ArrayList :

2. Search an Element ?

Program :

```
package Day8;
import java.util.*;
public class ArrayListSearch_Q2 {
	public static void main(String[] args) {
		// TODO Auto-generated method stub
		ArrayList<Integer> numbers = new ArrayList<>(Arrays.asList(10, 20, 30, 40, 50));
		Scanner sc = new Scanner(System.in);
		System.out.print("Enter number to search: ");
		int num = sc.nextInt();
		if (numbers.contains(num)) {
			System.out.println(num + " found in the list.");
		} else {
			System.out.println(num + " not found.");
		}
	}
}
```

Output :
```
Enter number to search: 20
20 found in the list.
```

3. Remove Specific Element ?

Program :

```
package Day8;
import java.util.ArrayList;
import java.util.Arrays;
public class ArrayListRemoveElement_Q3 {

	public static void main(String[] args) {
		// TODO Auto-generated method stub
		ArrayList<String> fruits = new ArrayList<>(Arrays.asList("Apple", "Banana", "Mango", 
"Grapes", "Orange"));
		System.out.println("Original list: " + fruits);
		fruits.remove("Mango");
		System.out.println("Updated list: " + fruits);
	}
}
```

Output : Original list: [Apple, Banana, Mango, Grapes, Orange]

Updated list: [Apple, Banana, Grapes, Orange]

4. Sort Elements?

Program :

```
package Day8;
import java.util.*;
public class ArrayListSort_Q4 {
	public static void main(String[] args) {
		// TODO Auto-generated method stub
		ArrayList<Integer> numbers = new ArrayList<>(Arrays.asList(50, 20, 90, 10, 30, 70, 
60));
		Collections.sort(numbers);
		System.out.println("Sorted list: " + numbers);
	}
}
```

Output : Sorted list: [10, 20, 30, 50, 60, 70, 90]

5. Reverse the ArrayList ?

Program :

```
package Day8;
import java.util.*;
public class ArrayListReverse_Q5 {
	public static void main(String[] args) {
		ArrayList<Character> chars = new ArrayList<>(Arrays.asList('A', 'B', 'C', 'D', 'E'));
		System.out.println("Original list: " + chars);
		Collections.reverse(chars);
		System.out.println("Reversed list: " + chars);
	}
}
```

Output : Original list: [A, B, C, D, E]

```
Reversed list: [E, D, C, B, A]
```

6. Update an Element ?

Program :

```
package Day8;
import java.util.*;
public class ArrayListUpdate_Q6 {
	public static void main(String[] args) {
		ArrayList<String> subjects = new ArrayList<>(Arrays.asList("Math", "Physics",
"Chemistry", "Biology"));
		System.out.println(subjects);
		int index = subjects.indexOf("Math");
		if (index != -1) {
			subjects.set(index, "Statistics");
		}
		System.out.println(subjects);
	}
}
```

Output : [Math, Physics, Chemistry, Biology]

```
[Statistics, Physics, Chemistry, Biology]
```

7. Remove All Elements ?

Program :

```
package Day8;
import java.util.*;
public class ArrayListClear_Q7 {
	public static void main(String[] args) {
		ArrayList<Integer> list = new ArrayList<>(Arrays.asList(10, 20, 30, 40, 50));
		System.out.println(list);
		list.clear();
		System.out.println(list);
		System.out.println(list.size());
	}
}
```

Output : [10, 20, 30, 40, 50]

```
[]
0
```

8. Iterate using Iterator ?

Program :

```
package Day8;
import java.util.*;
public class ArrayListIterator_Q8 {
	public static void main(String[] args) {
		ArrayList<String> cities = new ArrayList<>(Arrays.asList("Delhi", "Mumbai",
"Chennai", "Kolkata"));
		Iterator<String> iterator = cities.iterator();
		System.out.println("Cities:");
		while (iterator.hasNext()) {
			System.out.println(iterator.next());
		}
	}
}
```

Output : Cities:

```
Delhi
Mumbai
Chennai
Kolkata
```

9. Store Custom Objects ?

Program :

```
package Day8;

import java.util.*;

class Student {

	int id;

	String name;

	double marks;

	Student(int id, String name, double marks) {

		this.id = id;

		this.name = name;

		this.marks = marks;

	}

	public String toString() {

		return id + " - " + name + " - " + marks;

	}

}

public class ArrayListStudent_Q9 {

	public static void main(String[] args) {

		ArrayList<Student> students = new ArrayList<>();

		students.add(new Student(1, "Dev", 85));

		students.add(new Student(2, "Muktha", 92));

		students.add(new Student(3, "Yoga", 78));


		for (Student s : students) {

			System.out.println(s);

		}

	}

}
```

Output :
```
1 - Dev - 85.0

2 - Muktha - 92.0
```

3 - Yoga - 78.0

10. Copy One ArrayList to Another ?

Program :

```
package Day8;

import java.util.*;

public class ArrayListCopy_Q10 {

	public static void main(String[] args) {

		ArrayList<String> original = new ArrayList<>(Arrays.asList("Red", "Green", "Blue"));

		ArrayList<String> copy = new ArrayList<>();

		copy.addAll(original);

		System.out.println("Original: " + original);

		System.out.println("Copied: " + copy);

	}

}
```

Output : Original: [Red, Green, Blue]

Copied: [Red, Green, Blue]

LinkedList :

1. Create and Display a LinkedList ?

Program :

```
package Day8;

import java.util.*;

public class LinkedListDisplay_Q1 {

	public static void main(String[] args) {

		LinkedList<String> colors = new LinkedList<>();

		colors.add("Red");

		colors.add("Blue");

		colors.add("Green");

		colors.add("Yellow");

		colors.add("Pink");
```

```
                    for (String color : colors) {
                              System.out.println(color);
                    }
          }
}
```

Output : Red

Blue

Green

Yellow

Pink

2. Add Elements at First and Last Position ?

Program :

```
package Day8;
import java.util.*;
public class LinkedListAddEnds_Q2 {
          public static void main(String[] args) {
                    LinkedList<Integer> numbers = new LinkedList<>();
                    numbers.addFirst(100);
                    numbers.add(50);
                    numbers.add(75);
                    numbers.addLast(200);
                    System.out.println("LinkedList: " + numbers);
          }
}
```

Output : LinkedList: [100, 50, 75, 200]

3. Insert Element at Specific Position ?

Program :

```
package Day8;
import java.util.*;
```

```
public class LinkedListInsert_Q3 {
	public static void main(String[] args) {
		LinkedList<String> names = new LinkedList<>(Arrays.asList("Dev", "Yoga", "Kiran", 
"Muktha"));
		System.out.println(names);
		names.add(2, "Sai");
		System.out.println(names);
	}
}
```

```
Output : [Dev, Yoga, Kiran, Muktha]
[Dev, Yoga, Sai, Kiran, Muktha]
```

4. Remove Elements ?

Program :

```
package Day8;
import java.util.*;
public class LinkedListRemove_Q4 {
	public static void main(String[] args) {
		LinkedList<String> animals = new LinkedList<>(Arrays.asList("Dog", "Cat", "Lion", 
"Tiger", "Elephant"));
		System.out.println(animals);
		animals.removeFirst();
		System.out.println(animals);
		animals.removeLast();
		System.out.println(animals);
		animals.remove("Lion");
		System.out.println(animals);
	}
}
```

```
Output : [Dog, Cat, Lion, Tiger, Elephant]
[Cat, Lion, Tiger, Elephant]
[Cat, Lion, Tiger]
```

```
[Cat, Tiger]
```

5. Search for an Element ?

Program :

```
package Day8;
import java.util.*;
public class LinkedListSearch_Q5 {
	public static void main(String[] args) {
		LinkedList<String> strings = new LinkedList<>(Arrays.asList("One", "Two", "Three", 
"Four"));
		Scanner sc = new Scanner(System.in);
		System.out.print("Enter string to search: ");
		String input = sc.nextLine();
		if (strings.contains(input)) {
			System.out.println(input + " found in the list.");
		} else {
			System.out.println(input + " not found.");
		}
	}
}
```

Output :
```
Enter string to search: Four
Four found in the list.
```

6. Iterate using ListIterator ?

Program :

```
package Day8;
import java.util.*;
public class LinkedListIterator_Q6 {
	public static void main(String[] args) {
		LinkedList<String> cities = new LinkedList<>(Arrays.asList("Delhi", "Mumbai", 
"Bangalore", "Kolkata"));
```

```
        ListIterator<String> iterator = cities.listIterator();
        System.out.println("Forward traversal:");
        while (iterator.hasNext()) {
            System.out.println(iterator.next());
        }
        System.out.println("Backward traversal:");
        while (iterator.hasPrevious()) {
            System.out.println(iterator.previous());
        }
    }
}
```

Output : Forward traversal:

```
Delhi
Mumbai
Bangalore
Kolkata
Backward traversal:
Kolkata
Bangalore
Mumbai
Delhi
```

7. Sort a LinkedList ?

Program :

```
package Day8;
import java.util.*;
public class LinkedListSort_Q7 {
    public static void main(String[] args) {
        LinkedList<Integer> list = new LinkedList<>(Arrays.asList(50, 10, 40, 20, 30));
        Collections.sort(list);
```

```
		System.out.println(list);
	}
}
```

Output : [10, 20, 30, 40, 50]

8. Convert LinkedList to ArrayList ?

Program :

```
package Day8;
import java.util.*;
public class LinkedListToArrayList_Q8 {
	public static void main(String[] args) {
		LinkedList<String> linkedList = new LinkedList<>(Arrays.asList("Apple", "Banana",
"Mango"));
		ArrayList<String> arrayList = new ArrayList<>(linkedList);
		System.out.println("LinkedList: " + linkedList);
		System.out.println("ArrayList: " + arrayList);
	}
}
```

Output : LinkedList: [Apple, Banana, Mango]

ArrayList: [Apple, Banana, Mango]

9. Store Custom Objects in LinkedList ?

Program :

```
package Day8;
import java.util.*;
class Book {
	int id;
	String title, author;
	Book(int id, String title, String author) {
		this.id = id;
		this.title = title;
```

```
        this.author = author;
    }
    public String toString() {
        return id + " - " + title + " by " + author;
    }
}
public class LinkedListBook_Q9 {
    public static void main(String[] args) {
        LinkedList<Book> books = new LinkedList<>();
        books.add(new Book(1, "Java", "James"));
        books.add(new Book(2, "Python", "Guido"));
        books.add(new Book(3, "C++", "Bjarne"));
        for (Book b : books) {
            System.out.println(b);
        }
    }
}
```

Output : 1 - Java by James

2 - Python by Guido

3 - C++ by Bjarne

10. Clone a LinkedList ?

Program :

```
package Day8;

import java.util.*;

public class LinkedListClone_Q10 {
    public static void main(String[] args) {
        LinkedList<Integer> original = new LinkedList<>(Arrays.asList(10, 20, 30, 40));
        LinkedList<Integer> cloned = (LinkedList<Integer>) original.clone();
```

```
                System.out.println("Original: " + original);
                System.out.println("Cloned: " + cloned);
        }
}
```

Output : Original: [10, 20, 30, 40]

Cloned: [10, 20, 30, 40]

Vector :

1. Create a Vector Of Integeres ?

Program :

```
package Day8;
import java.util.*;
public class VectorInteger_Q1 {
        public static void main(String[] args) {
                Vector<Integer> numbers = new Vector<>();
                numbers.add(10);
                numbers.add(20);
                numbers.add(30);
                numbers.add(40);
                numbers.add(50);
                numbers.insertElementAt(99, 2);
                numbers.removeElementAt(1);
                Enumeration<Integer> e = numbers.elements();
                System.out.println("Vector elements:");
                while (e.hasMoreElements()) {
                        System.out.println(e.nextElement());
                }
        }
}
```

Output : Vector elements:

```
10
99
30
40
50
```

2. Create a Vector Of Strings ?

Program :

```
package Day8;
import java.util.*;
public class VectorString_Q2 {
	public static void main(String[] args) {
		Vector<String> names = new Vector<>();
		names.add("Dev");
		names.add("Yoga");
		names.add("Muktha");
		names.add("Sai");
		String search = "Sai";
		System.out.println(search + " exists: " + names.contains(search));
		int index = names.indexOf("Muktha");
		if (index != -1) {
			names.set(index, "Yoga");
		}
		System.out.println("After replacement: " + names);
		names.clear();
		System.out.println("After clearing: " + names);
	}
}
```

Output :
```
Sai exists: true
After replacement: [Dev, Yoga, Yoga, Sai]
After clearing: []
```

3. Write a Program to Compare Vectors ?

Program :

```
package Day8;

import java.util.*;

public class VectorCompare_Q3 {

	public static void main(String[] args) {

		Vector<String> v1 = new Vector<>(Arrays.asList("Red", "Green", "Blue"));

		Vector<String> v2 = new Vector<>();

		v2.addAll(v1);

		boolean areEqual = v1.equals(v2);

		System.out.println("Vector 1: " + v1);

		System.out.println("Vector 2: " + v2);

		System.out.println("Are equal: " + areEqual);

	}

}
```

Output :
```
Vector 1: [Red, Green, Blue]

Vector 2: [Red, Green, Blue]

Are equal: true
```

4. Write a method to return sum of all elements ?

Program :

```
package Day8;

import java.util.*;

public class VectorSum_Q4 {

	public static int sum(Vector<Integer> vector) {

		int total = 0;

		for (int num : vector) {

			total += num;

		}

		return total;

	}
	public static void main(String[] args) {
		Vector<Integer> numbers = new Vector<>(Arrays.asList(10, 20, 30, 40, 50));
		int result = sum(numbers);
		System.out.println("Sum of elements: " + result);
	}
}
```

Output : Sum of elements: 150

Stack :

1. Create a stack of integers ?

Program :

```
package Day8;
import java.util.*;
public class Stack_Q1 {
	public static void main(String[] args) {
		Stack<Integer> stack = new Stack<>();
		stack.push(10);
		stack.push(20);
		stack.push(30);
		stack.push(40);
		stack.push(50);
		System.out.println(stack.pop());
		System.out.println(stack.peek());
		System.out.println(stack.isEmpty());
		System.out.println(stack);
	}
}
```

```
Output : 50
40
false
```

```
[10, 20, 30, 40]
```

2. Reverse a string using stack ?

Program :

```
package Day8;
import java.util.*;
public class Stack_Q2 {
	public static void main(String[] args) {
		Scanner sc = new Scanner(System.in);
		System.out.print("Enter a string: ");
		String input = sc.nextLine();
		Stack<Character> stack = new Stack<>();
		for (char c : input.toCharArray()) {
			stack.push(c);
		}
		System.out.print("Reversed string: ");
		while (!stack.isEmpty()) {
			System.out.print(stack.pop());
		}
	}
}
```

Output :
```
Enter a string: Deva
Reversed string: aveD
```

3. Use Stack to check for balanced parenthesis ?

Program :

```
package Day8;
import java.util.*;
public class Stack_Q3 {
	public static boolean isBalanced(String expression) {
		Stack<Character> stack = new Stack<>();
		for (char ch : expression.toCharArray()) {
```

```
                if (ch == '(') {
                    stack.push(ch);
                } else if (ch == ')') {
                    if (stack.isEmpty())
                        return false;
                    stack.pop();
                }
            }
            return stack.isEmpty();
        }
        public static void main(String[] args) {
            String expr = "(a+b)*(c-d)";
            System.out.println("Expression: " + expr);
            if (isBalanced(expr)) {
                System.out.println("Valid Expression");
            } else {
                System.out.println("InValid Expression");
            }
        }
}
```

Output : Expression: (a+b)*(c-d)

Valid Expression

4. Convert a decimal to binary using Stack ?

Program :

```
package Day8;
import java.util.*;
public class Stack_Q4 {
        public static void main(String[] args) {
            Scanner sc = new Scanner(System.in);
            System.out.print("Enter decimal number: ");
```

```
                    int decimal = sc.nextInt();
                    Stack<Integer> stack = new Stack<>();
                    int num = decimal;
                    while (num > 0) {
                              stack.push(num % 2);
                              num /= 2;
                    }
                    System.out.print("Binary of " + decimal + ": ");
                    while (!stack.isEmpty()) {
                              System.out.print(stack.pop());
                    }
          }
}
```

Output : Enter decimal number: 18

Binary of 18: 10010

HashSet :

1. Crete a Hashset of strings ?

Program :

```
package Day8;
import java.util.*;
public class HashSetCities_Q1 {
          public static void main(String[] args) {
                    HashSet<String> cities = new HashSet<>();
                    cities.add("Delhi");
                    cities.add("Mumbai");
                    cities.add("Chennai");
                    cities.add("Kolkata");
                    cities.add("Bangalore");
                    boolean addcity = cities.add("Delhi");
                    System.out.println(addcity);
                    Iterator<String> iterator = cities.iterator();
                    while (iterator.hasNext()) {
                              System.out.println(iterator.next());
                    }
          }
}
```

Output : false

```
Delhi
Chennai
Kolkata
Mumbai
Bangalore
```

2. Perform Operations ?

Program :

```
package Day8;
import java.util.*;
public class HashSetOps_Q2 {
          public static void main(String[] args) {
                    HashSet<String> cities = new HashSet<>(Arrays.asList("Pune", "Hyderabad", "Jaipur",
"Bhopal", "Bangalore"));
                    cities.remove("Jaipur");
                    System.out.println(cities.contains("Bhopal"));
                    cities.clear();
                    System.out.println(cities);
          }
}
```

Output : true

```
[]
```

3. Write a method to return max element ?

Program :

```
package Day8;

import java.util.*;

public class HashSetMax_Q3 {

	public static int findMax(HashSet<Integer> set) {

		int max = Integer.MIN_VALUE;

		for (int num : set) {

			if (num > max)

				max = num;

		}

		return max;

	}

	public static void main(String[] args) {

		HashSet<Integer> numbers = new HashSet<>(Arrays.asList(10, 25, 7, 89, 42));

		int max = findMax(numbers);

		System.out.println("Maximum number: " + max);

	}

}
```

Output : Maximum number: 89

LinkedHashSet :

1. Create a LinkedHashSet of Integers ?

Program :

```
package Day8;

import java.util.*;

public class LinkedHashSet_Q1 {

	public static void main(String[] args) {

		LinkedHashSet<Integer> numbers = new LinkedHashSet<>();

		numbers.add(10);

		numbers.add(5);
		numbers.add(20);
		numbers.add(15);
		numbers.add(5);
		for (int num : numbers) {
			System.out.println(num);
		}
	}
}
```

Output : 10

```
5
20
15
```

2. Create a LinkedHashSet of Custom Objects ?

Program :

```
package Day8;
import java.util.*;
class Student1 {
	int id;
	String name;
	Student1(int id, String name) {
		this.id = id;
		this.name = name;
	}
	public boolean equals(Object o) {
		if (this == o)
			return true;
		if (!(o instanceof Student1))
			return false;
		Student1 s = (Student1) o;
		return id == s.id && name.equals(s.name);
	}
	public int hashCode() {
		return Objects.hash(id, name);
	}
	public String toString() {
		return id + " - " + name;
	}
}
public class LinkedHashSet_Q2 {
	public static void main(String[] args) {
		LinkedHashSet<Student1> students = new LinkedHashSet<>();
		students.add(new Student1(1, "Dev"));
		students.add(new Student1(2, "Yoga"));
		students.add(new Student1(3, "Muktha"));
		students.add(new Student1(2, "Dev"));
		for (Student1 s : students) {
			System.out.println(s);
		}
	}
}
```

```
Output : 1 - Dev
2 - Yoga
3 - Muktha
2 - Dev
```

3. Write a program to merge two LinkedHashSets ?

Program :

```
package Day8;
import java.util.*;
public class LinkedHashSet_Q3 {
	public static void main(String[] args) {
		LinkedHashSet<String> set1 = new LinkedHashSet<>(Arrays.asList("A", "B", "C"));
		LinkedHashSet<String> set2 = new LinkedHashSet<>(Arrays.asList("C", "D", "E"));
		set1.addAll(set2);
		System.out.println("Merged LinkedHashSet:");
		for (String item : set1) {
			System.out.println(item);
		}
	}
}
```

Output : Merged LinkedHashSet:

```
A
B
C
D
E
```

TreeSet :

1. Create a TreeSet of Strings ?

Program :

```
package Day8;
import java.util.*;
public class TreeSet_Q1 {
	public static void main(String[] args) {
		TreeSet<String> countries = new TreeSet<>();
		countries.add("India");
		countries.add("USA");
		countries.add("Germany");
		countries.add("Japan");
		countries.add("Brazil");
		for (String country : countries) {
			System.out.println(country);
```

```
                        }
            }
}
```

Output : Brazil

```
Germany
India
Japan
USA
```

2. Create a TreeSet of Integers ?

Program :

```
package Day8;
import java.util.*;
public class TreeSet_Q2 {
            public static void main(String[] args) {
                        TreeSet<Integer> numbers = new TreeSet<>(Arrays.asList(10, 30, 20, 50, 40));
                        System.out.println("TreeSet: " + numbers);
                        System.out.println("First element: " + numbers.first());
                        System.out.println("Last element: " + numbers.last());
                        int num = 30;
                        System.out.println("Lower than " + num + ": " + numbers.lower(num));
                        System.out.println("Higher than " + num + ": " + numbers.higher(num));
            }
}
```

Output : TreeSet: [10, 20, 30, 40, 50]

```
First element: 10
Last element: 50
Lower than 30: 20
Higher than 30: 40
```

3. Create a TreeSet with a custom comparator ?

Program :

```
package Day8;

import java.util.*;

public class TreeSet_Q3 {

	public static void main(String[] args) {

		TreeSet<String> names = new TreeSet<>(Collections.reverseOrder());

		names.add("Dev");

		names.add("Muktha");

		names.add("Yoga");

		names.add("Sai");

		names.add("Dhala");

		for (String name : names) {

			System.out.println(name);

		}

	}

}
```

```
Output : Yoga

Sai

Muktha

Dhala

Dev
```

Queue :

1. Bank Queue Simutaltion ?

Program :

```
package Day8;

import java.util.*;

public class Queue_Q1 {

	public static void main(String[] args) {

		Queue<String> queue = new LinkedList<>();
```

```
                    queue.add("Customer1");
                    queue.add("Customer2");
                    queue.add("Customer3");
                    queue.add("Customer4");
                    queue.add("Customer5");
                    System.out.println("Bank Queue: " + queue);
                    while (!queue.isEmpty()) {
                              System.out.println("Serving: " + queue.poll());
                              System.out.println("Remaining Queue: " + queue);
                    }
          }
}
```

Output :

```
Bank Queue: [Customer1, Customer2, Customer3, Customer4, Customer5]
Serving: Customer1
Remaining Queue: [Customer2, Customer3, Customer4, Customer5]
Serving: Customer2
Remaining Queue: [Customer3, Customer4, Customer5]
Serving: Customer3
Remaining Queue: [Customer4, Customer5]
Serving: Customer4
Remaining Queue: [Customer5]
Serving: Customer5
Remaining Queue: []
```

2. Task Manager ?

Program :

```
package Day8;
import java.util.*;
public class Queue_Q2 {
          public static void main(String[] args) {
                    Queue<String> tasks = new LinkedList<>();
```

```
		tasks.offer("Write report");
		tasks.offer("Check email");
		tasks.offer("Attend meeting");
		System.out.println("Next task: " + tasks.peek());
		System.out.println("Completed: " + tasks.poll());
		System.out.println("Remaining tasks: " + tasks);
	}
}
```

Output : Next task: Write report

```
Completed: Write report
Remaining tasks: [Check email, Attend meeting]
```

3. Write a method to return list of even numbers ?

Program :

```
package Day8;
import java.util.*;
public class Queue_Q3 {
	public static List<Integer> getEvenNumbers(Queue<Integer> queue) {
		List<Integer> evens = new ArrayList<>();
		for (int num : queue) {
			if (num % 2 == 0) {
				evens.add(num);
			}
		}
		return evens;
	}
	public static void main(String[] args) {
		Queue<Integer> numbers = new LinkedList<>(Arrays.asList(1, 2, 3, 4, 5, 6, 7, 8));
		List<Integer> evenList = getEvenNumbers(numbers);
		System.out.println("Original: " + numbers);
		System.out.println("Even numbers: " + evenList);
```

```
        }
}
```

Output : Original: [1, 2, 3, 4, 5, 6, 7, 8]

Even numbers: [2, 4, 6, 8]

PriorityQueue :

1. Hosiptal Emergency Queue ?

Program :

```
package Day8;
import java.util.*;
class Patient {
        String name;
        int severityLevel;
        Patient(String name, int severityLevel) {
                this.name = name;
                this.severityLevel = severityLevel;
        }
        public String toString() {
                return name + " (Severity: " + severityLevel + ")";
        }
}

public class PriorityQueue_Q1 {
        public static void main(String[] args) {
                PriorityQueue<Patient> queue = new PriorityQueue<>(
                                (p1, p2) -> Integer.compare(p2.severityLevel, p1.severityLevel));
                queue.add(new Patient("Dev", 2));
                queue.add(new Patient("Muktha", 5));
                queue.add(new Patient("Yoga", 3));
                while (!queue.isEmpty()) {
                        System.out.println("Treating: " + queue.poll());
        }
    }
}
```

```
Output : Treating: Muktha (Severity: 5)
Treating: Yoga (Severity: 3)
Treating: Dev (Severity: 2)
```

2. Print jobs Priority ?

Program :

```
package Day8;
import java.util.*;
class PrintJob {
    String document;
    int priority;
    PrintJob(String document, int priority) {
        this.document = document;
        this.priority = priority;
    }
    public String toString() {
        return document + " (Priority: " + priority + ")";
    }
}
public class PriorityQueue_Q2 {
    public static void main(String[] args) {
        PriorityQueue<PrintJob> jobQueue = new
        PriorityQueue<>(Comparator.comparingInt(job -> -job.priority));
        jobQueue.offer(new PrintJob("Resume.pdf", 3));
        jobQueue.offer(new PrintJob("Invoice.docx", 5));
        jobQueue.offer(new PrintJob("Poster.jpg", 2));
        while (!jobQueue.isEmpty()) {
            System.out.println("Printing: " + jobQueue.poll());
```

```
                }
        }
}
```

Output : Printing: Invoice.docx (Priority: 5)

Printing: Resume.pdf (Priority: 3)

Printing: Poster.jpg (Priority: 2)

3. Write a method to merge two Priority Queues and return sorted merged queue ?

Program :

```
package Day8;
import java.util.*;
public class PriorityQueue_Q3 {
        public static PriorityQueue<Integer> mergeQueues(PriorityQueue<Integer> q1, 
PriorityQueue<Integer> q2) {
                PriorityQueue<Integer> merged = new PriorityQueue<>(q1);
                merged.addAll(q2);
                return merged;
        }
        public static void main(String[] args) {
                PriorityQueue<Integer> queue1 = new PriorityQueue<>(Arrays.asList(1, 4, 6));
                PriorityQueue<Integer> queue2 = new PriorityQueue<>(Arrays.asList(2, 3, 5));
                PriorityQueue<Integer> result = mergeQueues(queue1, queue2);
                while (!result.isEmpty()) {
                        System.out.print(result.poll() + " ");
                }
        }
}
```

Output : 1 2 3 4 5 6

Deque :

1. Palindrome Checker ?

Program :

```
package Day8;

import java.util.*;

public class Deque_Q1 {

        public static boolean isPalindrome(String input) {

                Deque<Character> deque = new ArrayDeque<>();

                for (char c : input.toCharArray()) {

                        deque.addLast(c);

                }

                while (deque.size() > 1) {

                        if (deque.removeFirst() != deque.removeLast()) {

                                return false;

                        }

                }

                return true;

        }

        public static void main(String[] args) {

                Scanner sc = new Scanner(System.in);

                System.out.print("Enter a string: ");

                String word = sc.nextLine();

                if (isPalindrome(word)) {

                        System.out.println(word + " is a palindrome");

                } else {

                        System.out.println(word + " is not a palindrome");

                }

        }

}
```

Output : Enter a string: mom

mom is a palindrome

2. Double-ended Order System ?

Program :

```
package Day8;

import java.util.*;

public class Deque_Q2 {

	public static void main(String[] args) {

		Deque<String> orders = new ArrayDeque<>();

		orders.addFirst("Dev");

		orders.addLast("Muktha");

		orders.addFirst("Yoga");

		System.out.println(orders);

		orders.removeFirst();

		System.out.println(orders);

		orders.removeLast();

		System.out.println(orders);

	}

}
```

Output :

```
[Yoga, Dev, Muktha]

[Dev, Muktha]

[Dev]
```

3. Browser History Simulation ?

Program :

```
package Day8;

import java.util.*;

public class Deque_Q3 {

	public static void main(String[] args) {

		Deque<String> backStack = new ArrayDeque<>();

		Deque<String> forwardStack = new ArrayDeque<>();

		String currentPage = "Home";

		backStack.push(currentPage);
```

```
                    currentPage = "Page1";
                    backStack.push(currentPage);
                    currentPage = "Page2";
                    backStack.push(currentPage);
                    forwardStack.push(backStack.pop());
                    currentPage = backStack.peek();
                    System.out.println("After going back, current page: " + currentPage);
                    if (!forwardStack.isEmpty()) {
                              currentPage = forwardStack.pop();
                              backStack.push(currentPage);
                    }
                    System.out.println("After going forward, current page: " + currentPage);
          }
}
```

Output : After going back, current page: Page1

```
After going forward, current page: Page2
```