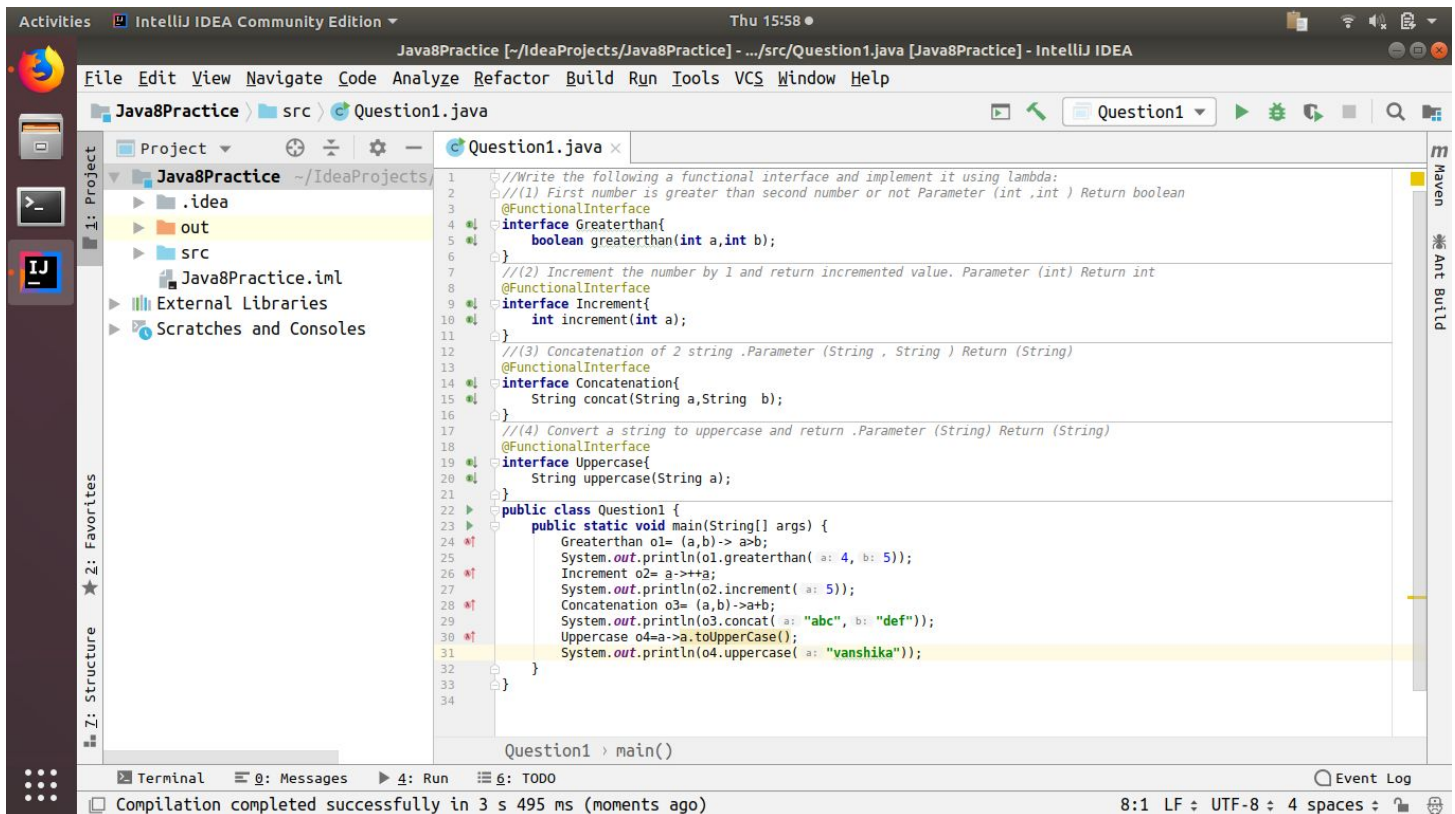


## Java 8-features - Assignment

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

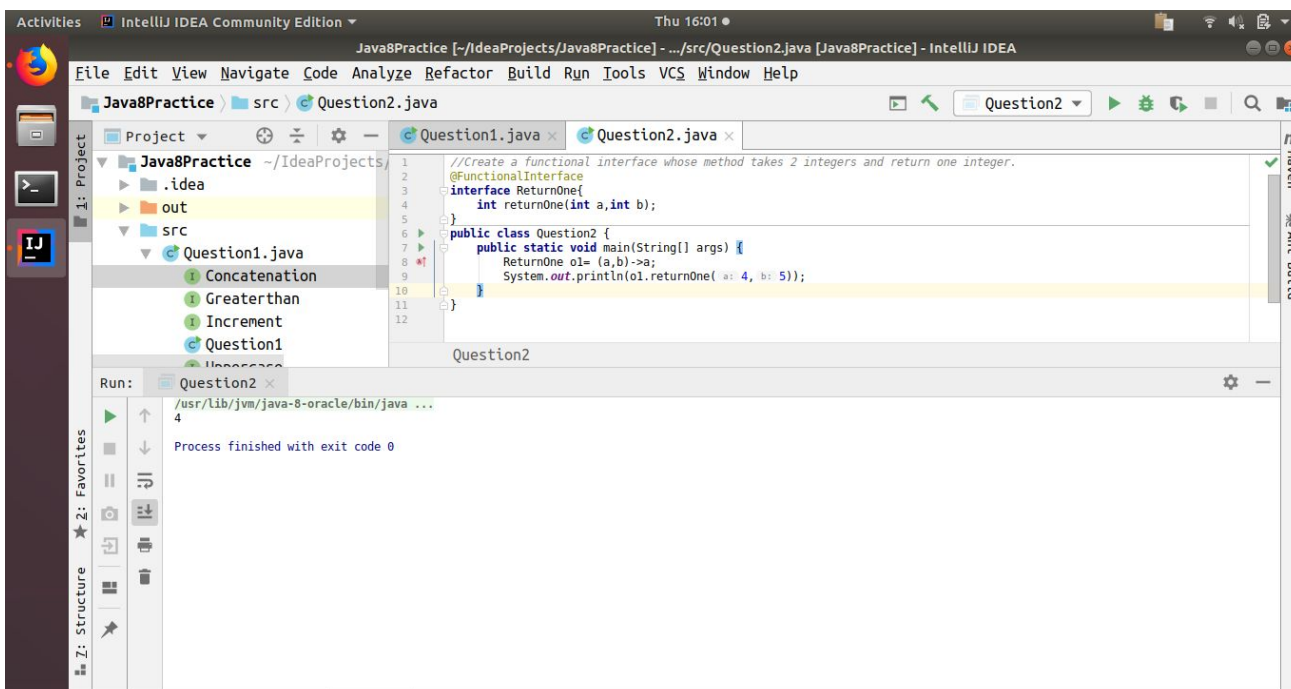


```
1 //Write the following a functional interface and implement it using lambda:
2 // (1) First number is greater than second number or not Parameter (int ,int ) Return boolean
3 @FunctionalInterface
4 interface Greaterthan{
5     boolean greaterthan(int a,int b);
6 }
7 // (2) Increment the number by 1 and return incremented value. Parameter (int) Return int
8 @FunctionalInterface
9 interface Increment{
10     int increment(int a);
11 }
12 // (3) Concatenation of 2 string .Parameter (String , String ) Return (String)
13 @FunctionalInterface
14 interface Concatenation{
15     String concat(String a,String b);
16 }
17 // (4) Convert a string to uppercase and return .Parameter (String) Return (String)
18 @FunctionalInterface
19 interface Uppercase{
20     String uppercase(String a);
21 }
22 public class Question1 {
23     public static void main(String[] args) {
24         Greaterthan o1= (a,b)-> a>b;
25         System.out.println(o1.greaterthan( 4, b: 5));
26         Increment o2= a-> ++a;
27         System.out.println(o2.increment( a: 5));
28         Concatenation o3= (a,b)-> a+b;
29         System.out.println(o3.concat( a: "abc", b: "def"));
30         Uppercase o4=a-> a.toUpperCase();
31         System.out.println(o4.uppercase( a: "vanshika"));
32     }
33 }
34
```

Question1 > main()

Compilation completed successfully in 3 s 495 ms (moments ago)

- Create a functional interface whose method takes 2 integers and return one integer.



```
1 //Create a functional interface whose method takes 2 integers and return one integer.
2 @FunctionalInterface
3 interface ReturnOne{
4     int returnOne(int a,int b);
5 }
6 public class Question2 {
7     public static void main(String[] args) {
8         ReturnOne o1= (a,b)-> a;
9         System.out.println(o1.returnOne( a: 4, b: 5));
10    }
11 }
12
```

Question2

Run: Question2

/usr/lib/jvm/java-8-oracle/bin/java ...

Process finished with exit code 0

- Using (instance) Method reference create and apply add and subtract method and using (Static)

```

1 interface MethodReferenceInterface {
2     void display();
3 }
4
5 public class Question8 {
6
7     static void staticMethod(){
8         System.out.println("Static method");
9     }
10
11     void instanceMethod(){
12         System.out.println("Instance Method");
13     }
14
15     public static void main(String[] args) {
16         MethodReferenceInterface staticMethod= Question8::staticMethod;
17         staticMethod.display();
18         MethodReferenceInterface instanceMethod=new Question8()::instanceMethod;
19         instanceMethod.display();
20     }
21 }

```

Run: Question8 x

```

/usr/lib/jvm/java-8-oracle/bin/java ...
Static method
Instance Method
Process finished with exit code 0

```

Compilation completed successfully in 2 s 988 ms (moments ago)

- Method reference create and apply multiplication method for the functional interface created.

```

1 //Using (instance) Method reference create and apply add and subtract method and using (Static)
2 @FunctionalInterface
3 interface MethodInfrnce{
4     int implementDoSomething(int a,int b);
5 }
6
7 public class Question3 {
8     @Contract(pure = true) public static int doSomething(int a, int b) { return a*b; }
9
10     public int add(int a,int b){return a+b;}
11     public int sub(int a,int b){return a-b;}
12     public static void main(String[] args) {
13         MethodInfrnce o1=Question3::doSomething;
14         MethodInfrnce o2=new Question3()::add;
15         MethodInfrnce o3=new Question3()::sub;
16
17         System.out.println(o1.implementDoSomething( a: 4, b: 5));
18         System.out.println(o2.implementDoSomething( a: 4, b: 5));
19         System.out.println(o3.implementDoSomething( a: 4, b: 5));
20     }
21 }

```

Run: Question3 x

```

/usr/lib/jvm/java-8-oracle/bin/java ...
20
9
-1
Process finished with exit code 0

```

Compilation completed successfully in 6 s 761 ms (moments ago)

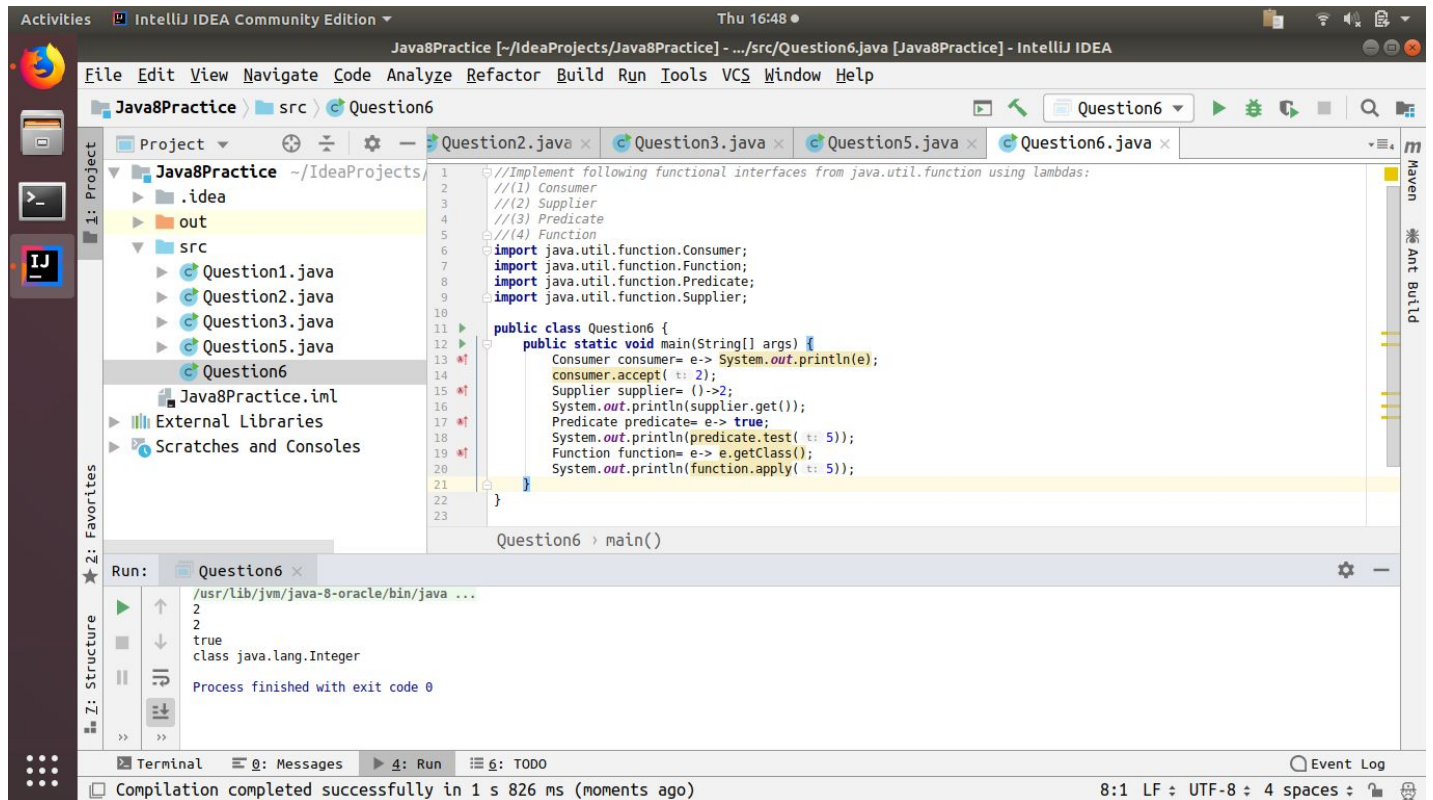
- Create an Employee Class with instance variables (String) name, (Integer)age, (String)city and get the instance of the Class using constructor reference

The screenshot shows the IntelliJ IDEA interface with the following components:

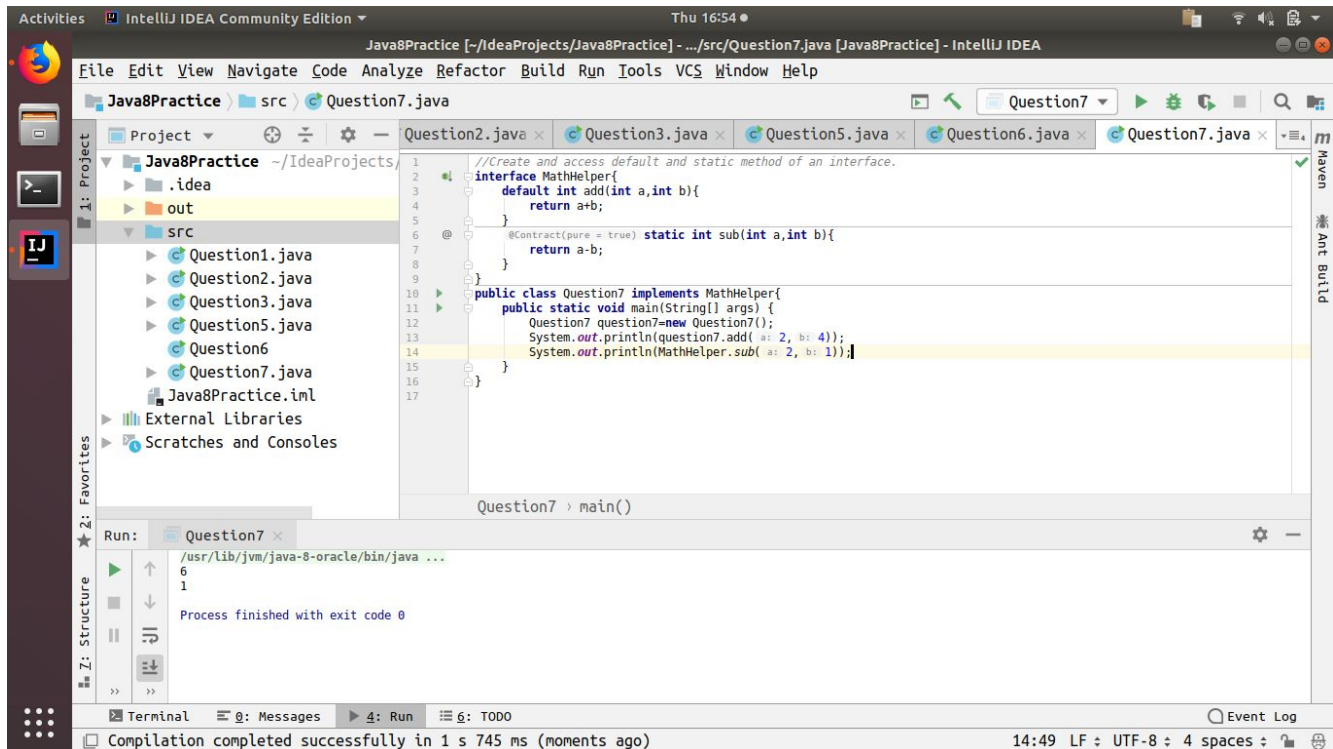
- Project Structure:** A tree view on the left showing the project 'Java8Practice' with subfolders '.idea', 'out', and 'src'. The 'src' folder contains files 'Question1.java', 'Question2.java', 'Question3.java', 'Question5.java', and 'MethodInference.java'.
- Code Editor:** The main window displays the code for 'Question5.java'. It includes a comment at the top: '//Create an Employee Class with instance variables (String) name, (Integer)age, (String)city and get the instance of the Class using constructor reference'. Below this is a functional interface 'GetInstance' with a method 'Employee getInstance(String name,Integer age,String city);'. The 'Employee' class is defined with instance variables 'name', 'age', and 'city', a constructor, and a 'toString()' method. The 'Question5' class has a 'main' method that creates an 'Employee' object and prints its details.
- Run Console:** At the bottom, the 'Run' tab shows the output: 'Employee{name='Vanshika', age=20, city='Delhi'}'.
- Status Bar:** The bottom status bar indicates 'Compilation completed successfully in 6 s 374 ms (moments ago)'.

- Implement following functional interfaces from java.util.function using lambdas:
  - (1) Consumer
  - (2) Supplier
  - (3) Predicate
  - (4) Function

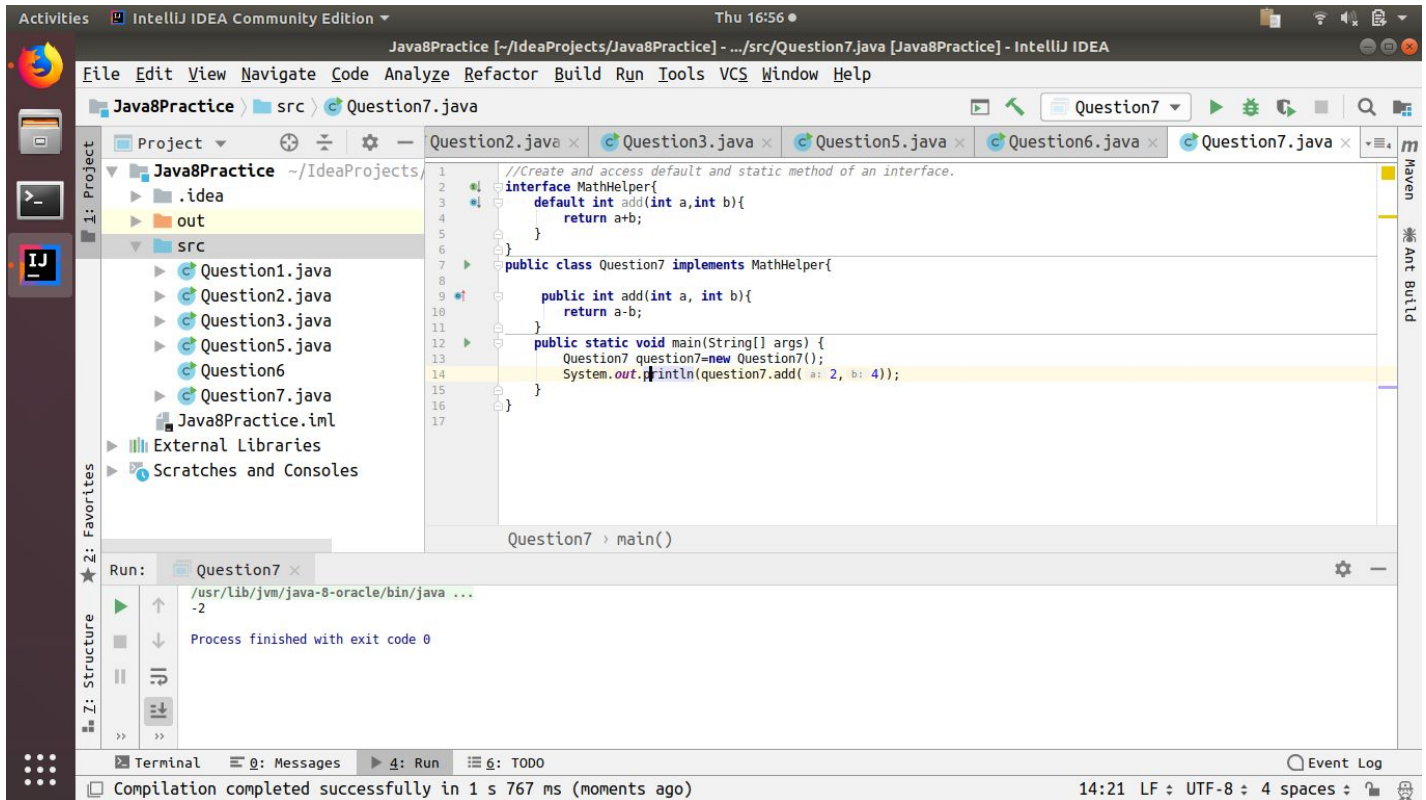




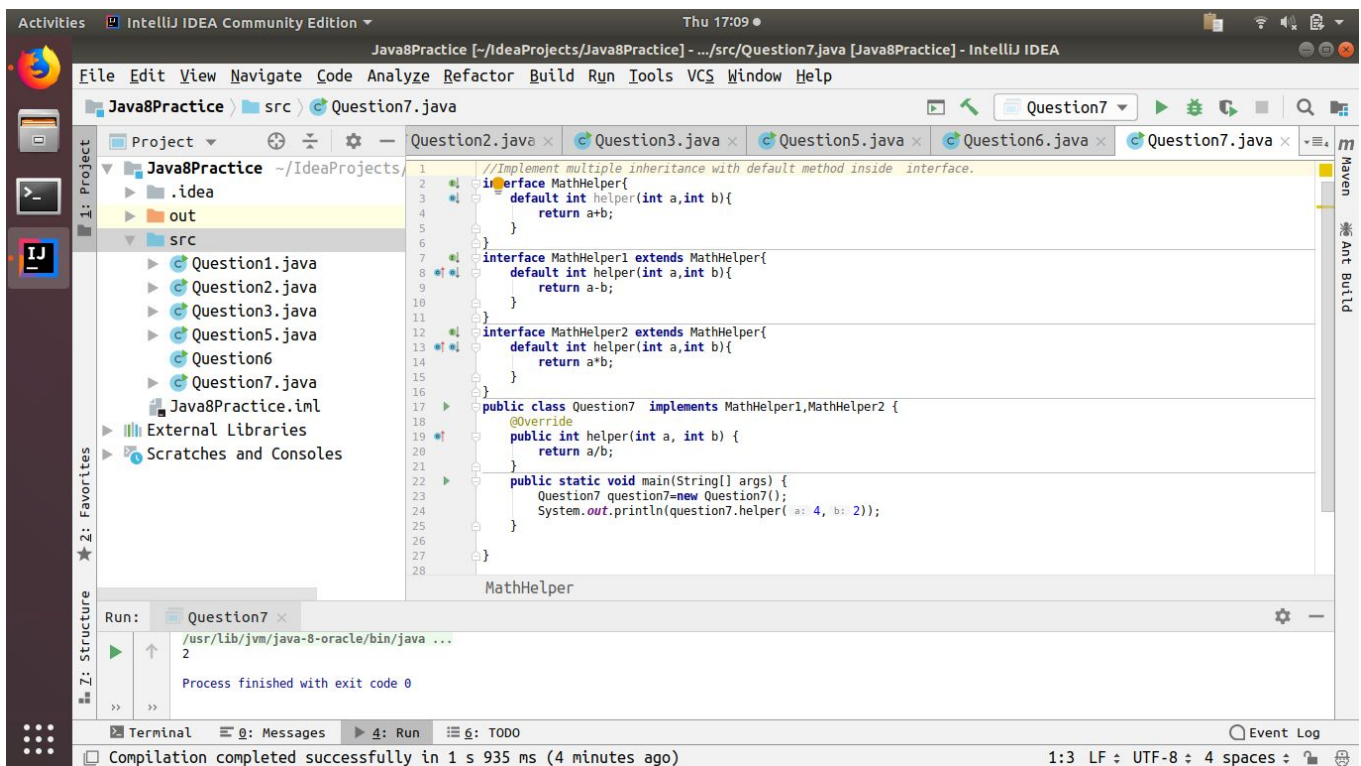
- Create and access default and static method of an interface.



- Override the default method of the interface.



- Implement multiple inheritance with default method inside interface.



- Collect all the even numbers from an integer list.

The screenshot shows the IntelliJ IDEA interface with the 'Question8.java' file open. The code implements a method to collect even numbers from a list. The 'Run' window shows the output of the program, which prints the even numbers from the list: 2, 4, 6, 8, 10.

```
//Collect all the even numbers from an integer list.
import java.util.Arrays;
import java.util.List;

public class Question8 {
    public static void main(String[] args) {
        List<Integer> list= Arrays.asList(1,2,3,4,5,6,7,8,9,10);
        list.stream().filter(e->e%2==0).forEach(System.out::println);
    }
}
```

Run: Question8

/usr/lib/jvm/java-8-oracle/bin/java ...

2  
4  
6  
8  
10

Process finished with exit code 0

Compilation completed successfully in 1 s 936 ms (moments ago)

- Sum all the numbers greater than 5 in the integer list.

The screenshot shows the IntelliJ IDEA interface with the 'Question8.java' file open. The code implements a method to sum all numbers greater than 5 in a list. The 'Run' window shows the output of the program, which prints the sum of numbers greater than 5: 40.

```
//Sum all the numbers greater than 5 in the integer list.
import java.util.Arrays;
import java.util.Collections;
import java.util.List;
import java.util.stream.Collectors;

public class Question8 {
    public static void main(String[] args) {
        List<Integer> list= Arrays.asList(1,2,3,4,5,6,7,8,9,10);
        int sum=0;
        System.out.println(list.stream().filter(e -> e > 5).mapToInt(e -> e).sum());
    }
}
```

Run: Question8

/usr/lib/jvm/java-8-oracle/bin/java ...

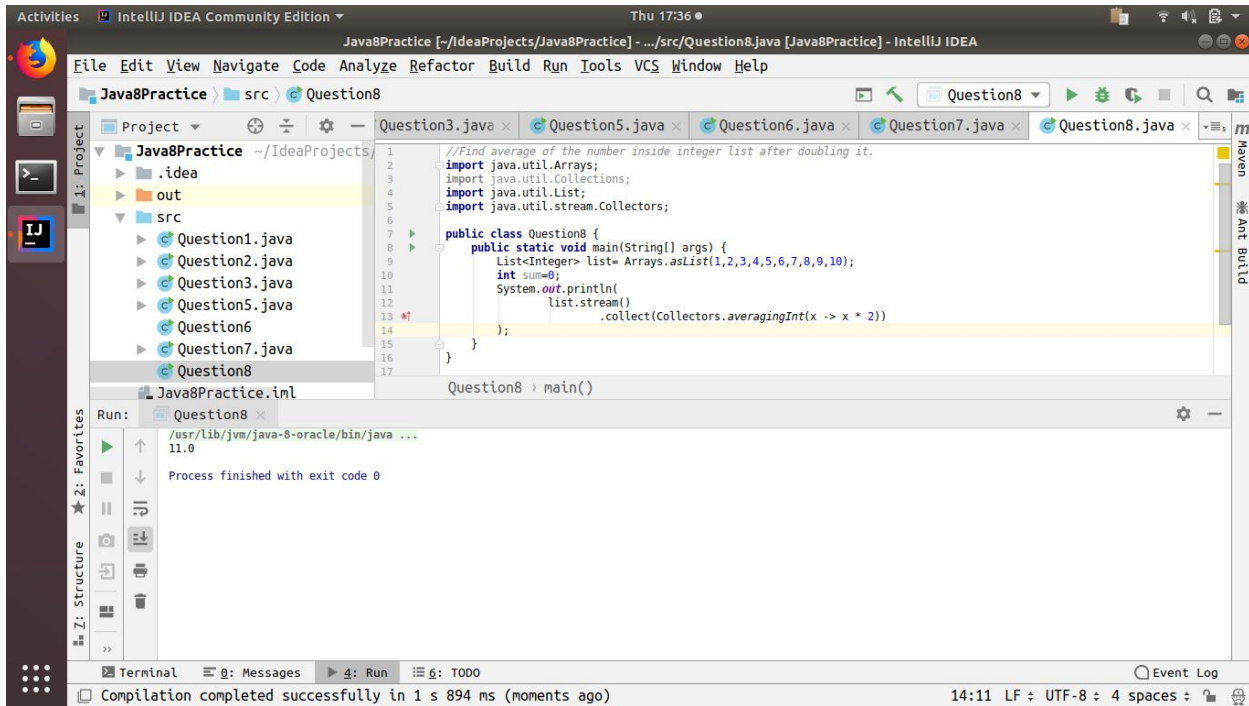
40

Process finished with exit code 0

Compilation completed successfully in 2 s 10 ms (moments ago)



- Find average of the number inside integer list after doubling it.



```
1 //Find average of the number inside integer list after doubling it.
2 import java.util.Arrays;
3 import java.util.Collections;
4 import java.util.List;
5 import java.util.stream.Collectors;
6
7 public class Question8 {
8     public static void main(String[] args) {
9         List<Integer> list= Arrays.asList(1,2,3,4,5,6,7,8,9,10);
10         int sum=0;
11         System.out.println(
12             list.stream()
13                 .collect(Collectors.averagingInt(x -> x * 2))
14         );
15     }
16 }
17
18 Question8 > main()
```

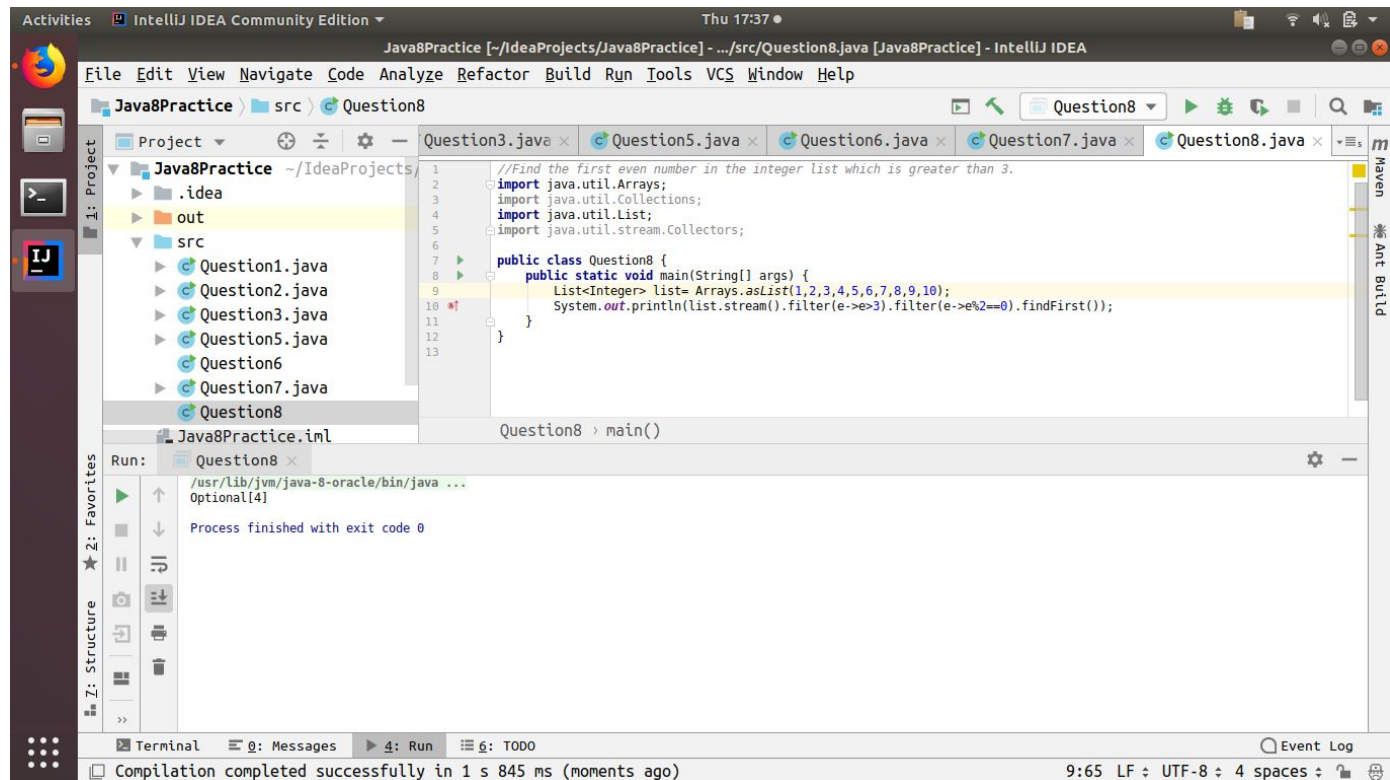
Run: Question8 x

/usr/lib/jvm/java-8-oracle/bin/java ...  
11.0

Process finished with exit code 0

Compilation completed successfully in 1 s 894 ms (moments ago)

- Find the first even number in the integer list which is greater than 3.



```
1 //Find the first even number in the integer list which is greater than 3.
2 import java.util.Arrays;
3 import java.util.Collections;
4 import java.util.List;
5 import java.util.stream.Collectors;
6
7 public class Question8 {
8     public static void main(String[] args) {
9         List<Integer> list= Arrays.asList(1,2,3,4,5,6,7,8,9,10);
10         System.out.println(list.stream().filter(e->e>3).filter(e->e%2==0).findFirst());
11     }
12 }
13
14 Question8 > main()
```

Run: Question8 x

/usr/lib/jvm/java-8-oracle/bin/java ...  
Optional[4]

Process finished with exit code 0

Compilation completed successfully in 1 s 845 ms (moments ago)