Java 8 Features Part-2

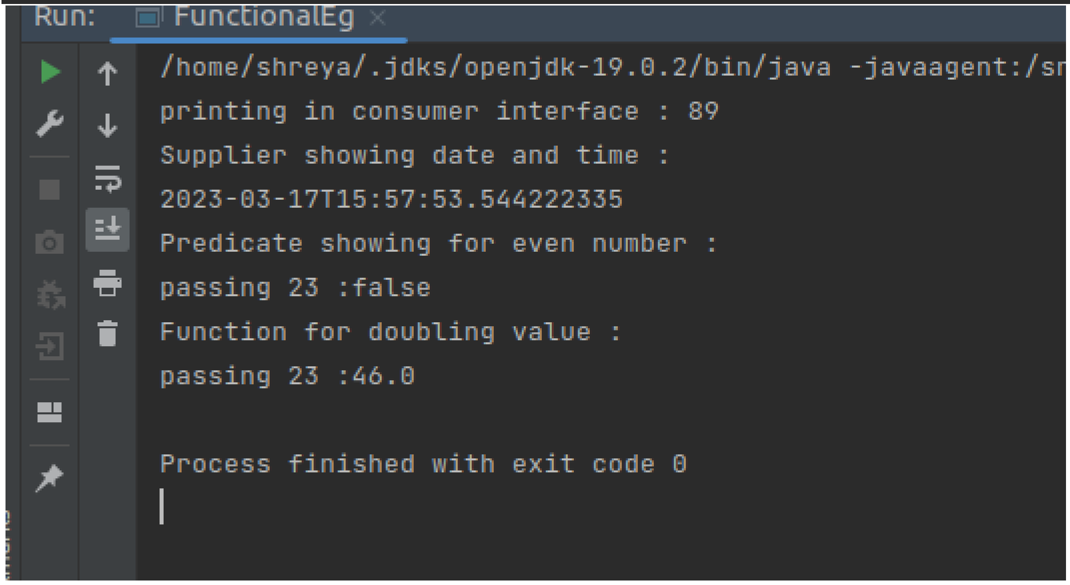
Assignment

Q1. Implement following functional interfaces from java.util.function using lambdas:

- Consumer
- Supplier
- Predicate
- Function

```java
import java.time.LocalDateTime;
import java.util.function.Consumer;
import java.util.function.Function;
import java.util.function.Predicate;
import java.util.function.Supplier;

public class FunctionalEg {
   public static void main(String[] args){
       Consumer<Integer> show=(a)->System.out.println("printing in
consumer interface : "+a);
       show.accept(89);

       Supplier<LocalDateTime> datetime=()->LocalDateTime.now();
       System.out.println("Supplier showing date and time :");
       System.out.println(datetime.get());

       Predicate<Integer> isEven=(a)->(a%2==0);
       System.out.println("Predicate showing for even number :");
       System.out.println("passing 23 :"+isEven.test(23));
```

```
        Function<Integer,Double> DOublevalue=(a)->(a*2.0);

        System.out.println("Function for doubling value :");

        System.out.println("passing 23 :"+DOublevalue.apply(23));

    }

}
```



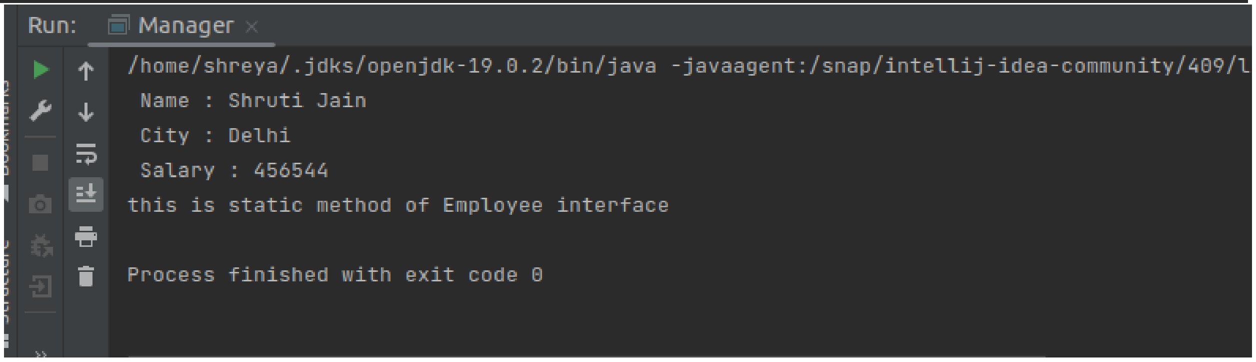
```
/home/shreya/.jdks/openjdk-19.0.2/bin/java -javaagent:/sr
printing in consumer interface : 89
Supplier showing date and time :
2023-03-17T15:57:53.544222335
Predicate showing for even number :
passing 23 :false
Function for doubling value :
passing 23 :46.0

Process finished with exit code 0
```

Q2. Create and access default and static method of an interface.

```
public interface Employ {

   public String getFullname();

   public long getSalary();

   public String getCity();

   default void print(){

       System.out.println(" Name : "+getFullname());

       System.out.println(" City : "+getCity());

       System.out.println(" Salary : "+getSalary());

   }

   static void info(){
```

```java
        System.out.println("this is static method of Employee interface");

    }

}

public class Manager implements Employ{

    String fullname,city;

    long salary;

    public Manager(String fullname,long salary,String city) {

        this.salary = salary;

        this.city=city;

        this.fullname=fullname;

    }


    @Override

    public String getFullname() {

        return fullname;

    }


    @Override

    public long getSalary() {

        return salary;

    }


    @Override

    public String getCity() {

        return city;

    }
```

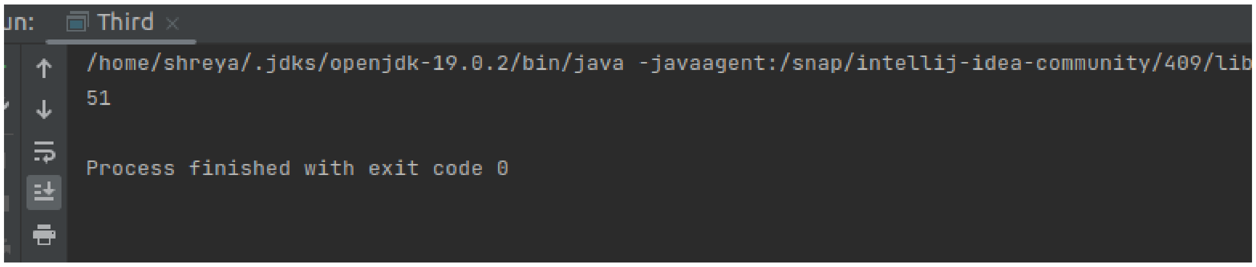
```
   public static void main(String[] args){

       Manager man=new Manager("Shruti Jain",456544,"Delhi");

       man.print();

       Employ.info();

   }

}
```



```
/home/shreya/.jdks/openjdk-19.0.2/bin/java -javaagent:/snap/intellij-idea-community/409/l
 Name : Shruti Jain
 City : Delhi
 Salary : 456544
this is static method of Employee interface

Process finished with exit code 0
```

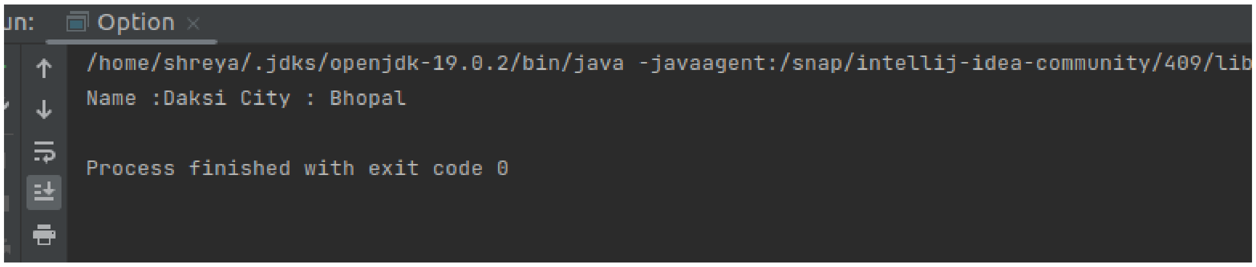
Q3. Sum all the numbers greater than 5 in the integer list using streams

```
import java.util.Arrays;

import java.util.List;


public class Third {

   public static void main(String[] args) {

       List<Integer> lis = Arrays.asList(1, 2, 5, 6, 7, 8, 9, 10, 11);

       Integer NumSum = lis.stream()

               .filter(e -> e > 5)

               .reduce(0, Integer::sum);

       System.out.println(NumSum);

   }

}
```


```
un:   Third
/home/shreya/.jdks/openjdk-19.0.2/bin/java -javaagent:/snap/intellij-idea-community/409/lib
51

Process finished with exit code 0
```

Q4. Write a program to showcase the use of optional class

```
import java.util.Arrays;

import java.util.List;

import java.util.Optional;



public class Option {

   public static void main(String[] args){

       List<Emp> employees= Arrays.asList(

               new Emp("Rishi",4565,"Delhi"),

               new Emp("Shruti",4565,"Delhi"),

               new Emp("Rishabh",6744,"Delhi"),

               new Emp("Daksi",4676,"Bhopal"),

               new Emp("Rishi",3465,"Delhi")

       );

       Optional<Emp> empone=employees.stream()

               .filter(e->e.getCity().equalsIgnoreCase("Bhopal"))

               .findFirst();

       if(empone.isPresent()){

           System.out.println("Name :"+ empone.get().getFullname()+" City :
"+empone.get().getCity());

       }

       else{
           System.out.println("No Employee found for Bhopal");
       }
   }
}
public class Emp {
   String fullname,city;
   long salary;
   public String getFullname(){
       return fullname;
   }
   public long getSalary() {
       return salary;
   }

   public Emp(String fullname,long salary,String city) {
       this.salary = salary;
       this.city=city;
       this.fullname=fullname;
   }

   public String getCity() {
       return city;
   }
}
```



```
un:   Option ×
/home/shreya/.jdks/openjdk-19.0.2/bin/java -javaagent:/snap/intellij-idea-community/409/lib
Name :Daksi City : Bhopal

Process finished with exit code 0
```

Q5. Given a list of objects of following class:

class Employee{

String fullName;

Long salary;

String city;

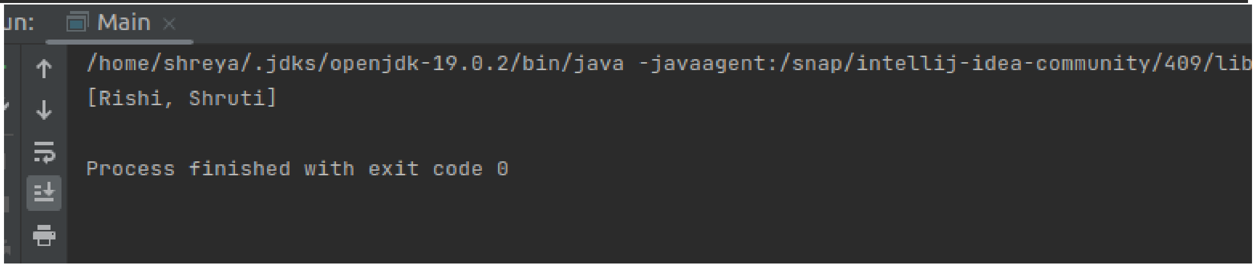
}

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.

```
public class Emp {

   String fullname,city;

   long salary;

   public String getFullname(){

       return fullname;

   }

   public long getSalary() {

       return salary;

   }



   public Emp(String fullname,long salary,String city) {

       this.salary = salary;
```

```java
        this.city=city;

        this.fullname=fullname;

    }


    public String getCity() {

        return city;

    }

}

import java.util.Arrays;

import java.util.List;

import java.util.stream.Collectors;


public class Main {

    public static void main(String[] args) {

        List<Emp> employees= Arrays.asList(

                new Emp("Rishi Raj",4565,"Delhi"),

                new Emp("Shruti Jain",4565,"Delhi"),

                new Emp("Rishabh Jain",6744,"Delhi"),

                new Emp("Daksi Jain",4676,"Bhopal"),

                new Emp("Rishi Singh",3465,"Delhi")

        );

        List<String> UniqueNames=employees.stream()

                .filter(e->e.getSalary()<5000 &&
e.getCity().equalsIgnoreCase("Delhi"))

                .map(e->e.getFullname().split("\\s+")[0])

                .distinct()
                .collect(Collectors.toList());

        System.out.println(UniqueNames);



    }

}
```

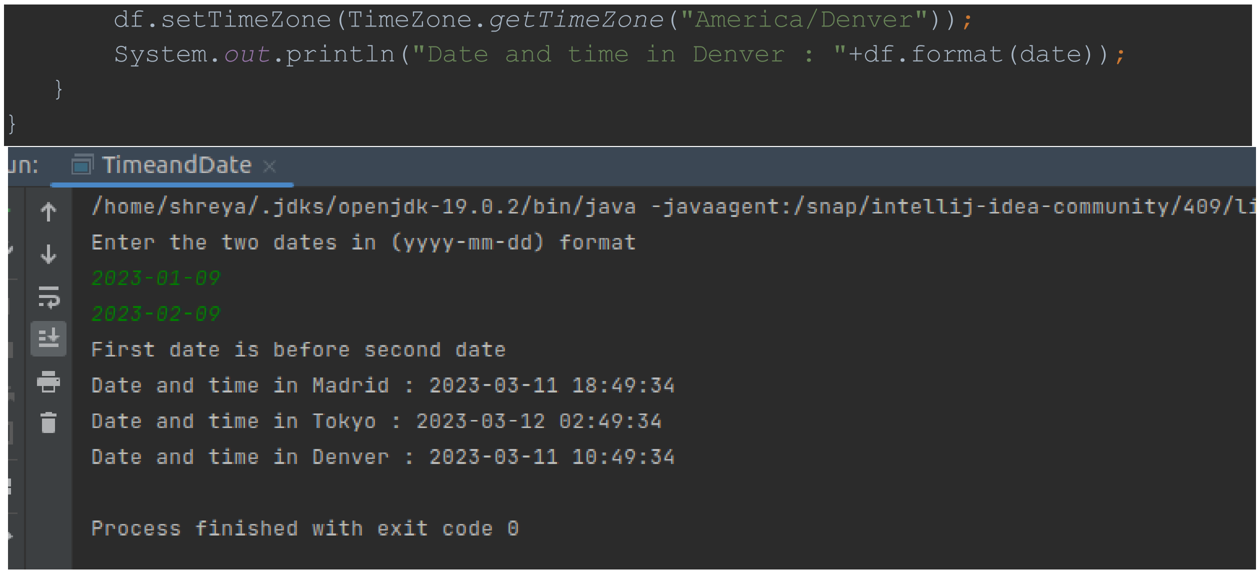


```
/home/shreya/.jdks/openjdk-19.0.2/bin/java -javaagent:/snap/intellij-idea-community/409/lib
[Rishi, Shruti]

Process finished with exit code 0
```

Q6. Using java 8 date/time api:

- WAP to get two dates from user and print if the first date occurs bfore or after the second date supplied by the user.
- WAP to print current date and time in 3 different time zones.

```
import java.text.DateFormat;
import java.text.SimpleDateFormat;
import java.time.*;
import java.util.*;
public class TimeandDate {
   public static void main(String[] args){
       Scanner sc=new Scanner(System.in);
       System.out.println("Enter the two dates in (yyyy-mm-dd) format ");
       String date1=sc.nextLine();
       String date2=sc.nextLine();
       LocalDate firstdate=LocalDate.parse(date1);
       LocalDate seconddate=LocalDate.parse(date2);
       if(firstdate.isAfter(seconddate)){
           System.out.println("First date is after second date");
       }
       else if(firstdate.isBefore(seconddate)){
           System.out.println("First date is before second date");
       }
       else System.out.println("First date is same as second date ");
       Date date=new Date();
       DateFormat df=new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");
       df.setTimeZone(TimeZone.getTimeZone("Europe/Madrid"));
       System.out.println("Date and time in Madrid : "+df.format(date));
       df.setTimeZone(TimeZone.getTimeZone("Asia/Tokyo"));
       System.out.println("Date and time in Tokyo : "+df.format(date));
```



```
        df.setTimeZone(TimeZone.getTimeZone("America/Denver"));
        System.out.println("Date and time in Denver : "+df.format(date));
    }
}
```

un: TimeandDate

```
/home/shreya/.jdks/openjdk-19.0.2/bin/java -javaagent:/snap/intellij-idea-community/409/li
Enter the two dates in (yyyy-mm-dd) format
2023-01-09
2023-02-09
First date is before second date
Date and time in Madrid : 2023-03-11 18:49:34
Date and time in Tokyo : 2023-03-12 02:49:34
Date and time in Denver : 2023-03-11 10:49:34

Process finished with exit code 0
```