

## 1. Implement following functional interfaces from java.util.function using lambdas:

1. Consumer
2. Supplier
3. Predicate
4. Function

The screenshot shows an IDE with a project named 'Java8feature2'. The 'src' directory contains 'FunctionalInterface.java' and 'Main.java'. The 'FunctionalInterface.java' file contains the following code:

```
1 import java.util.function.*;
2
3 public class FunctionalInterface {
4     public static void main(String[] args) {
5         Consumer<String> greet = String name -> System.out.println("Hello, " + name);
6         greet.accept("Aradhana");
7
8         Supplier<Double> randomValue = () -> Math.random();
9         System.out.println("Random value: " + randomValue.get());
10
11         Predicate<Integer> isEven = Integer number -> number % 2 == 0;
12         System.out.println("Is 10 even? " + isEven.test(10));
13
14         Function<String, Integer> stringLength = String str -> str.length();
15         System.out.println("Length of 'Functional': " + stringLength.apply("Functional"));
16     }
17 }
18
19
```

## Output

The screenshot shows the output of the program in the IDE's console:

```
/usr/lib/jvm/java-1.17.0-openjdk-amd64/bin/java -javaagent:/opt/intellij/lib/idea_rt.jar-36379 -Dfile.encoding=UTF-8 -classpath /home/aradhana/IdeaProjects/Java8feature2/out/production/Java8feature2 Fun
Hello, Aradhana
Random value: 0.26904247840258244
Is 10 even? true
Length of 'Functional': 10
Process finished with exit code 0
```

## 2. Create and access default and static method of an interface.

The screenshot shows an IDE with a project named 'Java8feature2'. The 'src' directory contains 'FunctionalInterface.java', 'InterfaceMethod.java', and 'Main.java'. The 'InterfaceMethod.java' file contains the following code:

```
1 interface MyInterface {
2     default void showMessage() {
3         System.out.println("This is a default method");
4     }
5
6     static void displayInfo() {
7         System.out.println("This is a static method");
8     }
9 }
10
11 class MyClass implements MyInterface {
12     public void callDefaultMethod() {
13         showMessage();
14     }
15 }
16
17 public class InterfaceMethod {
18     public static void main(String[] args) {
19         MyClass obj = new MyClass();
20         obj.callDefaultMethod();
21
22         MyInterface.displayInfo();
23     }
24 }
```

Output:

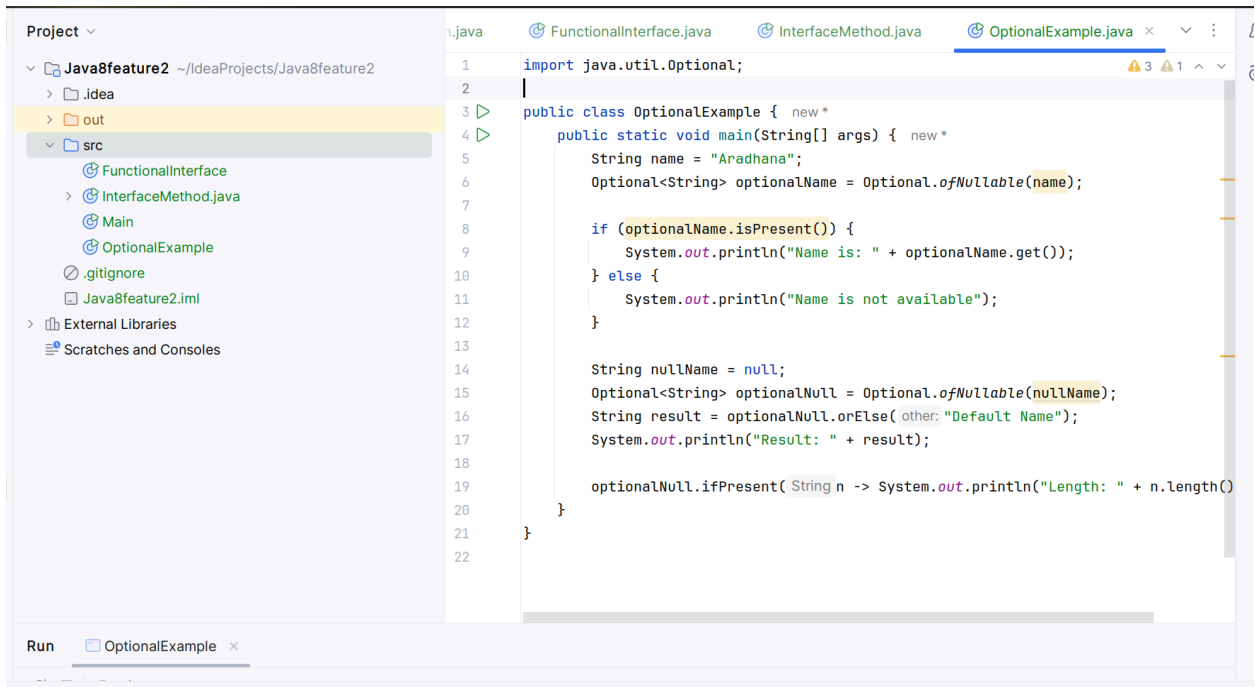
```

/usr/lib/jvm/java-1.17.0-openjdk-amd64/bin/java -javaagent:/opt/intellij/lib/idea_rt.jar=40927 -Dfile.encoding=UTF-8 -classpath /home/ara
This is a default method
This is a static method

Process finished with exit code 0

```

3. Write a program to showcase the use of optional class



```

1  import java.util.Optional;
2
3  public class OptionalExample {
4      public static void main(String[] args) {
5          String name = "Aradhana";
6          Optional<String> optionalName = Optional.ofNullable(name);
7
8          if (optionalName.isPresent()) {
9              System.out.println("Name is: " + optionalName.get());
10         } else {
11             System.out.println("Name is not available");
12         }
13
14         String nullName = null;
15         Optional<String> optionalNull = Optional.ofNullable(nullName);
16         String result = optionalNull.orElse("Default Name");
17         System.out.println("Result: " + result);
18
19         optionalNull.ifPresent(s -> System.out.println("Length: " + s.length()));
20     }
21 }
22

```

Output:

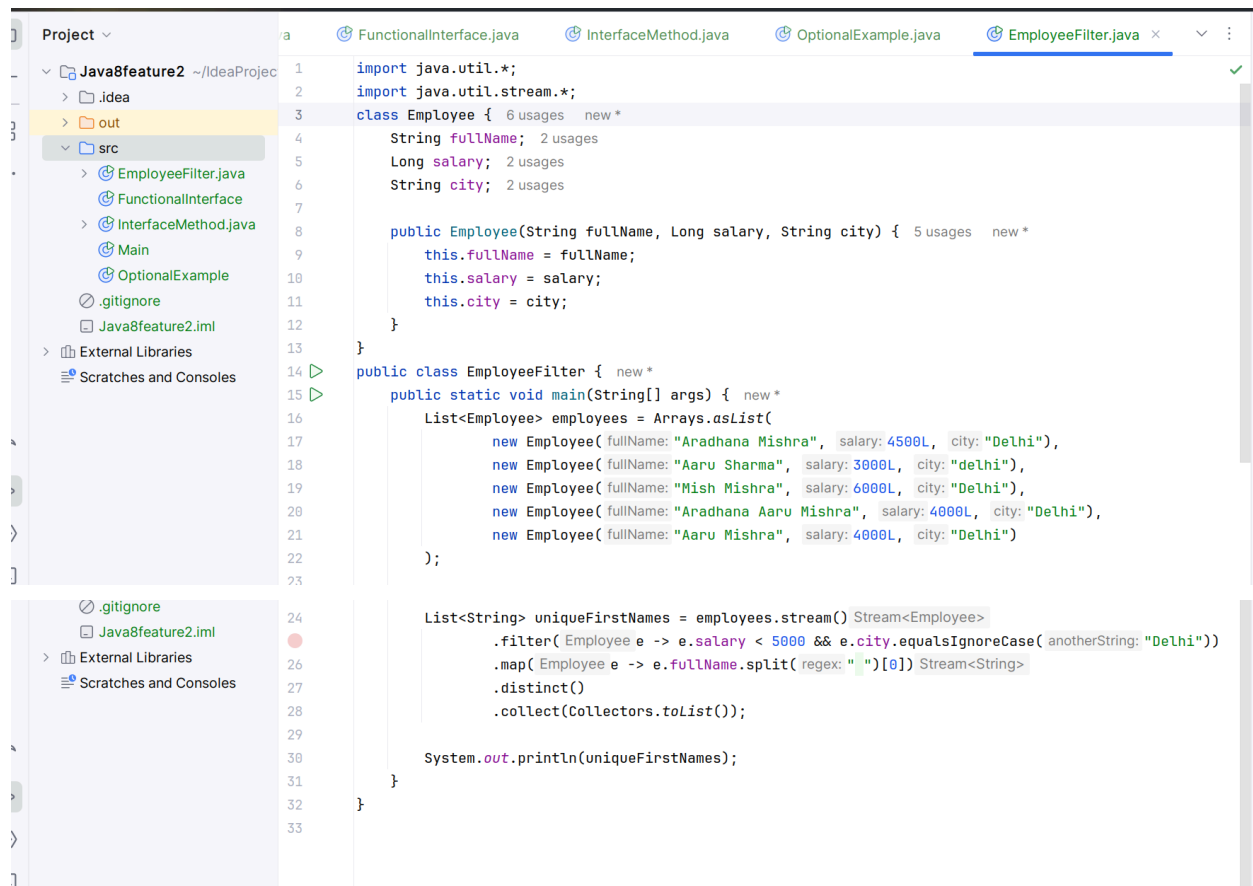
```

/usr/lib/jvm/java-1.17.0-openjdk-amd64/bin/java -javaagent:/opt/intellij/lib/idea_rt.jar=35553 -Dfile.encoding=UTF-8 -classpath /home/ara
Name is: Aradhana
Result: Default Name

Process finished with exit code 0

```

5. Given a list of objects of following class: class Employee{ String fullName; Long salary; String city; } Get list of all unique firstNames of employees where their salary is less than 5000 and who live in delhi. Note: Full name is concatenation of first name, middle name and last name with single space in between



```
1 import java.util.*;
2 import java.util.stream.*;
3 class Employee {
4     String fullName;
5     Long salary;
6     String city;
7
8     public Employee(String fullName, Long salary, String city) {
9         this.fullName = fullName;
10        this.salary = salary;
11        this.city = city;
12    }
13 }
14 public class EmployeeFilter {
15     public static void main(String[] args) {
16         List<Employee> employees = Arrays.asList(
17             new Employee( "Aradhana Mishra", salary: 4500L, city: "Delhi"),
18             new Employee( "Aaru Sharma", salary: 3000L, city: "delhi"),
19             new Employee( "Mish Mishra", salary: 6000L, city: "Delhi"),
20             new Employee( "Aradhana Aaru Mishra", salary: 4000L, city: "Delhi"),
21             new Employee( "Aaru Mishra", salary: 4000L, city: "Delhi")
22         );
23
24         List<String> uniqueFirstNames = employees.stream()
25             .filter( Employee e -> e.salary < 5000 && e.city.equalsIgnoreCase( "Delhi") )
26             .map( Employee e -> e.fullName.split( regex: " ")[0] )
27             .distinct()
28             .collect( Collectors.toList() );
29
30         System.out.println(uniqueFirstNames);
31     }
32 }
33
```

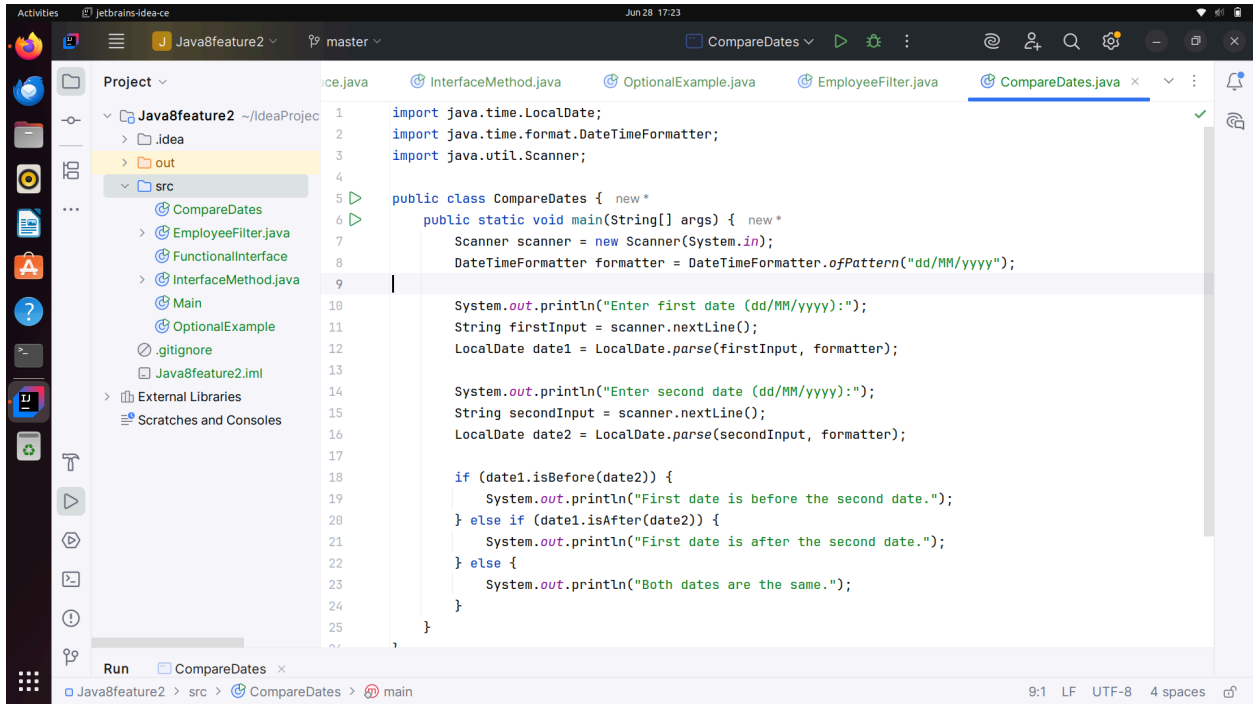
Output:



```
Run EmployeeFilter x
/usr/lib/jvm/java-1.17.0-openjdk-amd64/bin/java -javaagent:/opt/intellij/lib/idea_rt.jar=36801 -Dfile.encoding=UTF-8 -classpath /home/ara
[Aradhana, Aaru]
Process finished with exit code 0
```

## 6. Using java 8 date/time api:

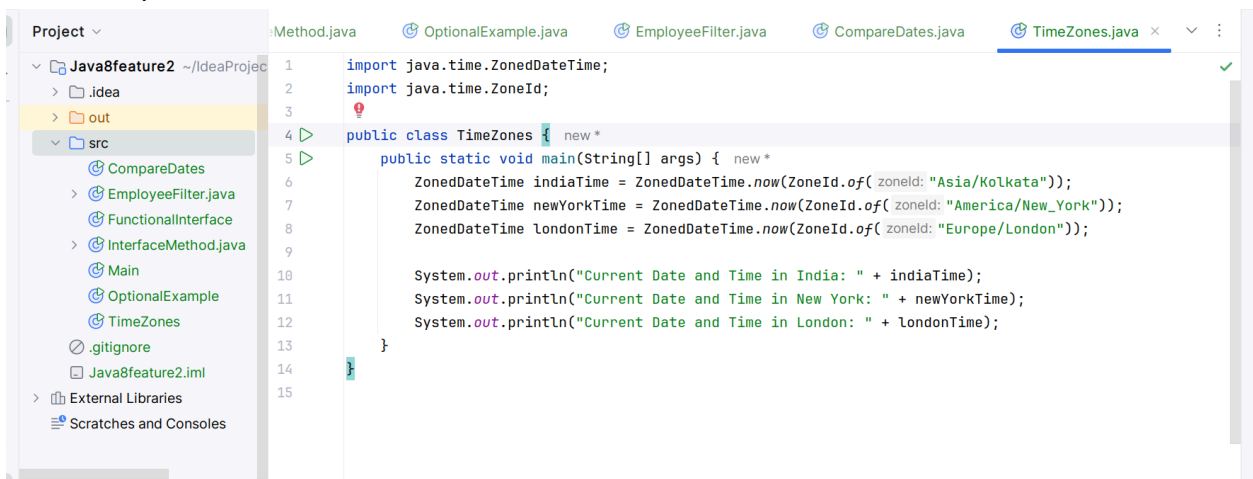
1. WAP to get two dates from user and print if the first date occurs before or after the second date supplied by the user.



Output:



## 2. WAP to print current date and time in 3 different time zones



Output:



A terminal window with a light blue header bar. On the left is a vertical toolbar with icons for back, forward, search, and other navigation functions. The main area contains the following text:

```
/usr/lib/jvm/java-1.17.0-openjdk-amd64/bin/java -javaagent:/opt/intellij/lib/idea_rt.jar=35347 -Dfile.encoding=UTF-8 -classpath /home/ara
Current Date and Time in India: 2025-06-28T17:26:18.968283869+05:30[Asia/Kolkata]
Current Date and Time in New York: 2025-06-28T07:56:18.969396061-04:00[America/New_York]
Current Date and Time in London: 2025-06-28T12:56:18.970887730+01:00[Europe/London]

Process finished with exit code 0
```