- 1. Write the following a functional interface and implement it using lambda:
  - 1. To check whether the first number is greater than second number or not, Parameter (int ,int ) Return type boolean
  - 2. Increment the number by 1 and return incremented value Parameter (int) Return int
  - 3. Concatination of 2 string Parameter (String, String) Return (String)
  - 4. Convert a string to uppercase and return . Parameter (String) Return (String)

```
Main.java
                                                                                         ( LambdaExamples.java
∨ □ Java8feat 1
                                                              @FunctionalInterface 1usage
       | Savasteat | Security | Security
     > 🗀 out
           > @FunctionalInterface
interface Increment {
          > fh External Lit 10 🖫 interface Concatenator {
                                                                       String concatenate(String a, String b); 1usage new*
       Scratches a 11 🔍
                                                                 @FunctionalInterface 1usage new*
                                          14 th interface ToUpperCase {
                                                               String convert(String str); 1usage new*
                                         15 🕰
                                          17 Dublic class LambdaExamples { new*
                                                                 public static void main(String[] args) { new*
                                           18
                                                                               GreaterCheck greater = (int a, int b) -> a > b;
System.out.println(greater.isGreater(a: 10, b: 5));
                                                                               Increment increment = Int a -> a + 1;
System.out.println(increment.incrementByOne( a: 7));
                                                                            Concatenator concat = ( String a, String b) -> a + b;
System.out.println(concat.concatenate( a: "Hello, ", b: "World!"));
                                                                                        ToUpperCase upper = String str -> str.toUpperCase();
                                           29
                                                                                         System.out.println(upper.convert( str: "java"));
                                           30
a8feature1 > src > 🕝 LambdaExamples.java > 🕝 LambdaExamples
                                                                                                                                                                                                                                                                                                                                                                                                                                                                         17:1 LF UTF-8 4 sr
```

## Output

```
($ ■ | ② :

/ /usr/lib/jym/java-1.17.0-openjdk-amd64/bin/java -javaagent:/opt/intellij/lib/idea_rt.jar=40223 -Dfile.encoding=UTF-8 -classpath /home/aradhana/IdeaProjects/Java8fe.true

8

Hello, World!

JAVA

Process finished with exit code 8

#### Process finished with exit code 8
```

 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

```
Project v
                                                   MethodReference.java ×
                           @FunctionalInterface 3 usages new*

√ □ Java8feature1 ~/IdeaProject

                          int apply(int a,int b); 3 usages new*
}
                      2 © interface Operation{
  > 🗀 .idea
                      3 🔍
  > nout
 ∨ 🗀 src
                           class MathOperation{ 2 usages new*
                         > @LambdaExamples.java
     @ Main
    ⊘ .gitignore
   ■ Java8feature1.iml
                                return a-b;
 > fill External Libraries
                            Scratches and Consoles
                     14
                      16
                     17 ▷
                           public class MethodReference {  new*
                      18 ⊳
                             public static void main(String[] args) { new*
                                 MathOperation math=new MathOperation();
                                 Operation addition= math::add;
                      20
                                 System.out.println(addition.apply(a: 20, b: 10));
                                 Operation subtraction=math::sub;
                                 System.out.println(subtraction.apply( a: 20, b: 10));
                     24
                          Operation multiplication =math::multiply;
                                 System.out.println(multiplication.apply( a: 20, b: 10));
                      26
                      28
                      29
24:39 LF UTF-8 4 sp
Output
G . :
    /usr/lib/jvm/java-1.17.0-openjdk-amd64/bin/java -javaagent:/opt/intellij/lib/idea_rt.jar=35681 -Dfile.encoding=UTF-8 -classpath /home/aradhana/IdeaProjects/Java8fe
    30
   10
<del>=</del> 200
=+
Process finished with exit code 0
î
                                                                                                                24:39 LF UTF-8 4 sp
```

3. Implement multiple inheritance with default method inside interface.

```
@ 24 Q ®

■ Java8feature1 ∨

                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             ي

⊕ LambdaExamples.java

                                                                                                                                                                                                                                                                                                                 MultipleInheritance.java ×
                 Project

    Main.java

                                                                                                                                                                                                                                   MethodReference.iava
                  ∨ □ Java8feature1 ~/IdeaProject 1 □ interface A { 2 usages 1 implementation new*
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              @
                                                                                                                                          default void show() { 2 usages 1 override new*
                         idea 🗀 .idea
                                                                                                                                                     System.out.println("Show from A");
吕
                               > @ LambdaExamples.java 5
                                 Main
                                > @ MultipleInheritance.java 9
                                                                                                                                                     System.out.println("Show from B");
                              .gitignore
                              ■ Java8feature1.iml
                  ☐ External Libraries

☐ Scratches and Consoles

13
14 ⓒ 14
                                                                                                                                      public void show() { 4 usages new*
                                                                                                                                                       A.super.show();
                                                                                                                                                         B.super.show():
                                                                                                                                                         System.out.println("Show from MyClass");
                                                                                                        19
20
                                                                                                      Prename usages

21 D public class MultipleInheritance new*

22 D public static void main(String[] args) { new*

MyClass obj = new MyClass();

April - Transfer | April - Transfer |

April - Transfer | April - Transfer |

April - Transfer | April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April - Transfer |

April -
T
\triangleright
Ø
>_
(!)
21:33 LF UTF-8 4 spaces ⊕
```

## Output

4. Write a program to implement constructor reference

```
≣ U Java8feature1 ∨ β maste
                                                                                                                                                                                                                                                                                                                        @ 4 Q $
                                                                                                                                                                                                                       CambdaExamples.java
                                                                                                                                                                               MultipleInheritance.java
                                                                                                                                                                                                                                                                                                                                                                         C
                                                                                                                                         MethodReference.java
                                                                              2 ①, interface StudentCreatorDefault {
3 ②, Student create(); lusage
                                                                          5 PrunctionalInterface 1usage
                                                                                      interface StudentCreatorOneParam {
   Student create(String name); 1usage
                                                                                      }
@FunctionalInterface 1usage
interface StudentCreatorTwoParam {
Student create(String name, int age); 1usage
                              MultipleInheritance.iava
                       ⊘ .gitignore
□ Java8feature1.iml

□ External Libraries
                                                                                       class Student { 9 usages
String name; 4 usage
int age; 5 usages
                         Scratches and Consoles
                                                                                             Student() { lusage
this.name = "Unknown";
this.name = 0;
System.out.println("Default Constructor > Name: " + name + ", Age: " + age);
                                                                                             Student(String name) { lusage
this.name = name;
this.name = name;
this.name = name;
this.name = name + ", Age: " + age);
System.out.println("One-Param Constructor -> Name: " + name + ", Age: " + age);
4
o
                                                                                             Student(String name, int age) { lusage
    this.name * name;
    this.name s name;
    this.age = age;
    System.out.println("Two-Param Constructor + Name: " + name + ", Age: " + age);
                                                                                      public class ConstructorReference {
           \triangleright
                                                                                            public static void main(String[] args) {
StudentTreatorDefault creator1 = Student::new;
Student s1 = creator1.create();
           D
           >_
                                                                                                    StudentCreatorOneParam creator2 = Student::new;
Student s2 = creator2.create("Aradhana");
           (!)
           99
:::
                   Run
                             ConstructorReference
                                                                           e.iava > ① StudentCreatorOneParam
                                                                                                                                                                                                                                                                                                                                  5:1 LF UTF-8 4 spaces m
                                                        38 D public class ConstructorReference {
36 D public static void main(String[] args) {
37 StudentCreatorDefault creator1 = Student::new;
38 Student s1 = creator1.create();
                                                                                StudentCreatorOneParam creator2 = Student::new;
                                                                                Student s2 = creator2.create("Aradhana");
                                                                                  StudentCreatorTwoParam creator3 = Student::new;
Student s3 = creator3.create( name: "Aradhana", age: 22);
```

## Output