.getDateAfterXYearYMonthZDays(year, month, day);
        System.out.println("The date after " + year + " year, " + month +
" months, and " + day + " days is: " + dateAfterXYZDays);
    }
}
```

Output:

```
Your age is: 34 years
The date after 15 days is: 2023-12-30
The date after 1 year, 2 months, and 3 days is: 2025-02-18
```

Exercise 2:

```
import java.time.LocalDate;
import java.time.format.DateTimeFormatter;
import java.time.temporal.ChronoUnit;

public class EmployeeService {
    public static long calculateLOPs(String startDateLOP, String
endDateLOP) {
        DateTimeFormatter formatter =
DateTimeFormatter.ofPattern("dd/MM/yyyy");
        LocalDate start = LocalDate.parse(startDateLOP, formatter);
```

```
        LocalDate end = LocalDate.parse(endDateLOP, formatter);

        long lops = ChronoUnit.DAYS.between(start, end) + 1; // +1 to
include both start and end dates
        return lops;
    }

    // Other fields and methods
    public static void main(String[] args) {
        String startDateLOP = "01/01/2023";
        String endDateLOP = "15/01/2023";

        long daysBetween = calculateLOPs(startDateLOP, endDateLOP);
        System.out.println("Number of LOPs between dates: " +
daysBetween);
    }
}
```

Output:

```
Number of LOPs between dates: 15
```

Exercise 3:

```
import java.time.LocalDateTime;
import java.time.ZoneId;
import java.time.ZonedDateTime;
import java.time.format.DateTimeFormatter;

public class FlightTimeConverter {

    public static void main(String[] args) {
        // Given flight times in IST
        LocalDateTime departureTime = LocalDateTime.of(2016, 11, 30, 21,
30);
        LocalDateTime arrivalTime = LocalDateTime.of(2016, 12, 1, 6, 0);

        // Customer-selected zone (For example, 'America/New_York' zone)
        String selectedZone = "America/New_York";

        // Convert departure time to the selected zone
```

```
        ZonedDateTime departureZonedDateTime =
departureTime.atZone(ZoneId.of("Asia/Kolkata"))

.withZoneSameInstant(ZoneId.of(selectedZone));

        // Convert arrival time to the selected zone
        ZonedDateTime arrivalZonedDateTime =
arrivalTime.atZone(ZoneId.of("Asia/Kolkata"))

.withZoneSameInstant(ZoneId.of(selectedZone));

        // Formatting the times
        DateTimeFormatter formatter =
DateTimeFormatter.ofPattern("dd/MMM/yyyy HH:mm:ss");

        System.out.println("Departure Time: " +
departureZonedDateTime.format(formatter));
        System.out.println("Arrival Time: " +
arrivalZonedDateTime.format(formatter));
    }
}
```

Output:

```
Departure Time: 30/Nov/2016 11:00:00
Arrival Time: 30/Nov/2016 19:30:00
```