Java Functional Interface & Date API Solutions (Q1 - Q10)

Q1. String Length Using Functional Interface

```
@FunctionalInterface
interface StringProcessor {
    int process(String str);
}

public class Q1 {
    public static void main(String[] args) {
        StringProcessor sp = str -> str.length();
        System.out.println("Length: " + sp.process("Hello World"));
    }
}
// Output: Length: 11
```

Q2. Calculator Functional Interface

```
@FunctionalInterface
interface Calculator {
    double operation(double a, double b);
}

public class Q2 {
    public static void main(String[] args) {
        Calculator add = (a, b) -> a + b;
        Calculator sub = (a, b) -> a - b;
        Calculator mul = (a, b) -> a * b;
        Calculator div = (a, b) -> a / b;

        System.out.println("Add: " + add.operation(5, 3));
        System.out.println("Sub: " + sub.operation(5, 3));
        System.out.println("Mul: " + mul.operation(5, 3));
        System.out.println("Div: " + div.operation(6, 3));
    }
}
// Output: Add: 8.0, Sub: 2.0, Mul: 15.0, Div: 2.0
```

Q3. Sort Strings by Length Descending

```
import java.util.*;

public class Q3 {
    public static void main(String[] args) {
        List<String> list = Arrays.asList("Java", "Python", "C", "JavaScript");
        list.sort((s1, s2) -> Integer.compare(s2.length(), s1.length()));
        System.out.println(list);
    }
}
// Output: [JavaScript, Python, Java, C]
```

Q4. Shape Area with Default Method

```
@FunctionalInterface
interface Shape {
    double area();
    default void printArea() {
        System.out.println("Area: " + area());
    }
}

public class Q4 {
    public static void main(String[] args) {
        Shape circle = () -> Math.PI * 3 * 3;
        Shape square = () -> 4 * 4;
        Shape rectangle = () -> 4 * 6;

        circle.printArea();
        square.printArea();
        rectangle.printArea();
    }
}
// Output: Areas of circle, square and rectangle.
```

Q5. Function Returning Function (Square)

```
import java.util.function.Function;

public class Q5 {
    public static Function<Integer, Integer> getSquareFunction() {
        return x -> x * x;
    }

    public static void main(String[] args) {
        Function<Integer, Integer> square = getSquareFunction();
        System.out.println("Square: " + square.apply(5));
    }
}
// Output: Square: 25
```

Q6. Factorial Using Lambda and Function

```
import java.util.function.Function;

public class Q6 {
    public static Function<Integer, Integer> factorial() {
        return n -> {
            int result = 1;
            for (int i = 2; i <= n; i++)
                result *= i;
            return result;
        };
    }
```

```
    public static void main(String[] args) {
        Function<Integer, Integer> fact = factorial();
        System.out.println("Factorial: " + fact.apply(5));
    }
}
// Output: Factorial: 120
```

Q7. Java Date API - Retrieve, Add, Format

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

public class Q7 {
    public static void main(String[] args) {
        LocalDate now = LocalDate.now();
        LocalDate twoWeeksLater = now.plusWeeks(2);
        DateTimeFormatter formatter = DateTimeFormatter.ofPattern("MM/dd/yyyy");

        System.out.println("Current: " + now);
        System.out.println("2 Weeks Later: " + twoWeeksLater);
        System.out.println("Formatted: " + now.format(formatter));
    }
}
// Output: Shows dates in various formats
```

Q8. Format LocalDateTime using DateTimeFormatter

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

public class Q8 {
    public static void main(String[] args) {
        LocalDateTime dt = LocalDateTime.of(2025, 5, 2, 14, 30);
        DateTimeFormatter formatter = DateTimeFormatter.ofPattern("yyyy-MM-dd HH:mm:ss");
        System.out.println("Formatted: " + dt.format(formatter));
    }
}
// Output: Formatted: 2025-05-02 14:30:00
```

Q9. Simple Interest Using Dates

```
import java.time.LocalDate;
import java.time.temporal.ChronoUnit;
import java.util.Scanner;

public class Q9 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter start date (YYYY-MM-DD): ");
        LocalDate start = LocalDate.parse(sc.nextLine());
        System.out.print("Enter initial amount: ");
        double amount = sc.nextDouble();
```

```
        LocalDate now = LocalDate.now();
        long days = ChronoUnit.DAYS.between(start, now);
        double interest = (amount * 8 * days) / (100 * 365);
        System.out.println("Total Amount: " + (amount + interest));
    }
}
// Output: Calculates total amount with interest.
```

Q10. Days Between Two Dates

```
import java.time.LocalDate;
import java.time.temporal.ChronoUnit;
import java.util.Scanner;

public class Q10 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter start date (YYYY-MM-DD): ");
        LocalDate start = LocalDate.parse(sc.nextLine());
        System.out.print("Enter end date (YYYY-MM-DD): ");
        LocalDate end = LocalDate.parse(sc.nextLine());

        long days = ChronoUnit.DAYS.between(start, end);
        System.out.println("Days Between: " + days);
    }
}
// Output: Days Between: 121 (example)
```