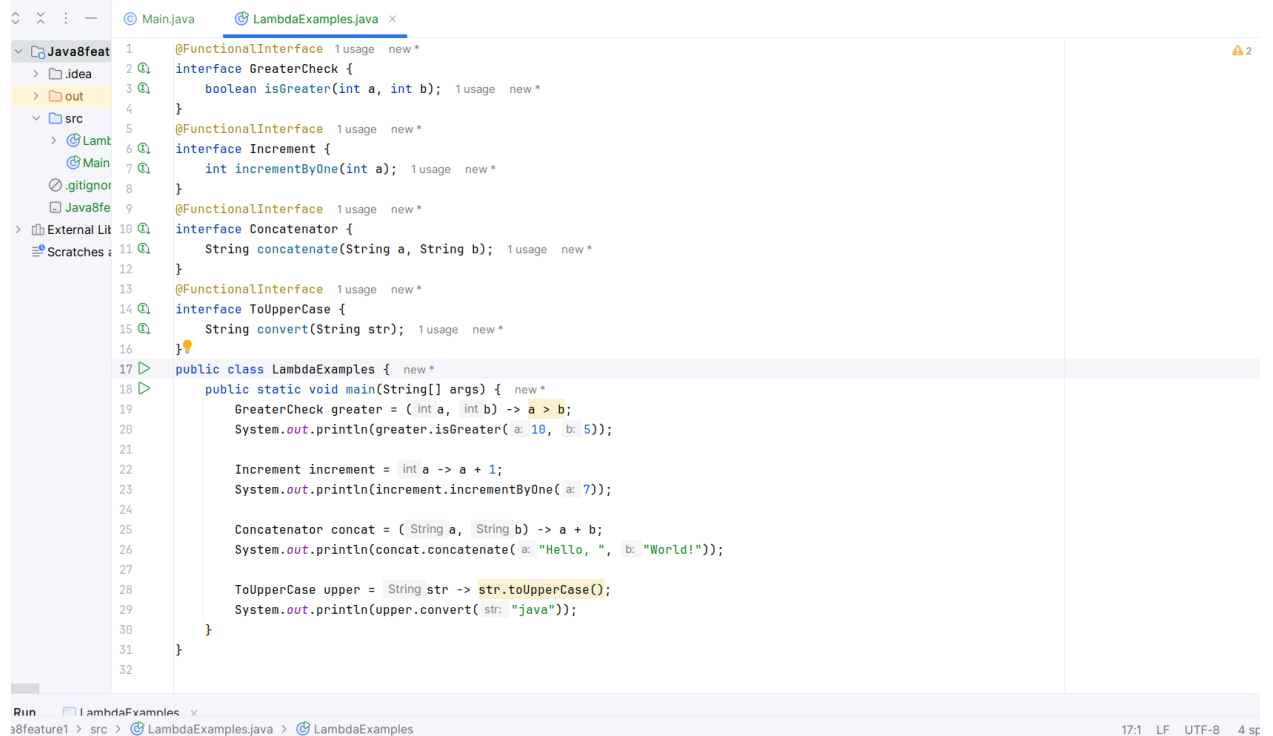


1. Write the following a functional interface and implement it using lambda:
  1. To check whether the first number is greater than second number or not, Parameter (int ,int ) Return type boolean
  2. Increment the number by 1 and return incremented value Parameter (int) Return int
  3. Concatination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



```
1  @FunctionalInterface 1 usage new *
2  interface GreaterCheck {
3      boolean isGreater(int a, int b); 1 usage new *
4  }
5  @FunctionalInterface 1 usage new *
6  interface Increment {
7      int incrementByOne(int a); 1 usage new *
8  }
9  @FunctionalInterface 1 usage new *
10 interface Concatenator {
11     String concatenate(String a, String b); 1 usage new *
12 }
13 @FunctionalInterface 1 usage new *
14 interface ToUpperCase {
15     String convert(String str); 1 usage new *
16 }
17 public class LambdaExamples { new *
18     public static void main(String[] args) { new *
19         GreaterCheck greater = (int a, int b) -> a > b;
20         System.out.println(greater.isGreater(a: 10, b: 5));
21
22         Increment increment = int a -> a + 1;
23         System.out.println(increment.incrementByOne(a: 7));
24
25         Concatenator concat = (String a, String b) -> a + b;
26         System.out.println(concat.concatenate(a: "Hello, ", b: "World!"));
27
28         ToUpperCase upper = String str -> str.toUpperCase();
29         System.out.println(upper.convert(str: "java"));
30     }
31 }
32
```

Run | LambdaExamples x  
s8feature1 > src > LambdaExamples.java > LambdaExamples 17:1 LF UTF-8 4 sp

## Output



```
1 /usr/lib/jvm/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
2 true
3 8
4 Hello, World!
5 JAVA
6
7 Process finished with exit code 0
8
9
```

s8feature1 > src > LambdaExamples.java > LambdaExamples 17:1 LF UTF-8 4 sp

2. 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

The screenshot displays an IDE with a project named 'Java8feature1'. The 'src' directory contains three files: 'LambdaExamples.java', 'Main.java', and 'MethodReference.java'. The 'MethodReference.java' file is open, showing the following code:

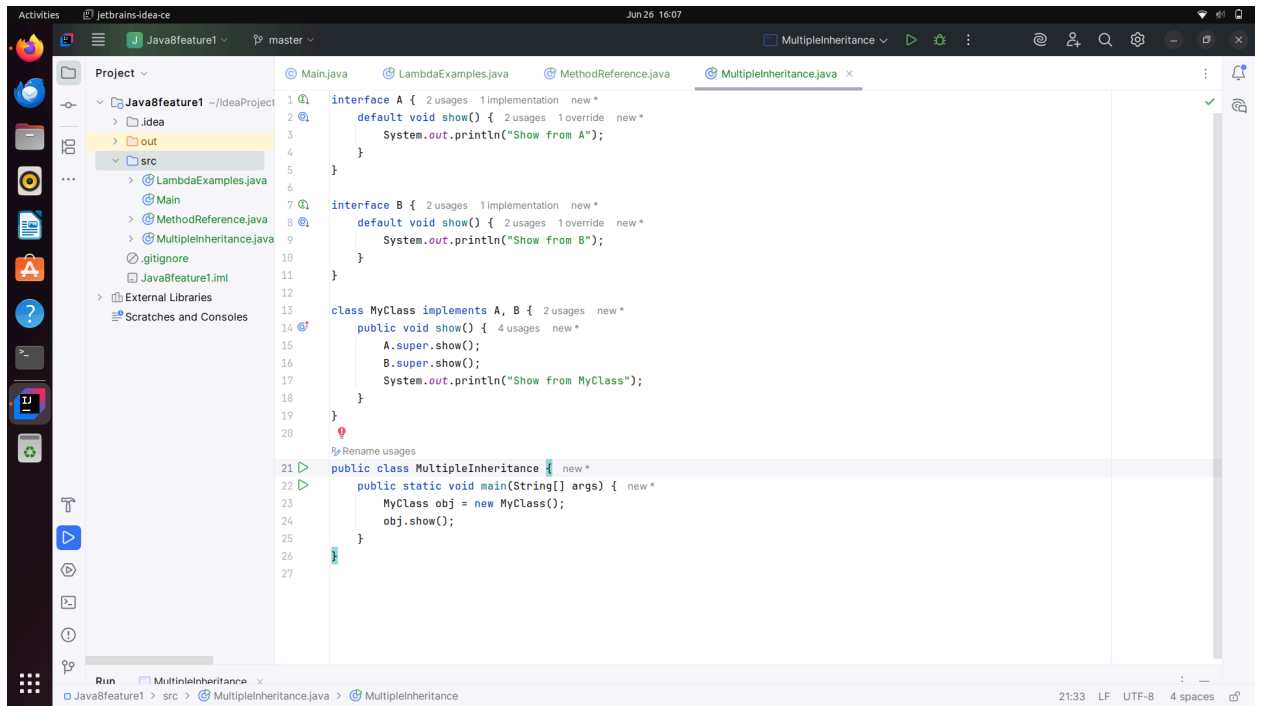
```
1 @FunctionalInterface 3 usages new *
2 interface Operation{
3     int apply(int a,int b); 3 usages new *
4 }
5 class MathOperation{ 2 usages new *
6     public int add(int a,int b){ 1 usage new *
7         return a+b;
8     }
9     public int sub(int a,int b){ 1 usage new *
10        return a-b;
11    }
12    public int multiply(int a, int b){ 1 usage new *
13        return a*b;
14    }
15 }
16
17 public class MethodReference { new *
18     public static void main(String[] args) { new *
19         MathOperation math=new MathOperation();
20         Operation addition= math::add;
21         System.out.println(addition.apply( a: 20, b: 10));
22         Operation subtraction=math::sub;
23         System.out.println(subtraction.apply( a: 20, b: 10));
24         Operation multiplication =math::multiply;
25         System.out.println(multiplication.apply( a: 20, b: 10));
26     }
27 }
28
29 }
30
```

The 'Run' button is highlighted, and the output window shows the following command and results:

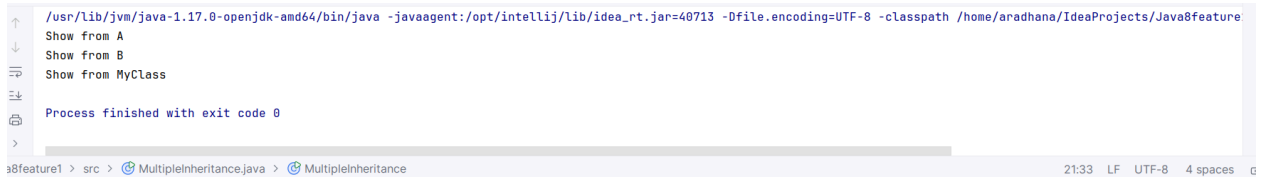
```
Run MethodReference x
/usr/lib/jvm/java-1.17.0-openjdk-amd64/bin/java -javaagent:/opt/intellij/lib/idea_rt.jar=35681 -Dfile.encoding=UTF-8 -classpath /home/aradhana/IdeaProjects/Java8fe
30
10
200
Process finished with exit code 0
```

The output window also shows the file path: 'src > MethodReference.java > MethodReference > main' and the status: '24:39 LF UTF-8 4 sp'.

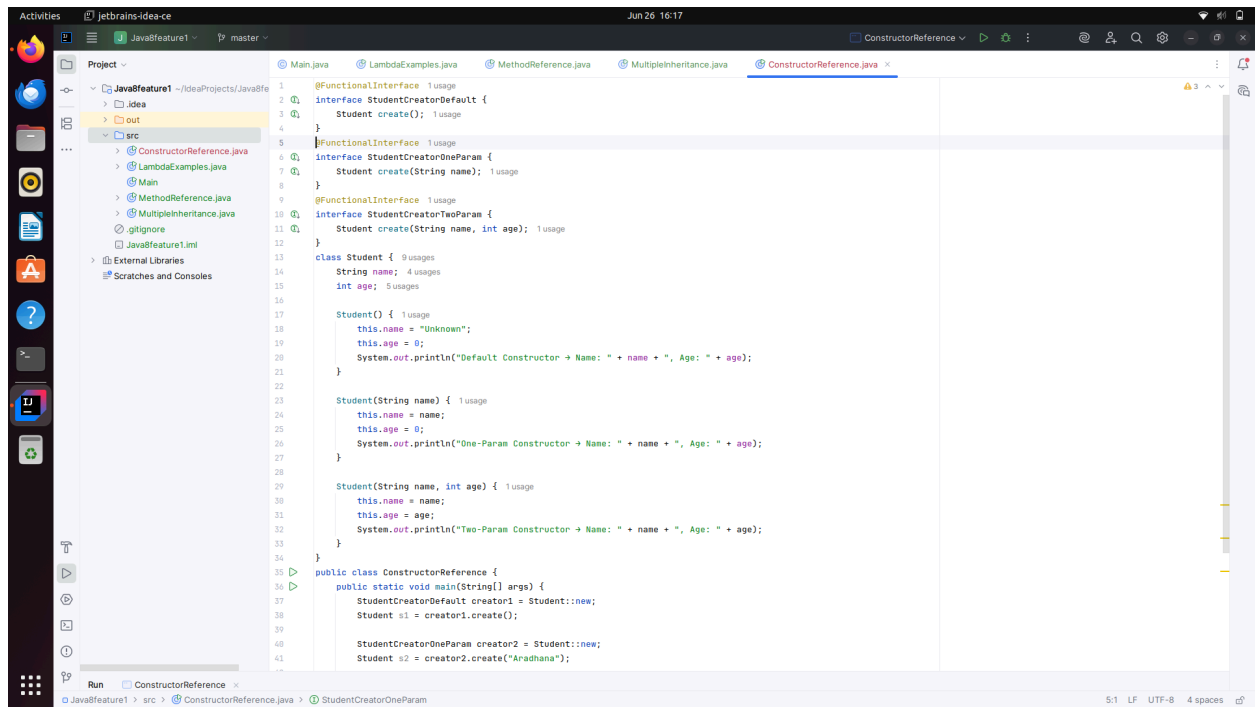
### 3. Implement multiple inheritance with default method inside interface.



## Output



## 4. Write a program to implement constructor reference



## Output

