

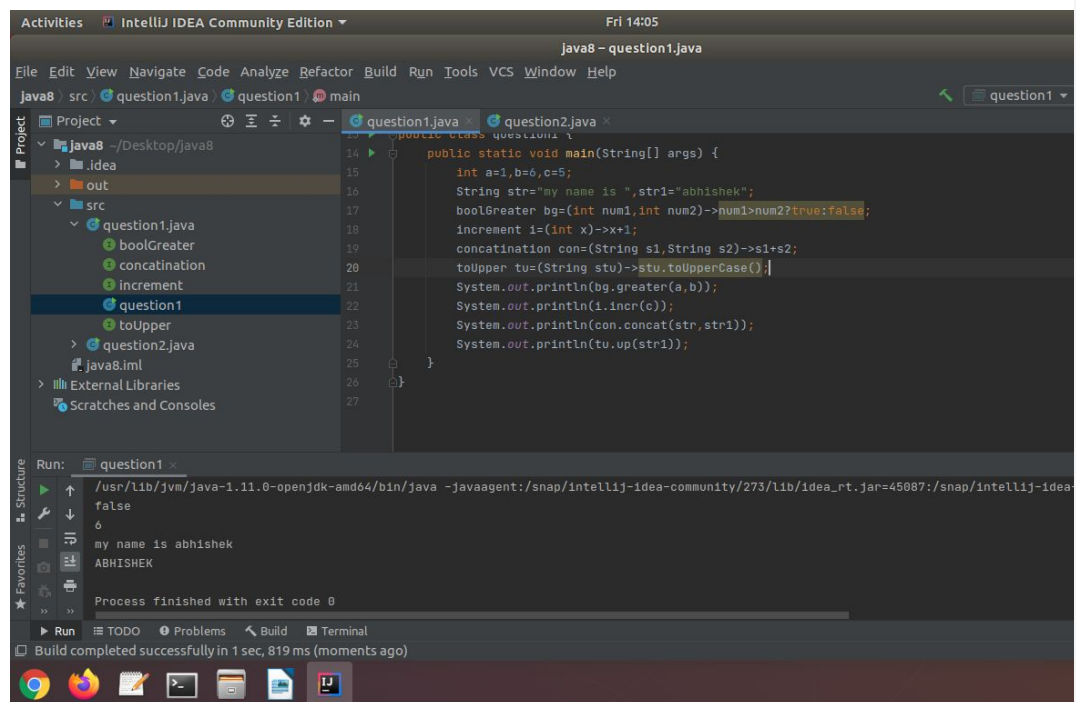
1. Write the following a functional interface and implement it using lambda:

(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) Concatination of 2 string Parameter (String , String) Return (String)

(4) Convert a string to uppercase and return . Parameter (String) Return (String)



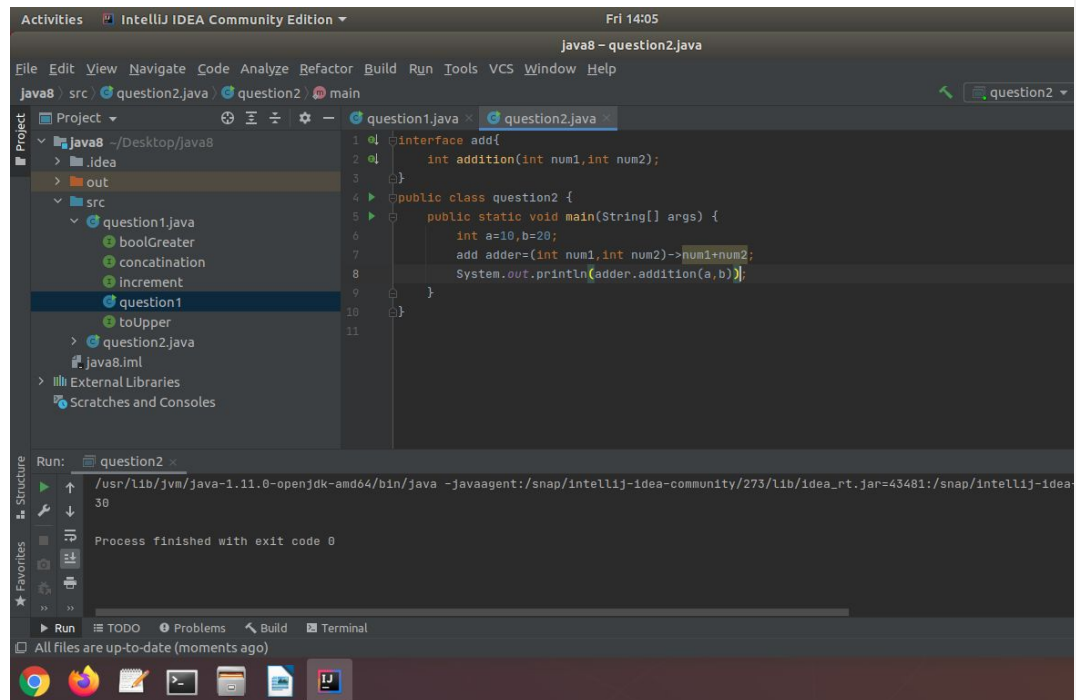
The screenshot shows the IntelliJ IDEA Community Edition interface. The main editor displays the code for `question1.java`, which implements four functional interfaces: `boolGreater`, `increment`, `concatination`, and `toUpper`. The code uses lambda expressions to implement these interfaces. The `main` method tests each interface with specific inputs and prints the results.

```
public class question1 {  
    public static void main(String[] args) {  
        int a=1,b=0,c=5;  
        String str="my name is ",str1="abhishek";  
        boolGreater bg=(int num1,int num2)->num1>num2?true:false;  
        increment i=(int x)->x+1;  
        concatination con=(String s1,String s2)->s1+s2;  
        toUpper tu=(String stu)->stu.toUpperCase();  
        System.out.println(bg.greater(a,b));  
        System.out.println(i.incr(c));  
        System.out.println(con.concat(str,str1));  
        System.out.println(tu.up(str1));  
    }  
}
```

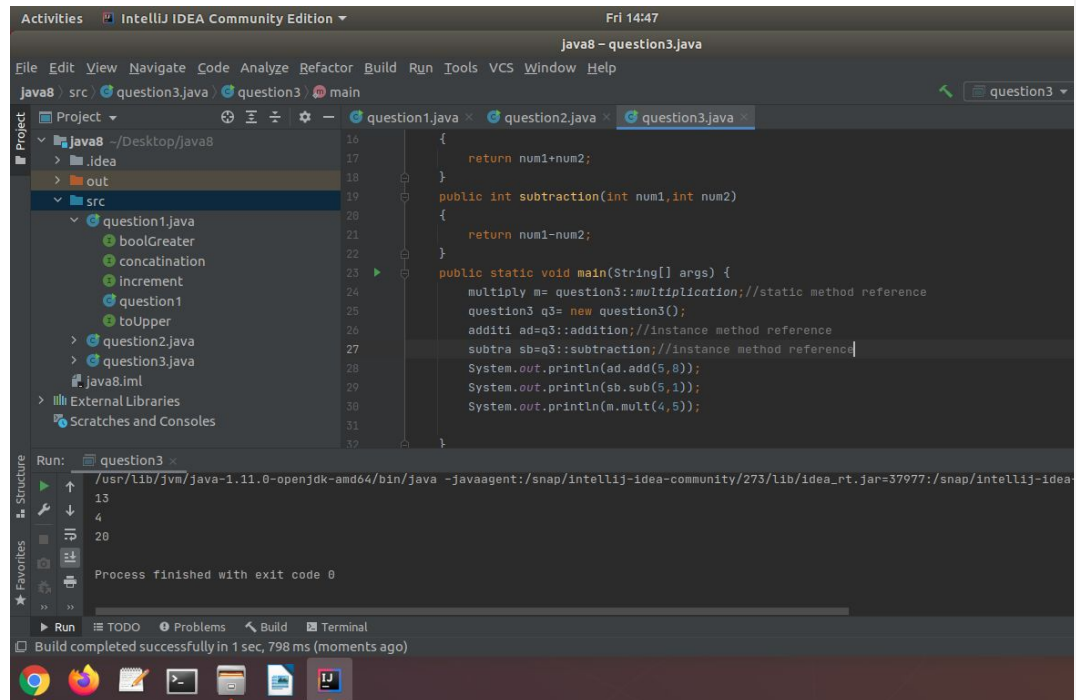
The Run window at the bottom shows the output of the program:

```
Run: question1 x  
/usr/lib/jvm/java-11.0-openjdk-amd64/bin/java -javaagent:/snap/intellij-idea-community/273/lib/idea_rt.jar=45087:/snap/intellij-idea  
false  
6  
my name is abhishek  
ABHISHEK  
Process finished with exit code 0  
Build completed successfully in 1 sec, 819 ms (moments ago)
```

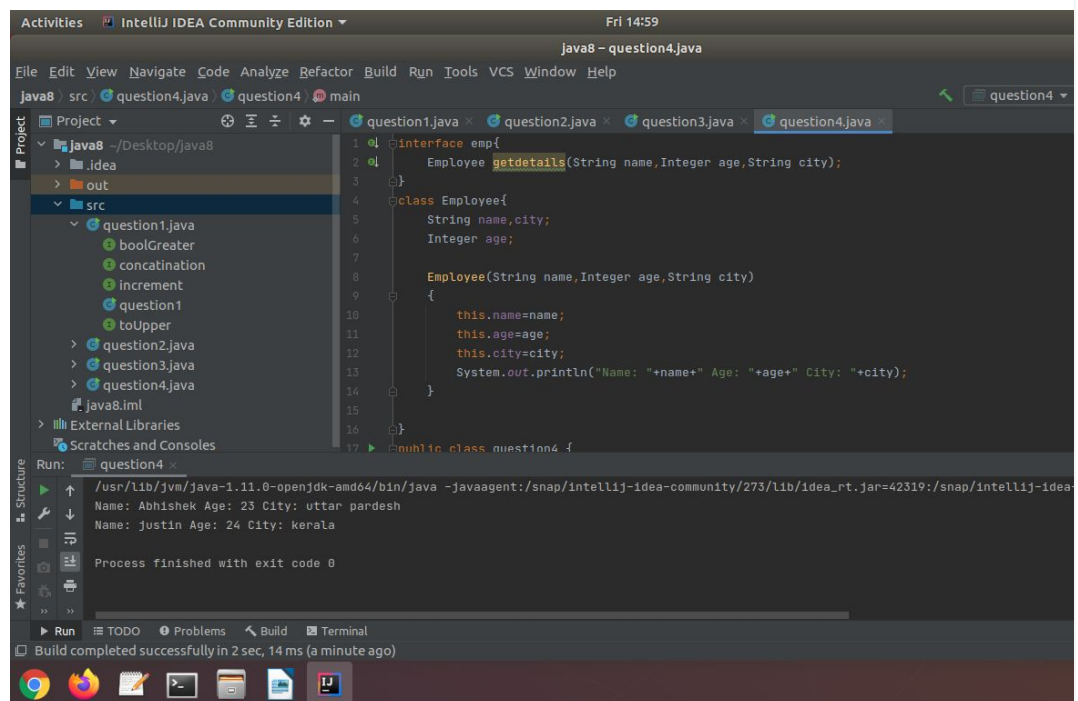
2. Create a functional interface whose method takes 2 integers and return one integer.



3. 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.

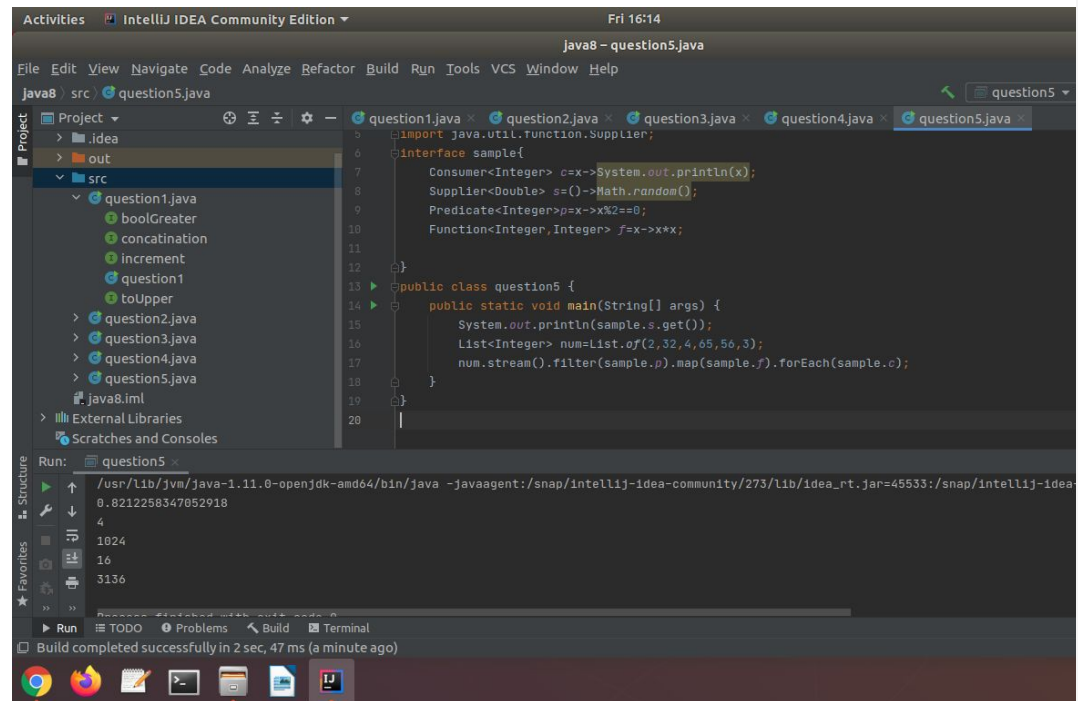


4. Create an Employee Class with instance variables (String) name, (Integer)age, (String)city and get the instance of the Class using constructor reference



5. Implement following functional interfaces from java.util.function using lambdas:

- (1) Consumer
- (2) Supplier
- (3) Predicate
- (4) Function



The screenshot shows the IntelliJ IDEA Community Edition interface. The main editor displays the code for `question5.java`. The code implements the `sample` interface with four methods: `Consumer<Integer>`, `Supplier<Double>`, `Predicate<Integer>`, and `Function<Integer, Integer>`. The `main` method uses these interfaces to demonstrate their usage with lambdas and streams.

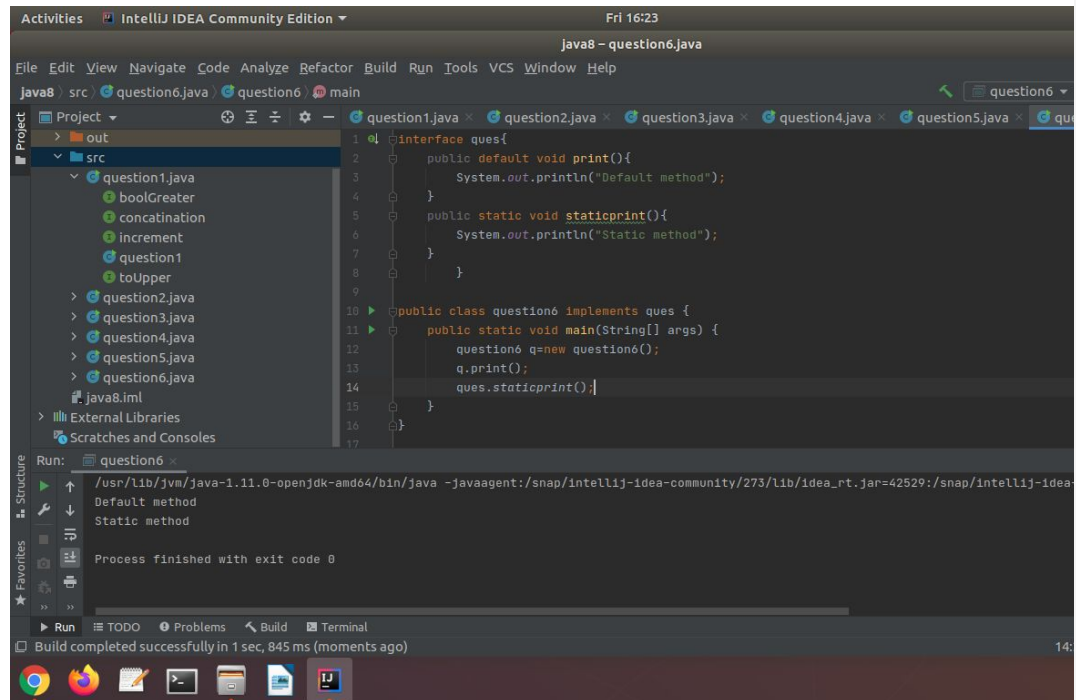
```
import java.util.function.Supplier;

interface sample {
    Consumer<Integer> c = x -> System.out.println(x);
    Supplier<Double> s = () -> Math.random();
    Predicate<Integer> p = x -> x % 2 == 0;
    Function<Integer, Integer> f = x -> x * x;
}

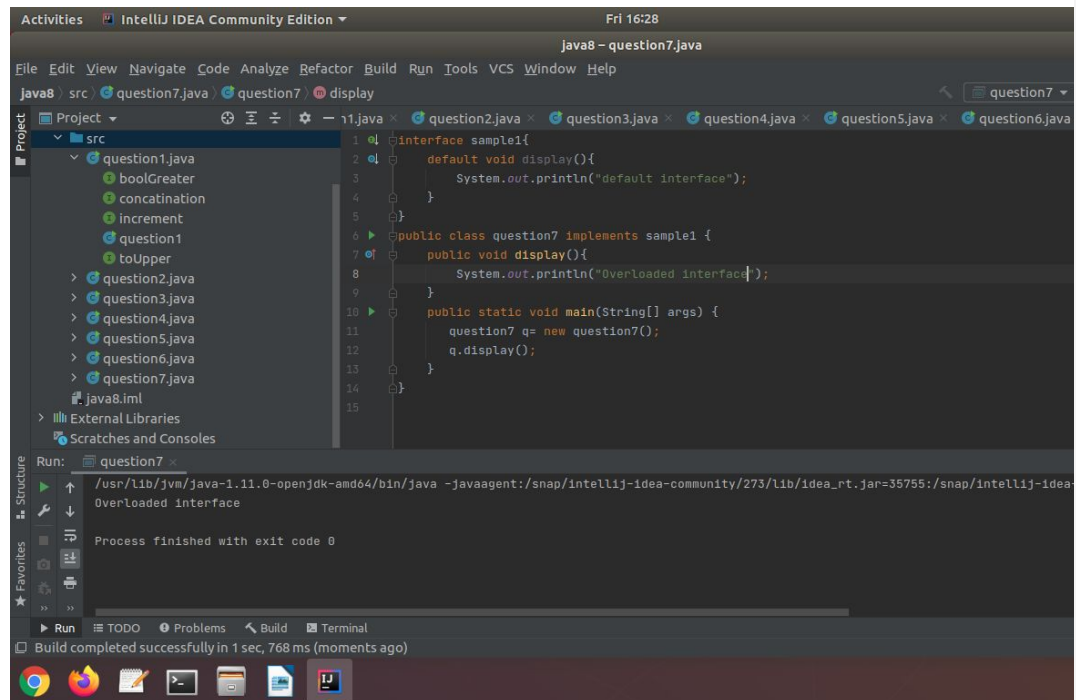
public class question5 {
    public static void main(String[] args) {
        System.out.println(sample.s.get());
        List<Integer> num = List.of(2, 32, 4, 65, 56, 3);
        num.stream().filter(sample.p).map(sample.f).forEach(sample.c);
    }
}
```

The Run tab at the bottom shows the execution output, indicating that the build completed successfully in 2 seconds and 47 milliseconds.

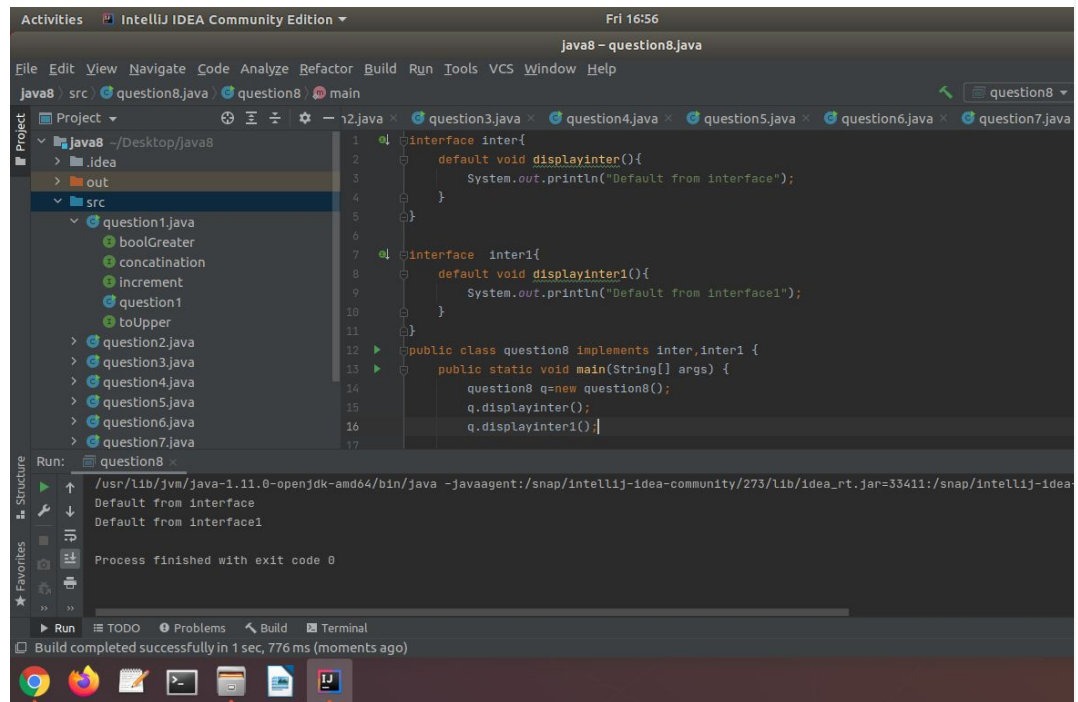
6. Create and access default and static method of an interface.



7. Override the default method of the interface.



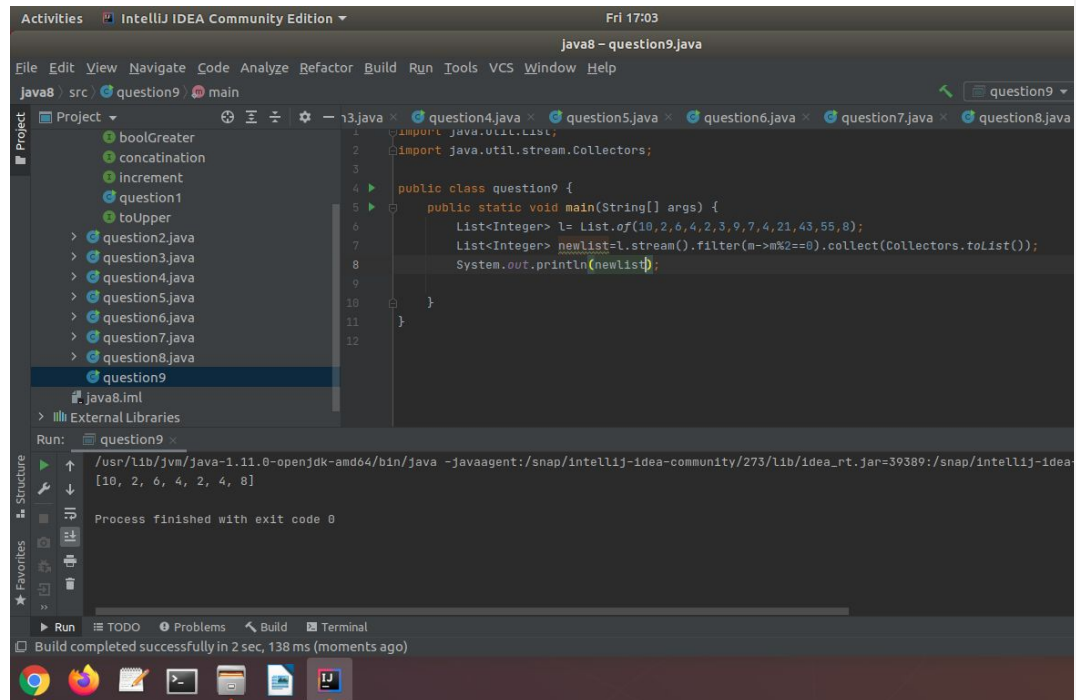
8. Implement multiple inheritance with default method inside interface.



The screenshot shows the IntelliJ IDEA Community Edition interface. The main editor displays the code for `question8.java`. The code defines two interfaces, `inter` and `inter1`, each with a default method `displayinter()`. The `inter` interface's default method prints "Default from interface", and the `inter1` interface's default method prints "Default from interface1". The `question8` class implements both interfaces and overrides the `displayinter()` method to print "Default from interface". The `main` method creates an instance of `question8` and calls `displayinter()`. The Run window shows the output: "Default from interface" and "Default from interface1". The status bar indicates the build completed successfully in 1 sec, 776 ms.

```
1 interface inter{
2     default void displayinter(){
3         System.out.println("Default from interface");
4     }
5 }
6
7 interface inter1{
8     default void displayinter1(){
9         System.out.println("Default from interface1");
10    }
11 }
12
13 public class question8 implements inter,inter1 {
14     public static void main(String[] args) {
15         question8 q=new question8();
16         q.displayinter();
17         q.displayinter1();
18     }
19 }
```

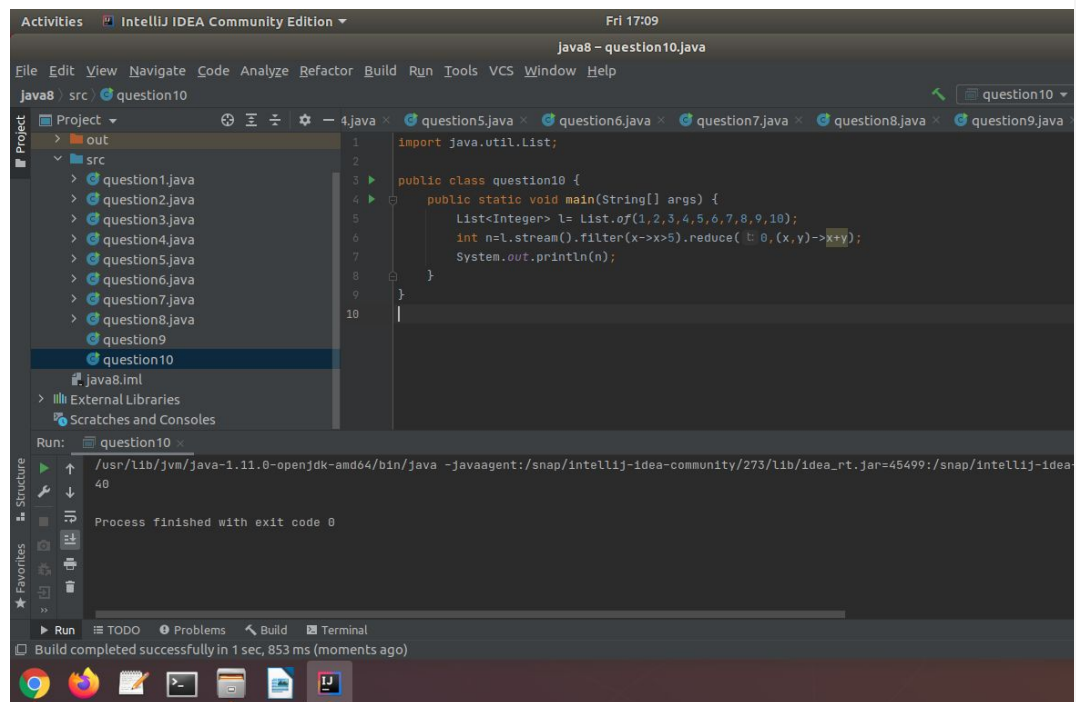
9. Collect all the even numbers from an integer list.



The screenshot shows the IntelliJ IDEA Community Edition interface. The main editor displays the code for `question9.java`. The code imports `java.util.List` and `java.util.stream.Collectors`. The `question9` class has a `main` method that creates a list of integers: `List.of(10,2,6,4,2,3,9,7,4,21,43,55,8)`. It then uses `stream().filter(m->m%2==0).collect(Collectors.toList())` to collect all even numbers from the list. The output of the program is `[10, 2, 6, 4, 2, 4, 8]`. The Run window shows the output: "[10, 2, 6, 4, 2, 4, 8]". The status bar indicates the build completed successfully in 2 sec, 138 ms.

```
1 import java.util.List;
2 import java.util.stream.Collectors;
3
4 public class question9 {
5     public static void main(String[] args) {
6         List<Integer> l= List.of(10,2,6,4,2,3,9,7,4,21,43,55,8);
7         List<Integer> newList=l.stream().filter(m->m%2==0).collect(Collectors.toList());
8         System.out.println(newList);
9     }
10 }
11
12 }
```


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



The screenshot shows the IntelliJ IDEA Community Edition interface. The project is named 'java8' and is located at 'src'. The file 'question10.java' is open in the editor. The code defines a class 'question10' with a 'main' method that creates a list of integers [1, 2, 3, 4, 5, 6, 7, 8, 9, 10], filters the elements greater than 5, and prints the sum of the remaining elements. The output is '40'. The run configuration is 'question10' and the process finished with exit code 0.

```
1 import java.util.List;
2
3 public class question10 {
4     public static void main(String[] args) {
5         List<Integer> l= List.of(1,2,3,4,5,6,7,8,9,10);
6         int n=l.stream().filter(x->x>5).reduce(0,(x,y)->x+y);
7         System.out.println(n);
8     }
9 }
10
```

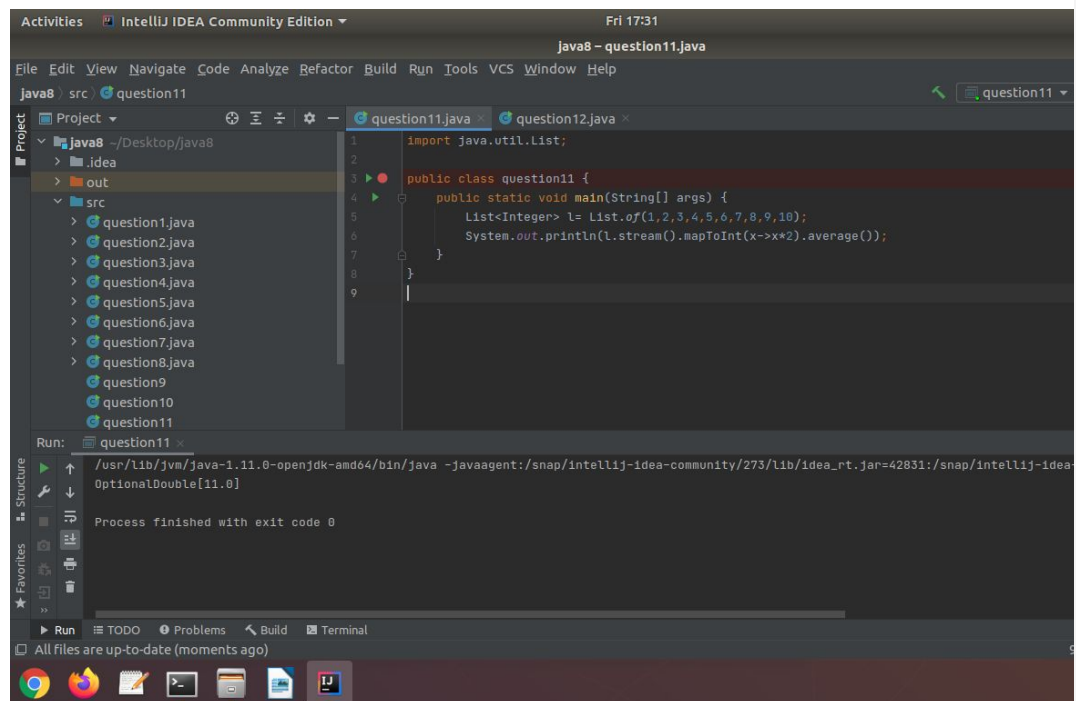
Run: question10

/usr/lib/jvm/java-1.11.0-openjdk-amd64/bin/java -javaagent:/snap/intellij-idea-community/273/lib/idea_rt.jar=45499:/snap/intellij-idea-40

Process finished with exit code 0

Build completed successfully in 1 sec, 853 ms (moments ago)

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



The screenshot shows the IntelliJ IDEA Community Edition interface. The project is named 'java8' and is located at 'src'. The file 'question11.java' is open in the editor. The code defines a class 'question11' with a 'main' method that creates a list of integers [1, 2, 3, 4, 5, 6, 7, 8, 9, 10], doubles each element, and prints the average of the resulting list. The output is 'OptionalDouble[11.0]'. The run configuration is 'question11' and the process finished with exit code 0.

```
1 import java.util.List;
2
3 public class question11 {
4     public static void main(String[] args) {
5         List<Integer> l= List.of(1,2,3,4,5,6,7,8,9,10);
6         System.out.println(l.stream().mapToInt(x->x*2).average());
7     }
8 }
9
```

Run: question11

/usr/lib/jvm/java-1.11.0-openjdk-amd64/bin/java -javaagent:/snap/intellij-idea-community/273/lib/idea_rt.jar=42831:/snap/intellij-idea-OptionalDouble[11.0]

Process finished with exit code 0

All files are up-to-date (moments ago)

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

