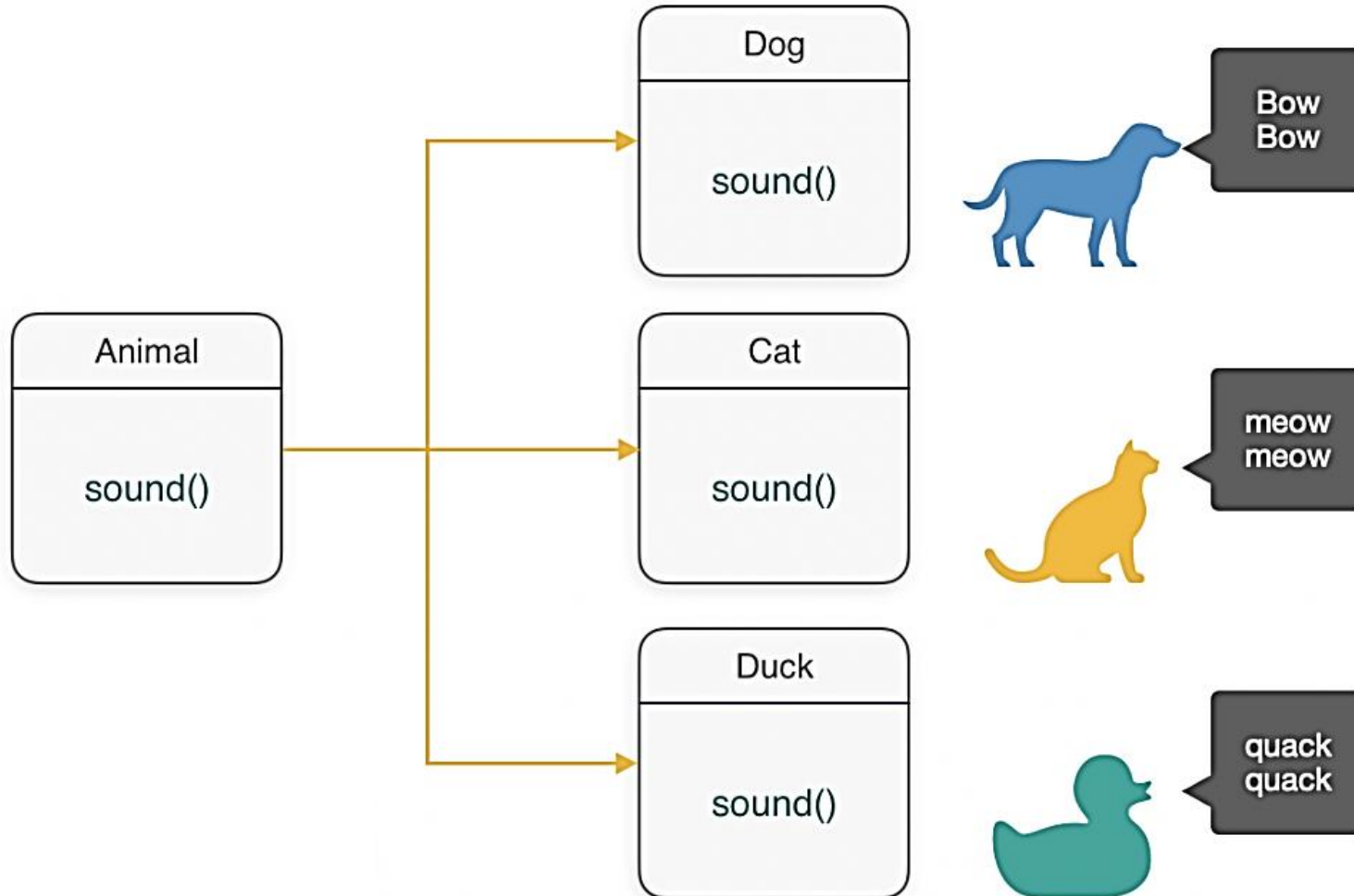


# POLYMORPHISM

(METHOD OVERLOADING)



IN THE NAME OF ALLAH

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

THE GRACIOUS, THE MERCIFUL.



# **GETS - GETTING EDUCATION WITH TECHNOLOGICAL SYSTEM**

Instructor: Sir A.Rehman Ali Brohi

# LECTURE: 8

## Polymorphism in java



# What is Polymorphism?

**Poly**  
**(many)**

**Morphism**  
**(forms)**

**Shapes: circle, square, triangle etc.**

**Sound: bark, roar etc**

**Water: solid, liquid, gas**

polymorphism in Java  
allows us to perform the  
same action in many  
different ways

# Types of Polymorphism

## 1. Compile-time/Static Polymorphism

Method Overloading

Compiler handle it

## 1. Run-Time/Dynamic Polymorphism

Method Overriding

JVM handle it

# Method Overloading

1. Same Name
2. Same Class
3. Different Arguments
  - No of Arg
  - Seg of Arg
  - Types of Arg

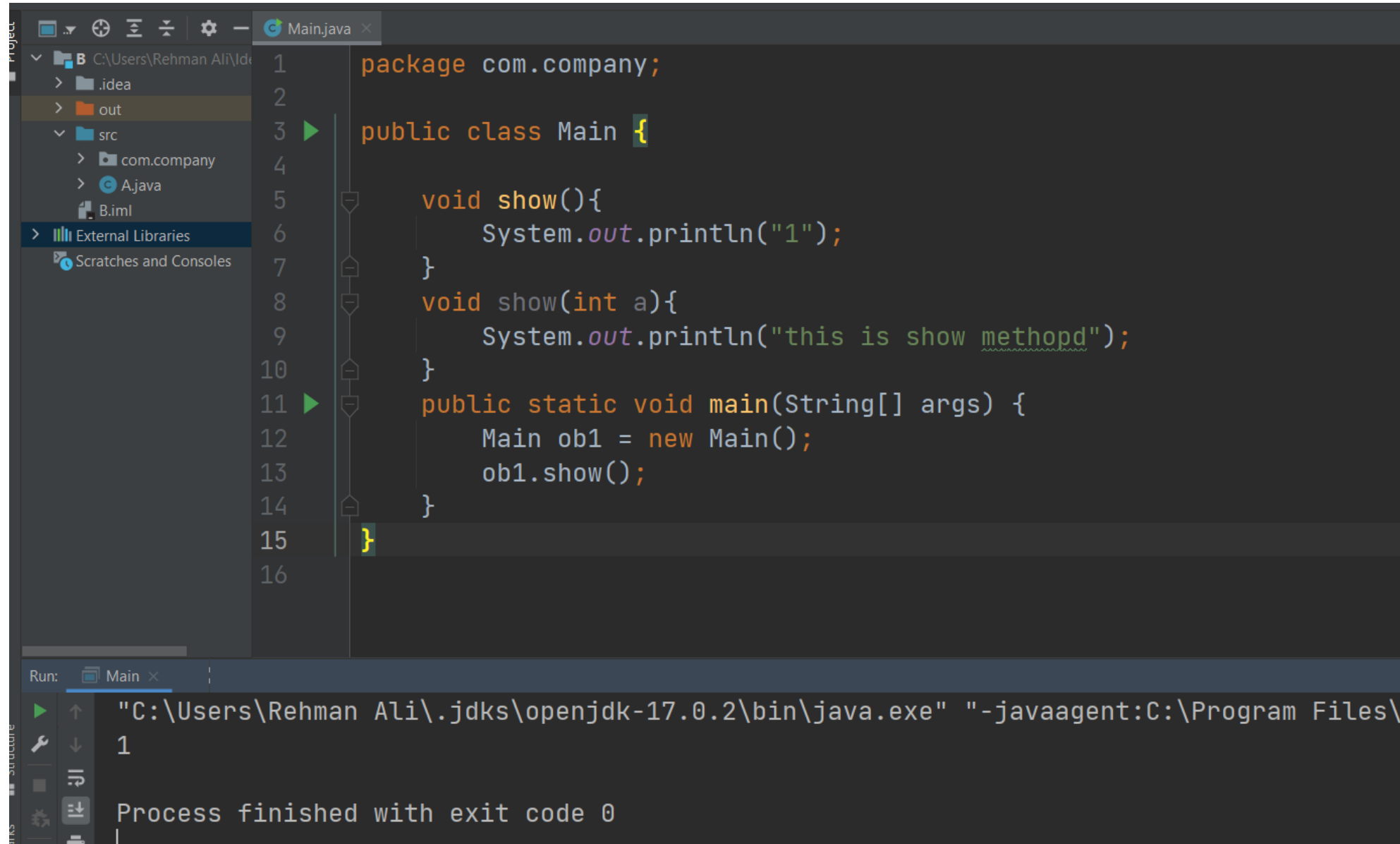
# Method Overriding

1. Same Name
2. Different Class
3. Same Arguments
  - No of Arg
  - Seg of Arg
  - Types of Arg
4. Inheritance (IS-A)



In this exercise we will  
practice with **method  
overloading**

Program to create **same-class** and **same-method** but **different-in-arg** it is known as Polymorphism



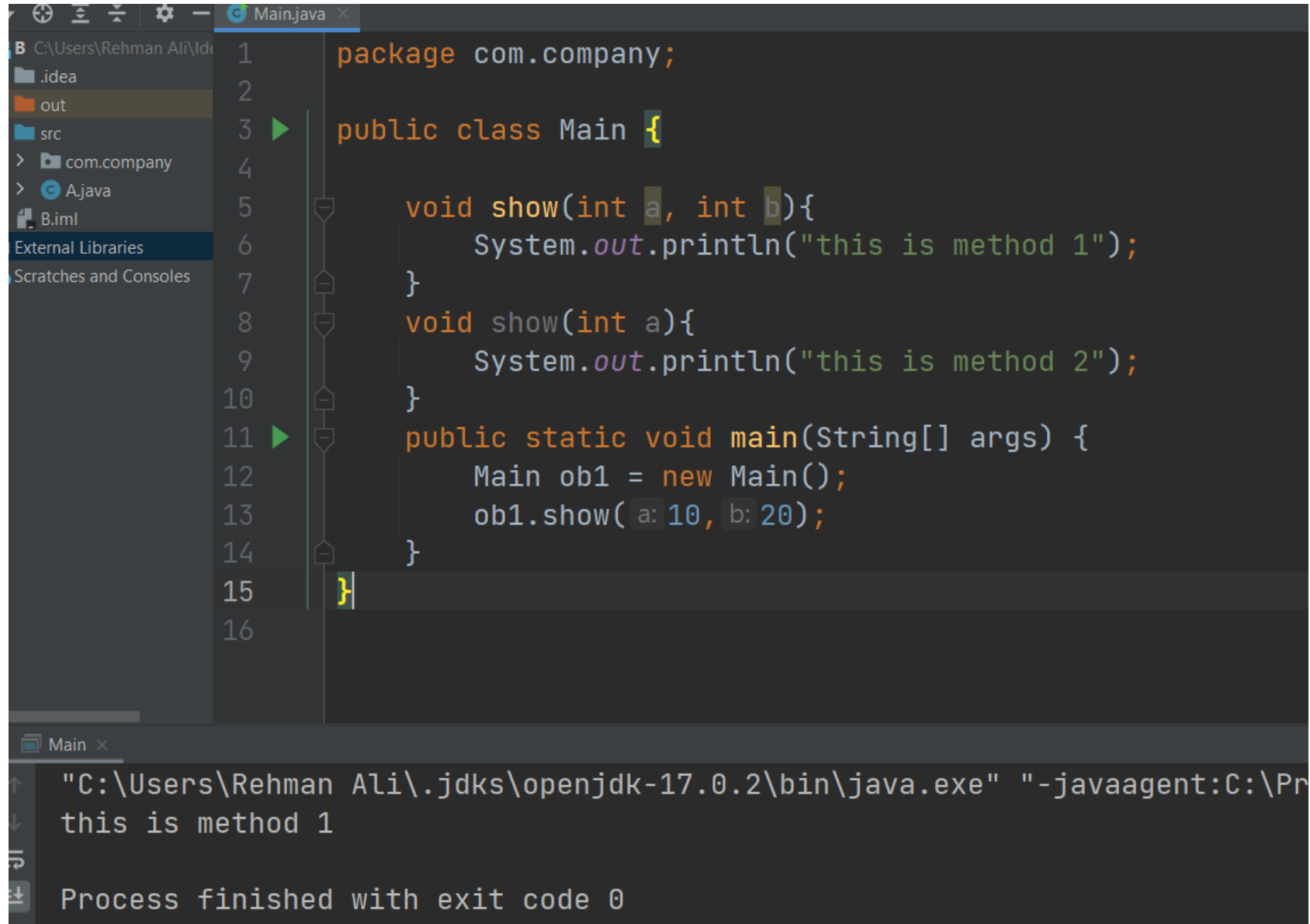
The screenshot displays an IDE with a project structure on the left and a code editor in the center. The project structure includes a 'src' directory with a 'com.company' package containing 'A.java' and 'B.iml'. The code editor shows the following Java code in 'Main.java':

```
1 package com.company;
2
3 public class Main {
4
5     void show(){
6         System.out.println("1");
7     }
8     void show(int a){
9         System.out.println("this is show methopd");
10    }
11    public static void main(String[] args) {
12        Main ob1 = new Main();
13        ob1.show();
14    }
15 }
16
```

Below the code editor, the 'Run' tab shows the execution command and output:

```
Run: Main x
"C:\Users\Rehman Ali\.jdk\openjdk-17.0.2\bin\java.exe" "-javaagent:C:\Program Files\
1
Process finished with exit code 0
```

# Program to change the No of arguments



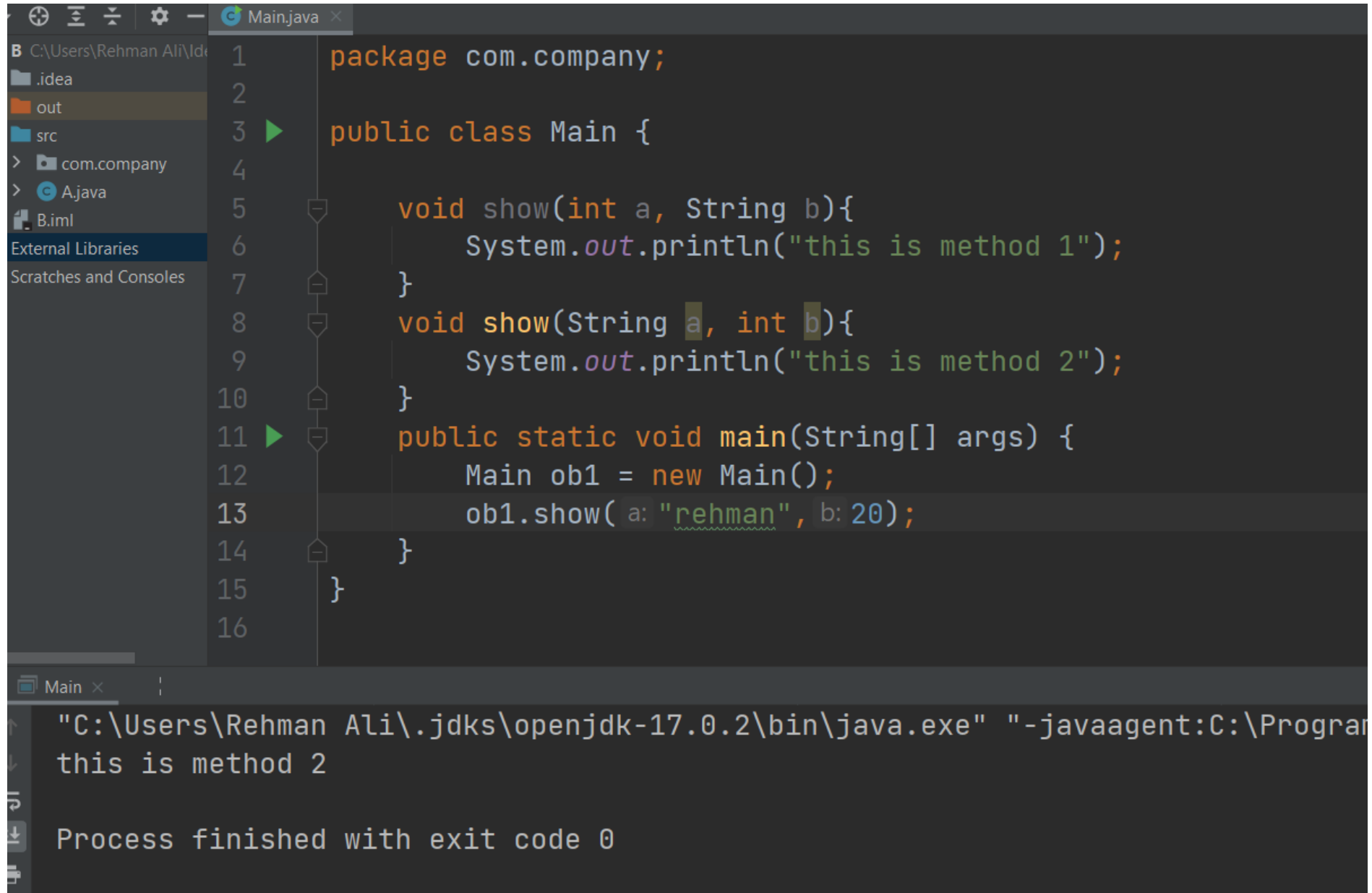
The screenshot shows an IDE with a project structure on the left and a code editor in the center. The project structure includes a folder named 'com.company' containing 'A.java' and 'B.iml'. The code editor displays the following Java code:

```
1 package com.company;
2
3 public class Main {
4
5     void show(int a, int b){
6         System.out.println("this is method 1");
7     }
8     void show(int a){
9         System.out.println("this is method 2");
10    }
11    public static void main(String[] args) {
12        Main ob1 = new Main();
13        ob1.show(a: 10, b: 20);
14    }
15 }
16
```

Below the code editor, the output console shows the execution of the program:

```
↑ "C:\Users\Rehman Ali\.jdk\openjdk-17.0.2\bin\java.exe" "-javaagent:C:\Pr
↓ this is method 1
↕
↑↓ Process finished with exit code 0
```

# Program to change the Seg of arguments



The screenshot shows an IDE with a project structure on the left and a code editor in the center. The project structure includes a folder named 'com.company' containing a file 'A.java'. The code editor displays the following Java code:

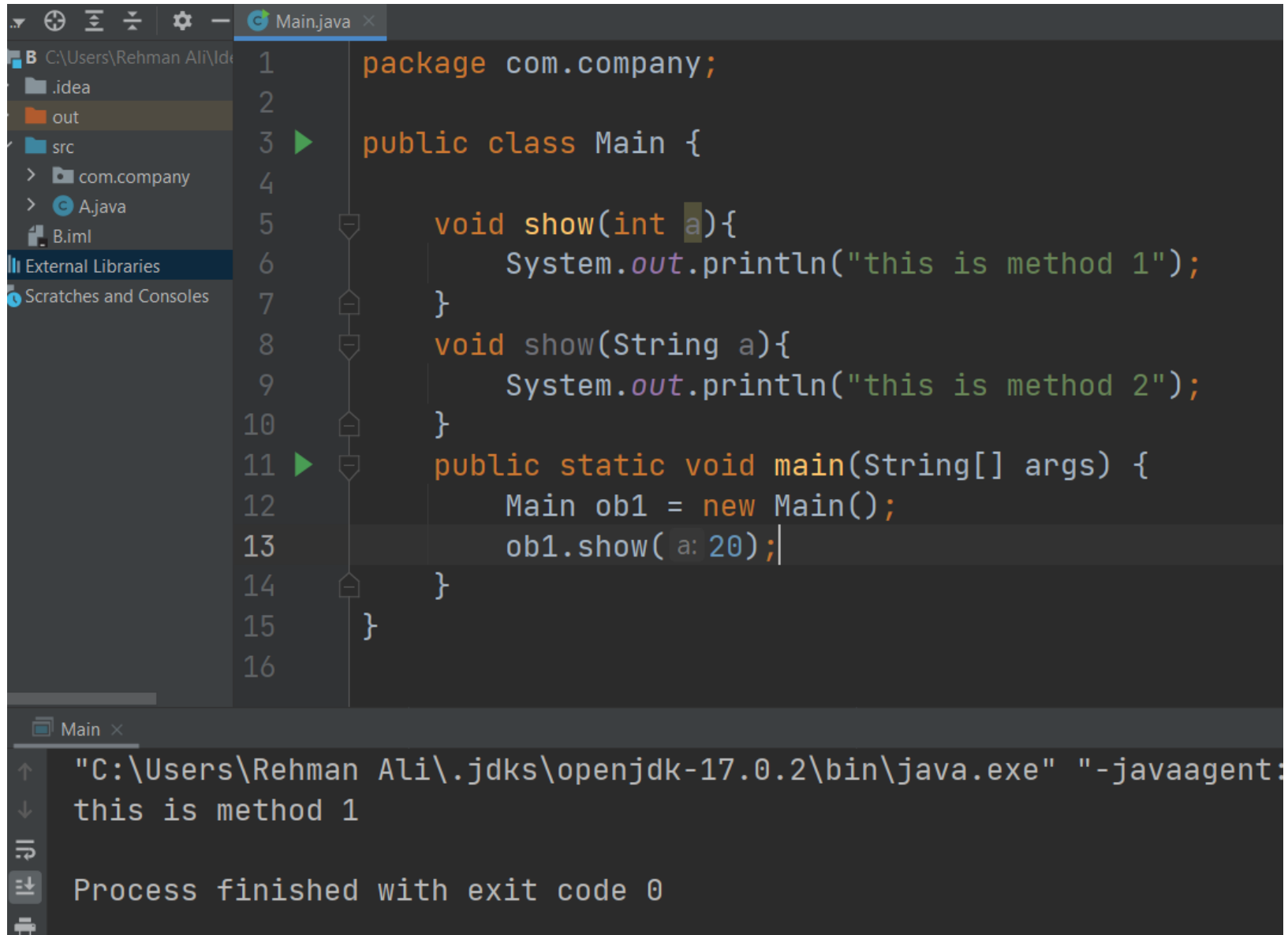
```
1 package com.company;
2
3 public class Main {
4
5     void show(int a, String b){
6         System.out.println("this is method 1");
7     }
8     void show(String a, int b){
9         System.out.println("this is method 2");
10    }
11    public static void main(String[] args) {
12        Main ob1 = new Main();
13        ob1.show(a: "rehman", b: 20);
14    }
15 }
16
```

The output window at the bottom shows the command used to run the program and the resulting output:

```
"C:\Users\Rehman Ali\.jdk\openjdk-17.0.2\bin\java.exe" "-javaagent:C:\Program
this is method 2

Process finished with exit code 0
```

# Program to change the Type of arguments



The screenshot shows an IDE with a project structure on the left and a code editor in the center. The project structure includes a folder named 'com.company' containing 'A.java' and 'B.iml'. The code editor displays the following Java code:

```
1 package com.company;
2
3 public class Main {
4
5     void show(int a){
6         System.out.println("this is method 1");
7     }
8     void show(String a){
9         System.out.println("this is method 2");
10    }
11    public static void main(String[] args) {
12        Main ob1 = new Main();
13        ob1.show( a: 20);
14    }
15 }
16
```

Below the code editor, the console output is visible, showing the command used to run the program and the resulting output:

```
"C:\Users\Rehman Ali\.jdk\openjdk-17.0.2\bin\java.exe" "-javaagent:
this is method 1
Process finished with exit code 0
```

IMPORTANT NOTE (useful interview questions)

1. Can we achieve method overloading by changing return type of method only.
2. Can we overload java main() method?

# 1. Answer is NO

```
Main.java x
3 public class Main {
4
5     void show(int a){
6         System.out.println("this is method 1");
7     }
8     String show(int a){
9         System.out.println("this is method 2");
10    }
11    public static void main(String[] args) {
12        Main ob1 = new Main();
13        ob1.show(a: 20);
14    }
15 }
16
```

At 12/07/2022 4:23 PM with 1 ei 2 sec, 951 ms  
src\com\company 1 error  
d show(int) is already defined in class com.com

C:\Users\Rehman Ali\IdeaProjects\B\src\com\company\Main.java:8:12  
java: method show(int) is already defined in class com.company.Main

X

## 2. Answer is YES



The screenshot shows an IDE with a project structure on the left and a code editor in the center. The project structure includes a folder named 'out' and a file named 'Main.java'. The code editor displays the following Java code:

```
1 package com.company;  
2  
3 public class Main {  
4  
5     public static void main(String[] args) {  
6         System.out.println("this is main method");  
7         Main ob1 = new Main();  
8         ob1.main(a: 20);  
9     }  
10    public static void main(int a) {  
11        System.out.println("this is 2nd main ");  
12    }  
13 }  
14
```

The output window at the bottom shows the execution of the program, displaying the following lines:

```
"C:\Users\Rehman Ali\.jdk\openjdk-17.0.2\bin\java.exe" "-javaagent:C:\Progra  
this is main method  
this is 2nd main
```