

Java 8 Features Part -1 Assignment

Q1) Write the following a functional interface and implement it using lambda:

- To check whether the first number is greater than second number or not, Parameter (int , int) Return type boolean
- Increment the number by 1 and return incremented value Parameter (int) Return int
- Concatenation of 2 string Parameter (String , String) Return (String)
- Convert a string to uppercase and return Parameter (String) Return (String)

```
public class Functional {  
  
    public static void main(String[] args) {  
  
        MyInterface function = (a, b) -> (a < b);  
  
        boolean ans = function.greater(12, 13);  
  
        System.out.println(ans);  
  
        Interf funct=(a) -> (a+1);  
  
        int res= funct.incrementbyone(34);  
  
        System.out.println(res);  
  
        Concatenate func=(a,b)->(a.concat(b));  
  
        String rest = func.Concatena("abc","def");  
  
        System.out.println(rest);  
  
  
        Uppercase fun=(a)->(a.toUpperCase());  
  
        String result=fun.ChangeCase("abcds");  
  
        System.out.println(result);  
  
    }  
}
```

```
@FunctionalInterface
```

```

public interface Uppercase {

    String ChangeCase(String str);

}

@FunctionalInterface

public interface MyInterface {

    boolean greater(int n,int m);

}

@FunctionalInterface

public interface Interf {

    int incrementbyone(int a);

}

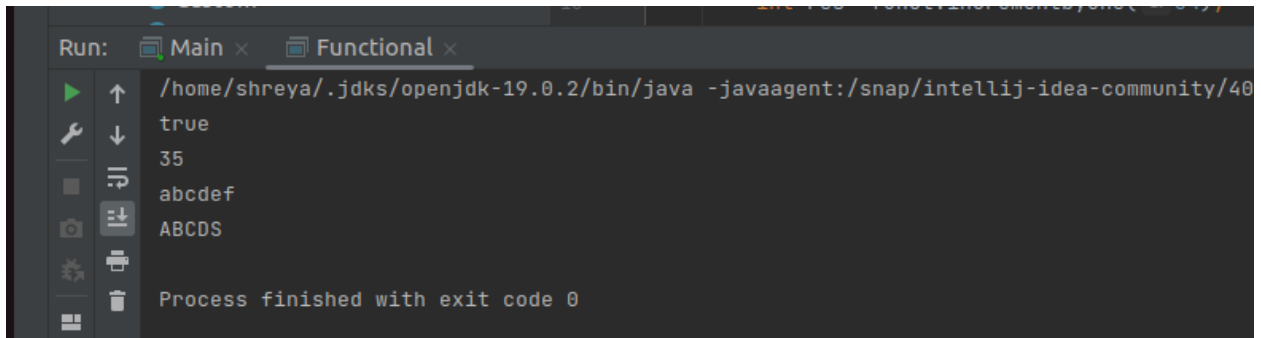
@FunctionalInterface

public interface Concatenate {

    String Concatena(String a,String b);

}

```



```

Run: Main x Functional x
true
35
abcdef
ABCDs
Process finished with exit code 0

```

Q2) Using (instance) Method reference create and apply add and subtract method and using (Static) Method reference create and apply multiplication method for the functional interface created

```

public class Second {

public int adder( int a,int b){

    int res;

```

```

        res=a+b;

        return res;
    }

    public int subtract(int a,int b){

        int rest;

        rest=a-b;

        return rest;
    }

    public static int multiply(int a,int b){

        int result;

        result=a*b;

        return result;
    }

    public static void main(String[] args){

        MyInterface adder= new Second()::adder;

        System.out.println(adder.operate(12,13));

        MyInterface subtract= new Second()::subtract;

        System.out.println(subtract.operate(22,13));

        MyInterface multiply= Second::multiply;

        System.out.println(multiply.operate(8,13));

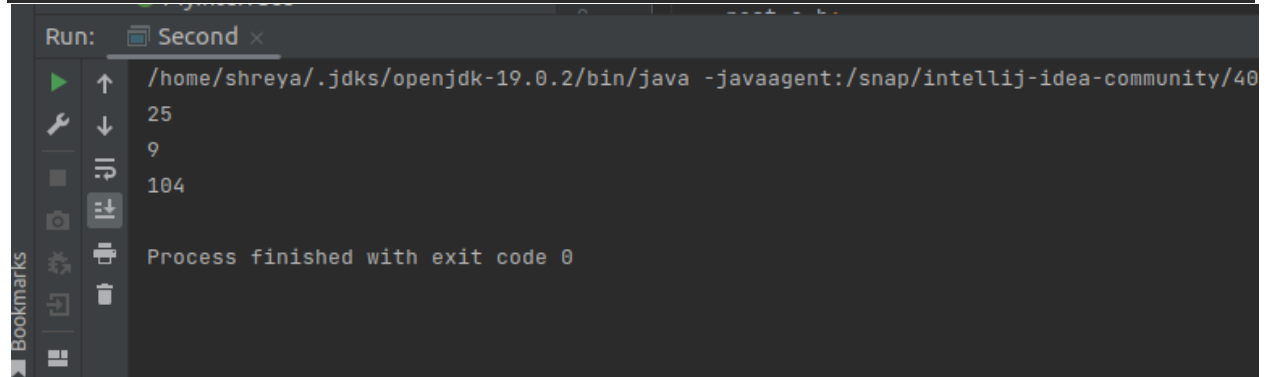
    }

}

@FunctionalInterface

```

```
public interface MyInterface {  
  
    int operate(int a,int b);  
  
}
```



```
Run: Second x  
/home/shreya/.jdk/openjdk-19.0.2/bin/java -javaagent:/snap/intellij-idea-community/40  
25  
9  
104  
Process finished with exit code 0
```

Q3) Implement multiple inheritance with default method inside interface.

```
public interface Parent {  
  
    default void display(){  
  
        System.out.println("this is parent interface");  
  
    }  
  
}  
  
public interface Childone extends Parent{  
  
    default void display(){  
  
        Parent.super.display();  
  
        System.out.println("this is the child one interface");  
  
    }  
  
}  
  
public interface Childtwo extends Parent {  
  
    default void display(){  
  
        Parent.super.display();  
  
        System.out.println("this is the child two interface");  
  
    }  
  
}
```

```

}

public class Childthree implements Childdone, Childtwo{

    public void display(){

        Childdone.super.display();

        Childtwo.super.display();

        System.out.println("this is the child three class.");

    }

    public static void main(String[] args){

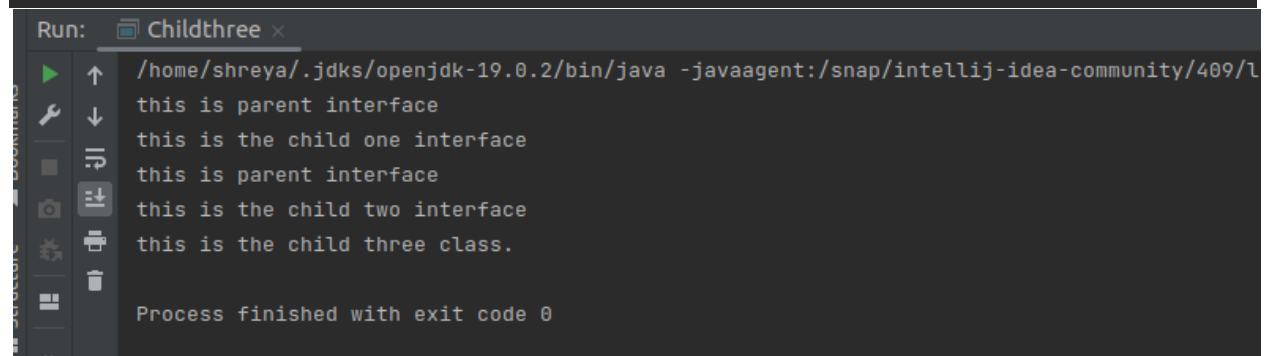
        Childthree three= new Childthree();

        three.display();

    }

}

```



```

Run: Childthree x
/home/shreya/.jdk/openjdk-19.0.2/bin/java -javaagent:/snap/intellij-idea-community/409/L
this is parent interface
this is the child one interface
this is parent interface
this is the child two interface
this is the child three class.
Process finished with exit code 0

```

Q4) Write a program to implement constructor reference

```

@FunctionalInterface

public interface DemoInterface {

    User getUser(String name);

}

public class User {

    String userName;

    public User(String userName){

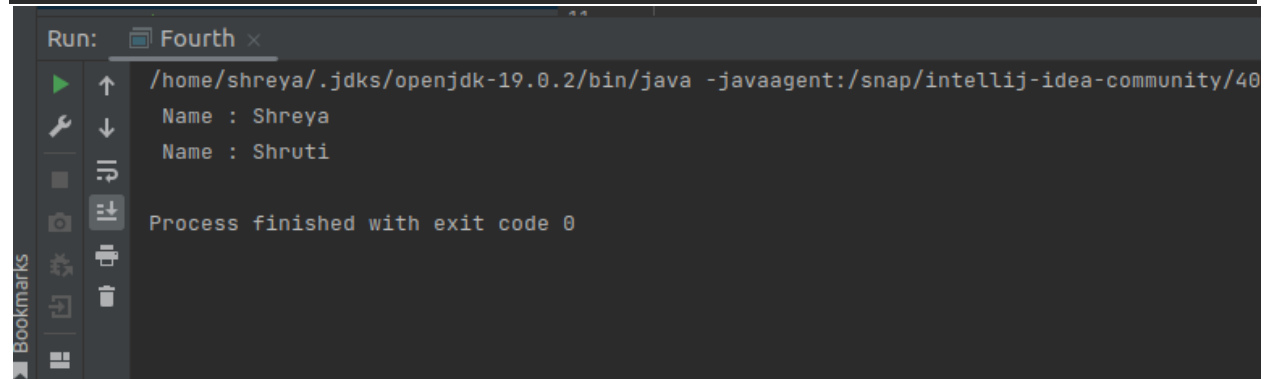
```

```
        this.userName=userName;

    }

}
```

```
public class Fourth {
    public static void main(String[] args){
        DemoInterface demouser=User::new;
        User user=demouser.getUser("Shreya");
        System.out.println(" Name : "+user.userName);
        DemoInterface userinterface=s->new User(s);
        User user1=userinterface.getUser("Shruti");
        System.out.println(" Name : "+user1.userName);
    }
}
```



Run: Fourth x

```
/home/shreya/.jdk/openjdk-19.0.2/bin/java -javaagent:/snap/intellij-idea-community/40
Name : Shreya
Name : Shruti
Process finished with exit code 0
```