Lesson 02 Demo 03

Using Methods in Java

Objective: To depict how methods are implemented in Java

Tools required: Eclipse IDE

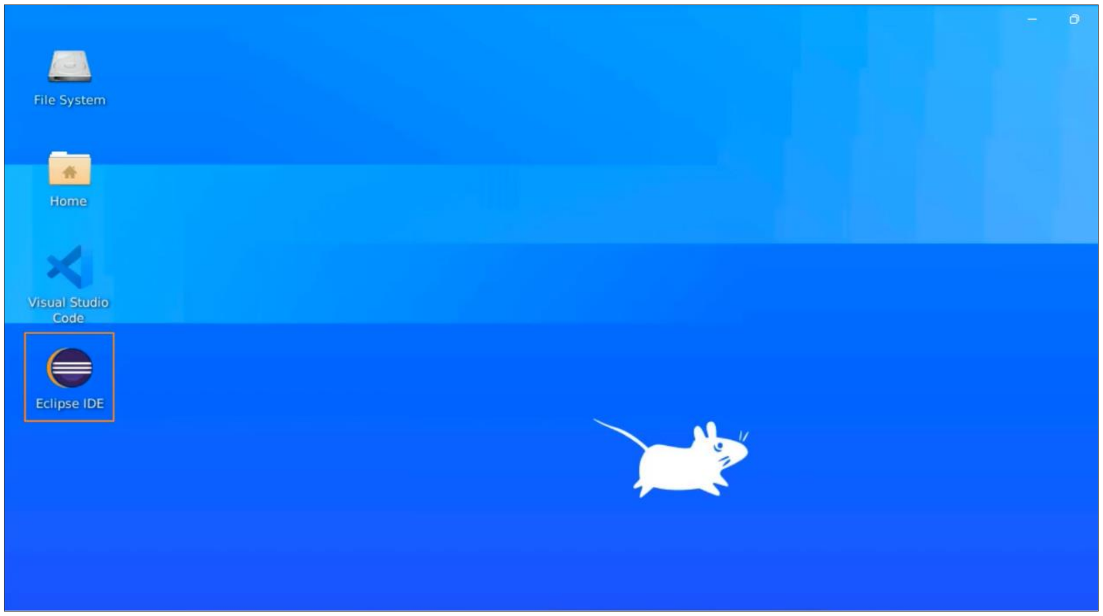
Prerequisites: None

Steps to be followed:

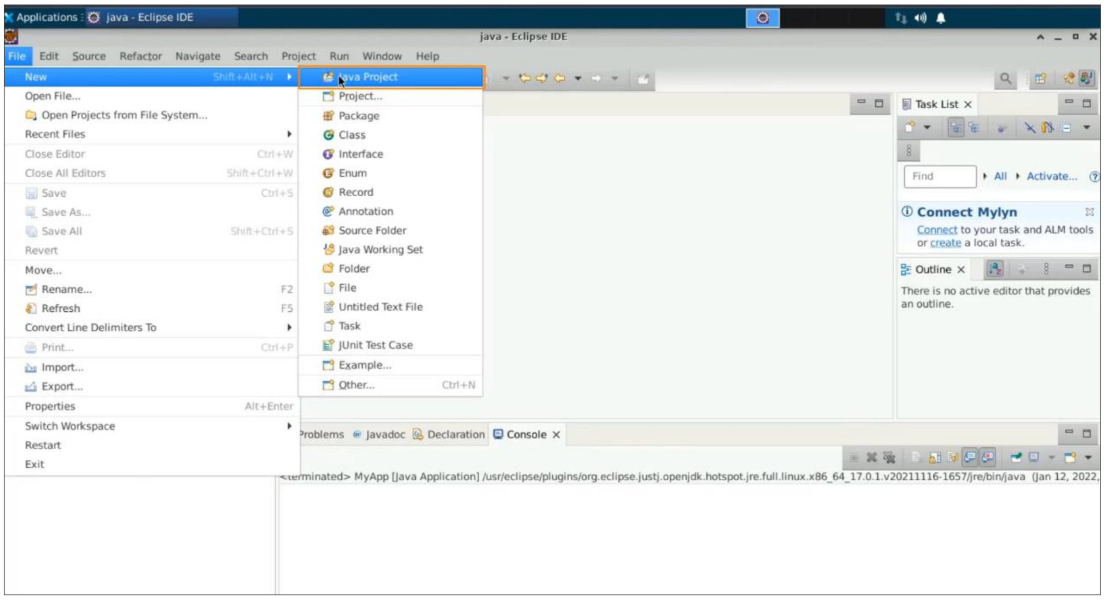
1. Write an algorithm and implement the same
2. Write arrays with cited examples
3. Run the code and get the output
4. Create and use a non-static method
5. Create an object with an object construction statement
6. Differentiate between running a static and a non-static method

Step 1: Write an algorithm and implement the same

1.1 Open the **Eclipse IDE**

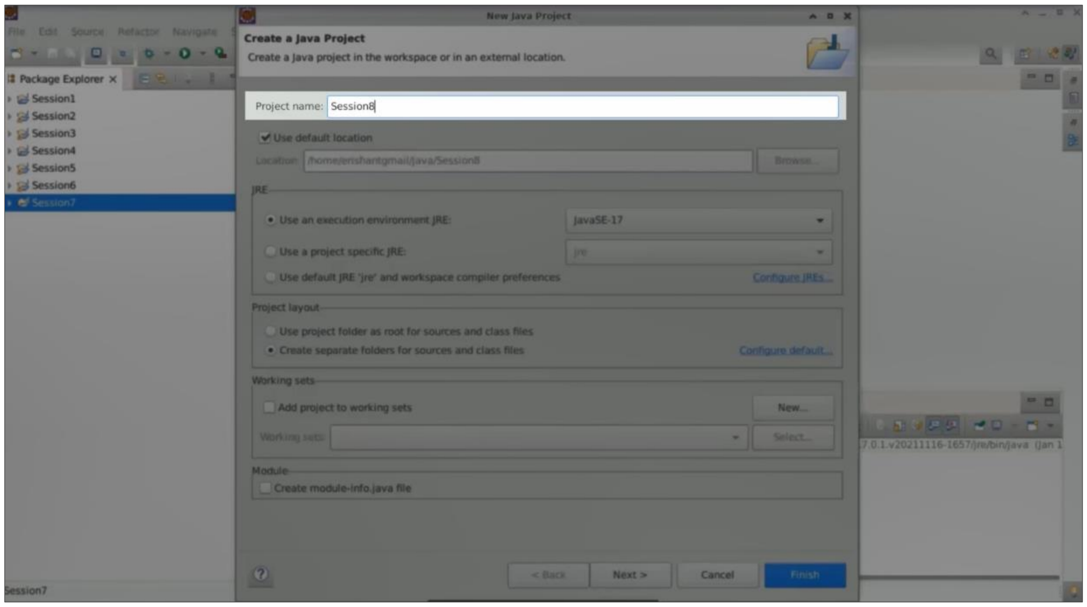


1.2 Select **File**, then **New,** and then **Java project**



```
<terminated> MyApp [Java Application] /usr/eclipse/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.linux.x86_64_17.0.1.v20211116-1657/jre/bin/java (Jan 12, 2022,
```

1.3 Name the project **"Session8",** uncheck **"Create a module info dot Java file"**, and press **Finish**



Create a Java Project

Create a Java project in the workspace or in an external location.

Project name: Session8

[✓] Use default location

Location: /home/enshantgmail/java/Session8

JRE

(●) Use an execution environment JRE: JavaSE-17

( ) Use a project specific JRE: jre

( ) Use default JRE 'jre' and workspace compiler preferences

Project layout

( ) Use project folder as root for sources and class files

(●) Create separate folders for sources and class files

Working sets

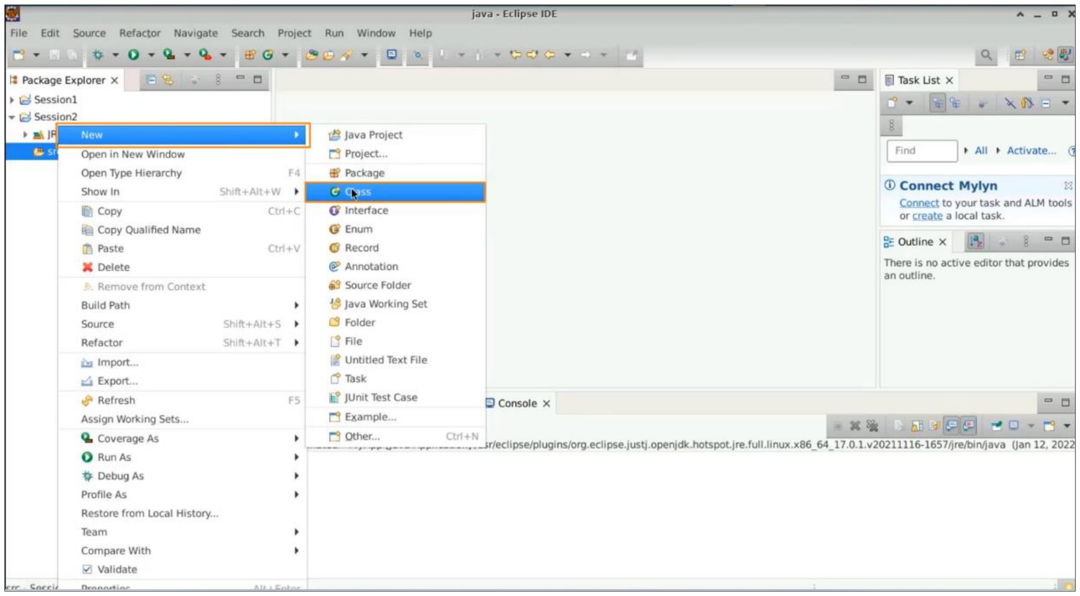
[ ] Add project to working sets

Working sets:

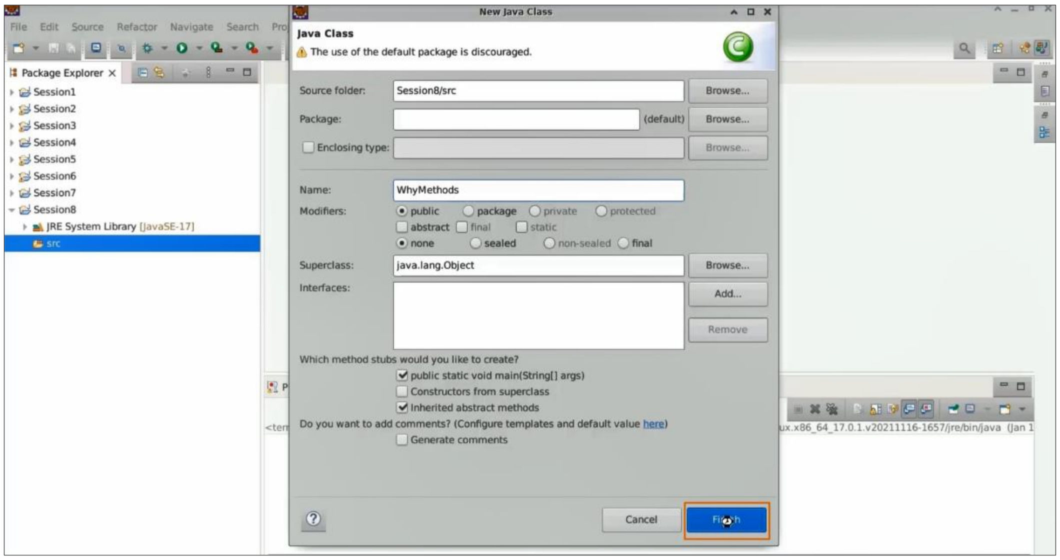
Module

[ ] Create module-info.java file

1.4 With a **Session8** on the src, do a right-click and create a **new class**



1.5 Name this class as **WhyMethods**, then select the **main method,** and then select **finish**


New Java Class

Java Class

The use of the default package is discouraged.

Source folder: Session8/src

Package: (default)

Enclosing type:

Name: WhyMethods

Modifiers: [✓ public] package private protected

abstract final static

[✓ none] sealed non-sealed final

Superclass: java.lang.Object

Interfaces:

Which method stubs would you like to create?

[✓] public static void main(String[] args)

[ ] Constructors from superclass

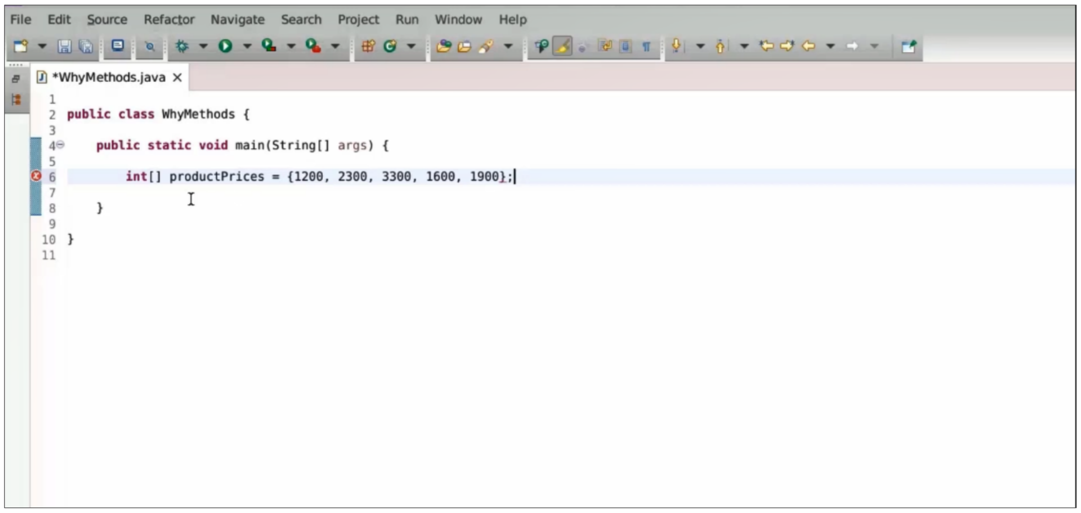
[✓] Inherited abstract methods

Do you want to add comments? (Configure templates and default value here)

[ ] Generate comments

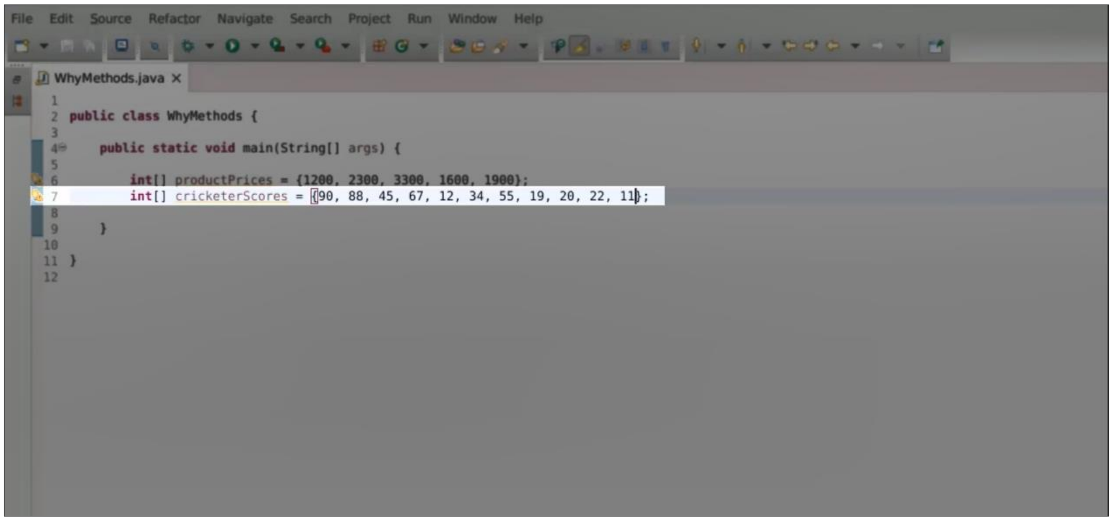
Step 2: Write arrays with cited examples

2.1 Write an array with product prices with five different elements in it


File Edit Source Refactor Navigate Search Project Run Window Help
*WhyMethods.java

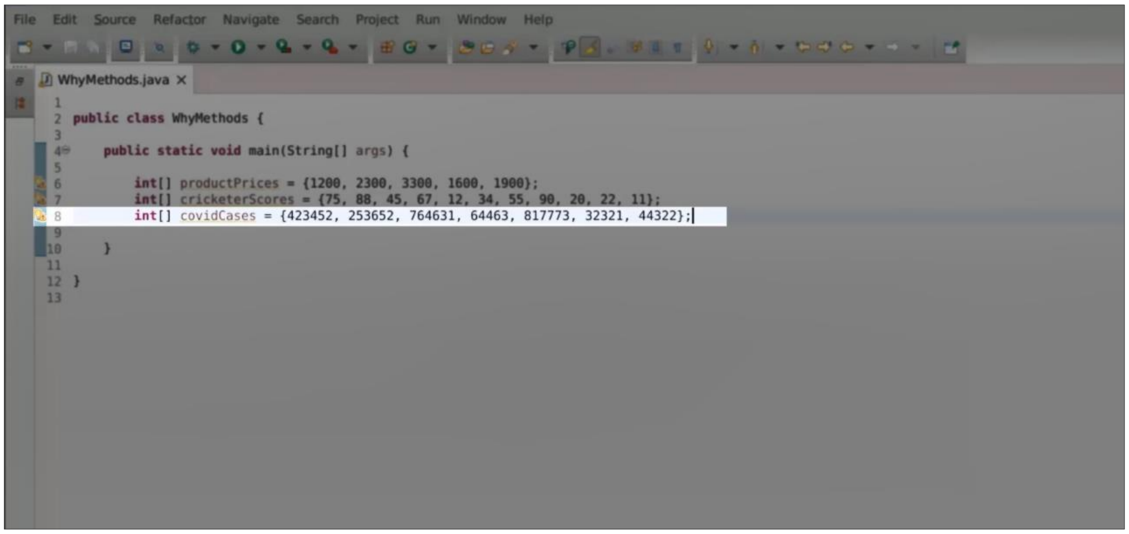
```java
public class WhyMethods {

    public static void main(String[] args) {

        int[] productPrices = {1200, 2300, 3300, 1600, 1900};

    }

}
```

2.2 Write cricketerScores with scores made by cricketers in a Test match


File Edit Source Refactor Navigate Search Project Run Window Help
WhyMethods.java

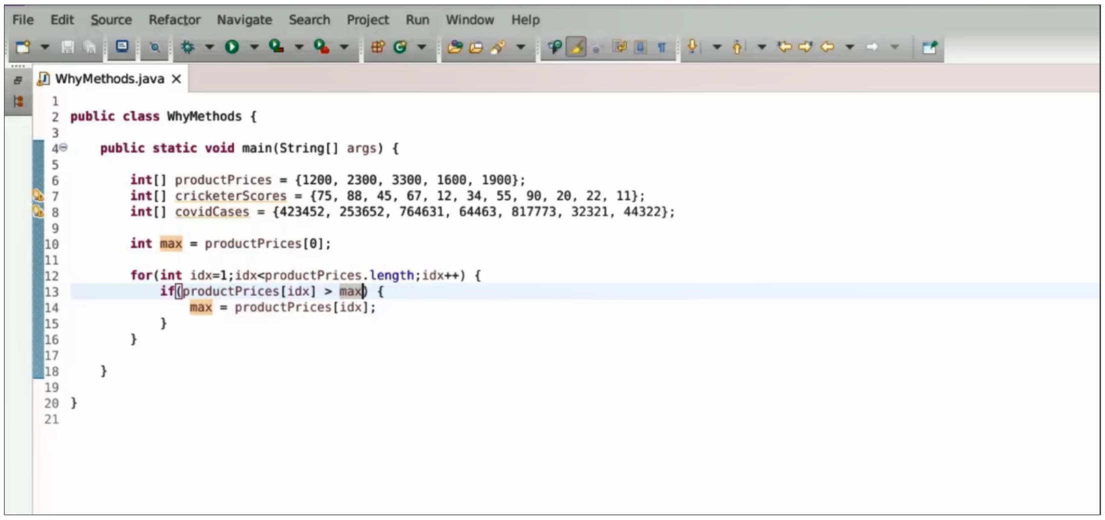
```java
public class WhyMethods {

    public static void main(String[] args) {

        int[] productPrices = {1200, 2300, 3300, 1600, 1900};
        int[] cricketerScores = {90, 88, 45, 67, 12, 34, 55, 19, 20, 22, 11};

    }

}
```

2.3 Now you have one more array, let us use covidCases for the world, and these are the active cases


WhyMethods.java

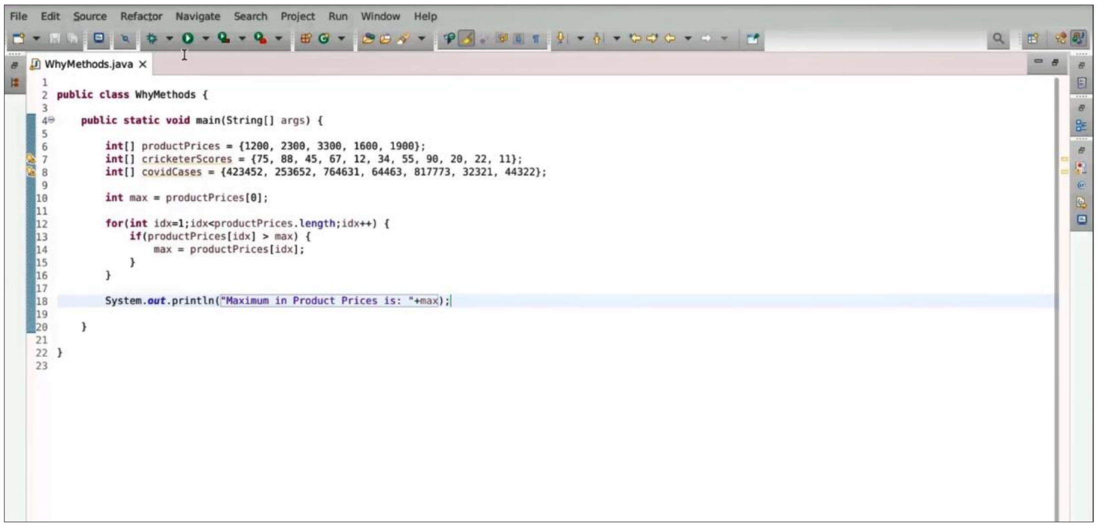
```
public class WhyMethods {

    public static void main(String[] args) {

        int[] productPrices = {1200, 2300, 3300, 1600, 1900};
        int[] cricketerScores = {75, 88, 45, 67, 12, 34, 55, 90, 20, 22, 11};
        int[] covidCases = {423452, 253652, 764631, 64463, 817773, 32321, 44322};

    }

}
```

2.4 Consider the maximum value in the product prices array to be the first element, which is at the zeroth index. You can assume that 1200 is the maximum value in this entire array. Now, start a loop that begins with the index at one and continues until the last element, which is **productPrices.length**. Iterate through the array one by one. If max is less than the product price at the current index, update max to be the product price at that index


WhyMethods.java

```
public class WhyMethods {

    public static void main(String[] args) {

        int[] productPrices = {1200, 2300, 3300, 1600, 1900};
        int[] cricketerScores = {75, 88, 45, 67, 12, 34, 55, 90, 20, 22, 11};
        int[] covidCases = {423452, 253652, 764631, 64463, 817773, 32321, 44322};

        int max = productPrices[0];

        for(int idx=1;idx<productPrices.length;idx++) {
            if(productPrices[idx] > max) {
                max = productPrices[idx];
            }
        }

    }

}
```

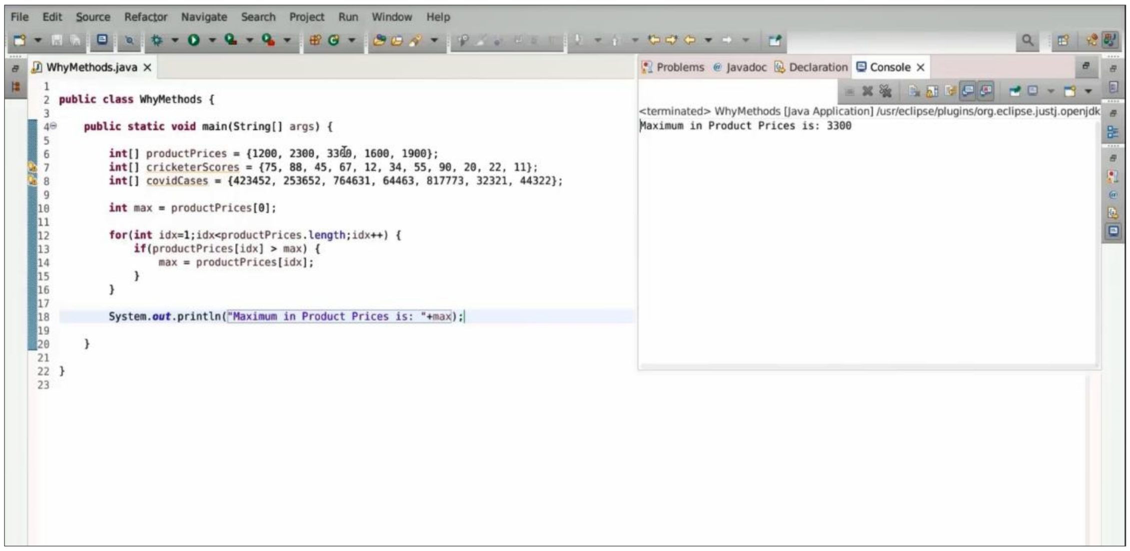
2.5 Print **"Maximum in product prices is "** followed by max. Here, you are getting the max value from the product prices array


File Edit Source Refactor Navigate Search Project Run Window Help
WhyMethods.java

```java
public class WhyMethods {

    public static void main(String[] args) {

        int[] productPrices = {1200, 2300, 3300, 1600, 1900};
        int[] cricketerScores = {75, 88, 45, 67, 12, 34, 55, 90, 20, 22, 11};
        int[] covidCases = {423452, 253652, 764631, 64463, 817773, 32321, 44322};

        int max = productPrices[0];

        for(int idx=1;idx<productPrices.length;idx++) {
            if(productPrices[idx] > max) {
                max = productPrices[idx];
            }
        }

        System.out.println("Maximum in Product Prices is: "+max);

    }

}
```

Step 3: Run the code and get the output

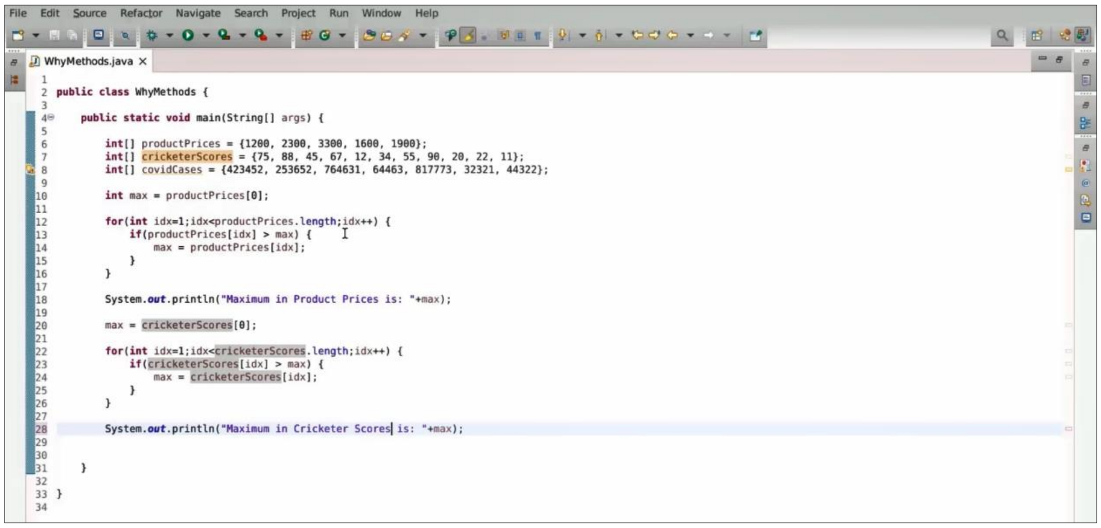
3.1 Run the code and see if it works fine or not, it states that the maximum is 3300 and that is the maximum value


File Edit Source Refactor Navigate Search Project Run Window Help
WhyMethods.java
Problems Javadoc Declaration Console

```java
public class WhyMethods {

    public static void main(String[] args) {

        int[] productPrices = {1200, 2300, 3300, 1600, 1900};
        int[] cricketerScores = {75, 88, 45, 67, 12, 34, 55, 90, 20, 22, 11};
        int[] covidCases = {423452, 253652, 764631, 64463, 817773, 32321, 44322};

        int max = productPrices[0];

        for(int idx=1;idx<productPrices.length;idx++) {
            if(productPrices[idx] > max) {
                max = productPrices[idx];
            }
        }

        System.out.println("Maximum in Product Prices is: "+max);

    }

}
```

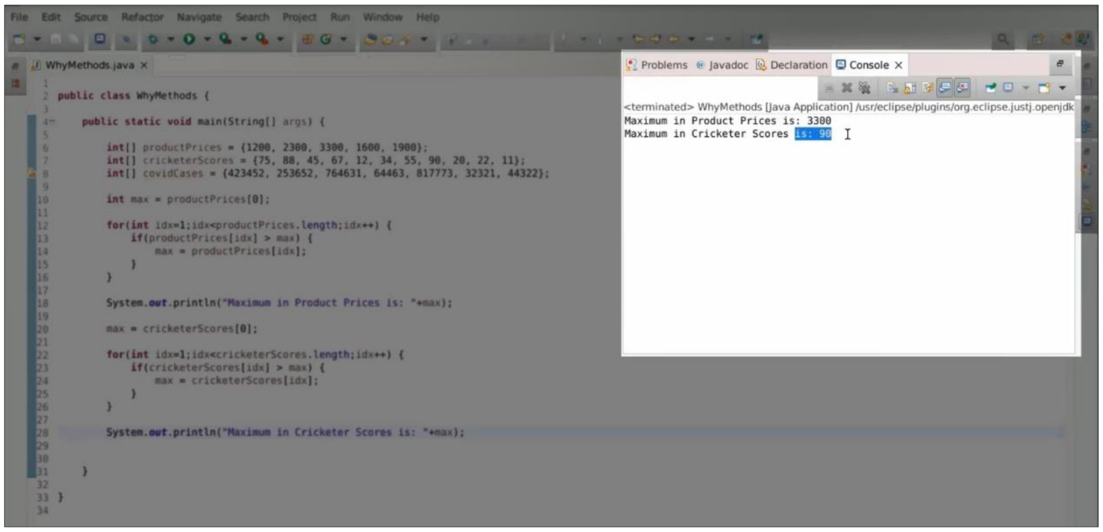
```
<terminated> WhyMethods [Java Application] /usr/eclipse/plugins/org.eclipse.justj.openjdk
Maximum in Product Prices is: 3300
```

3.2 Repeat the same algorithm by copying and pasting the previous code. Replace **productPrices** with **cricketerScores**.



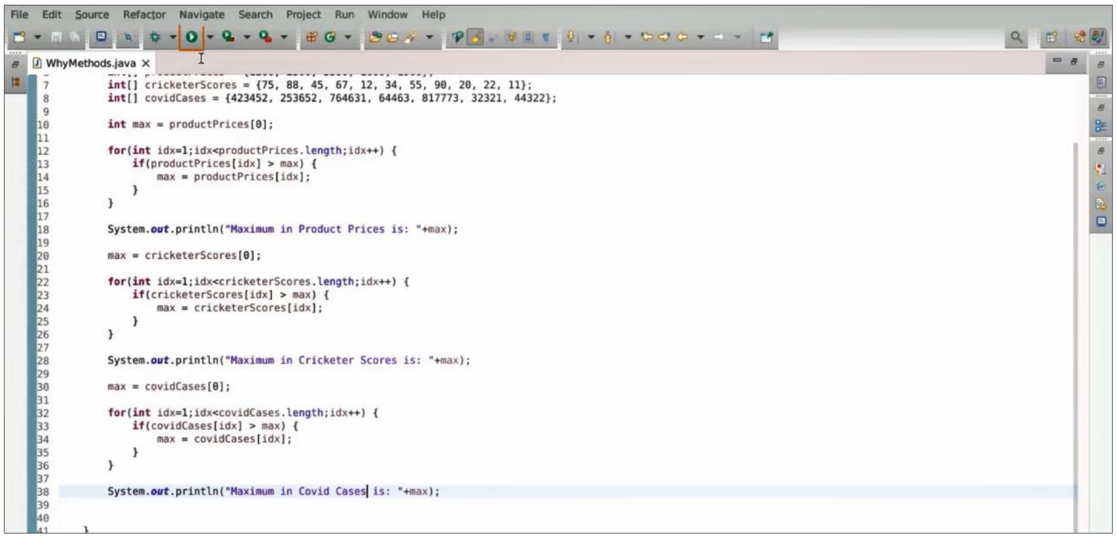
```
public class WhyMethods {

    public static void main(String[] args) {

        int[] productPrices = {1200, 2300, 3300, 1600, 1900};
        int[] cricketerScores = {75, 88, 45, 67, 12, 34, 55, 90, 20, 22, 11};
        int[] covidCases = {423452, 253652, 764631, 64463, 817773, 32321, 44322};

        int max = productPrices[0];

        for(int idx=1;idx<productPrices.length;idx++) {
            if(productPrices[idx] > max) {
                max = productPrices[idx];
            }
        }

        System.out.println("Maximum in Product Prices is: "+max);

        max = cricketerScores[0];

        for(int idx=1;idx<cricketerScores.length;idx++) {
            if(cricketerScores[idx] > max) {
                max = cricketerScores[idx];
            }
        }

        System.out.println("Maximum in Cricketer Scores is: "+max);


    }
}
```

3.3 Re-run the code, and here you are with the value of Max as 90, which is the highest value



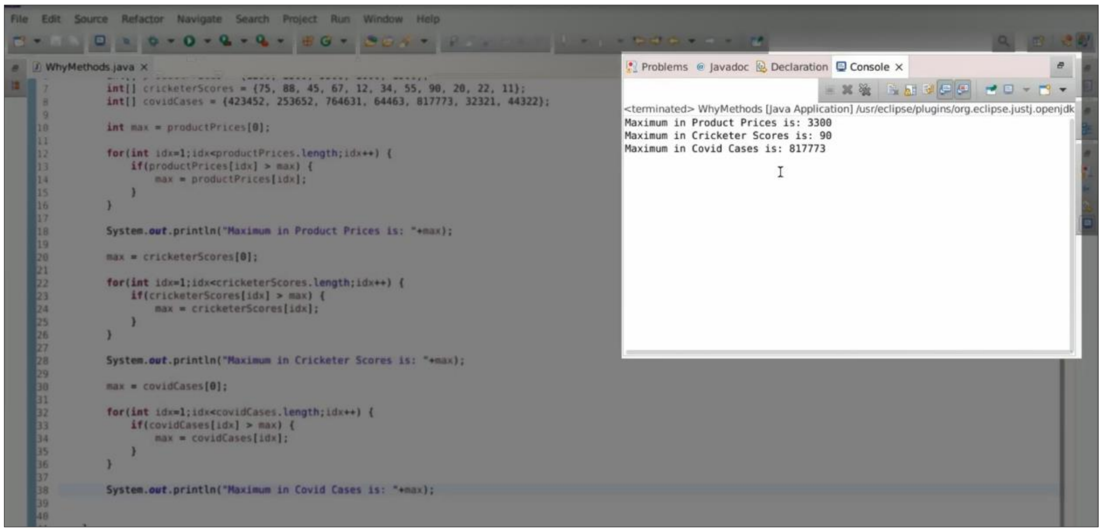
```
<terminated> WhyMethods [Java Application] /usr/eclipse/plugins/org.eclipse.justj.openjdk
Maximum in Product Prices is: 3300
Maximum in Cricketer Scores is: 90
```

3.4 Repeat the same algorithm by copy-pasting the previous code. Replace the **cricketerScores** with **covidCases**



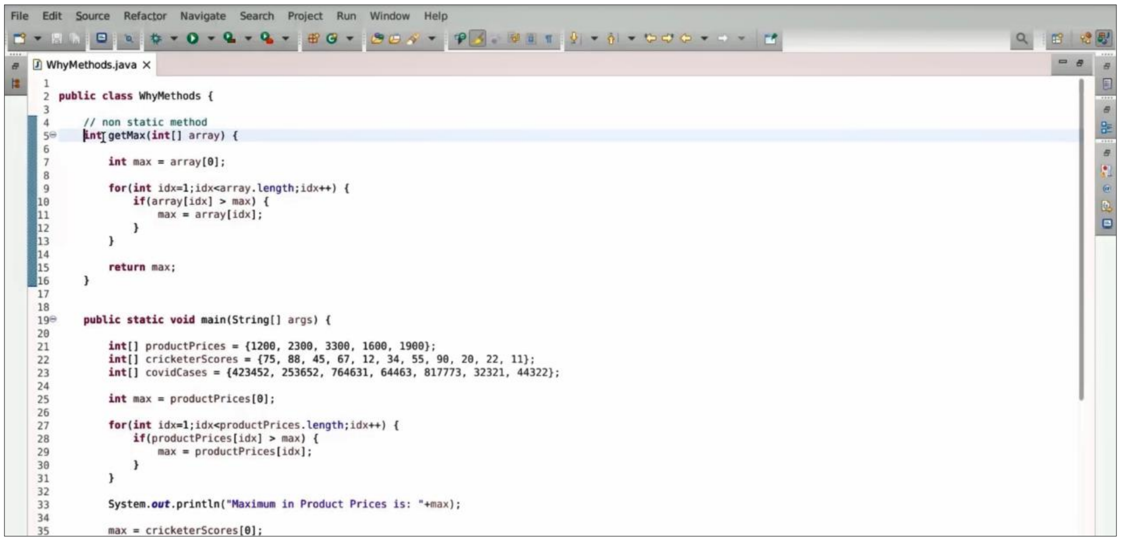
```java
        int[] cricketerScores = {75, 88, 45, 67, 12, 34, 55, 90, 20, 22, 11};
        int[] covidCases = {423452, 253652, 764631, 64463, 817773, 32321, 44322};

        int max = productPrices[0];

        for(int idx=1;idx<productPrices.length;idx++) {
            if(productPrices[idx] > max) {
                max = productPrices[idx];
            }
        }

        System.out.println("Maximum in Product Prices is: "+max);

        max = cricketerScores[0];

        for(int idx=1;idx<cricketerScores.length;idx++) {
            if(cricketerScores[idx] > max) {
                max = cricketerScores[idx];
            }
        }

        System.out.println("Maximum in Cricketer Scores is: "+max);

        max = covidCases[0];

        for(int idx=1;idx<covidCases.length;idx++) {
            if(covidCases[idx] > max) {
                max = covidCases[idx];
            }
        }

        System.out.println("Maximum in Covid Cases is: "+max);


    }
```

3.5 Run this code, here you are with the maximum in the covidCases with the value 817773



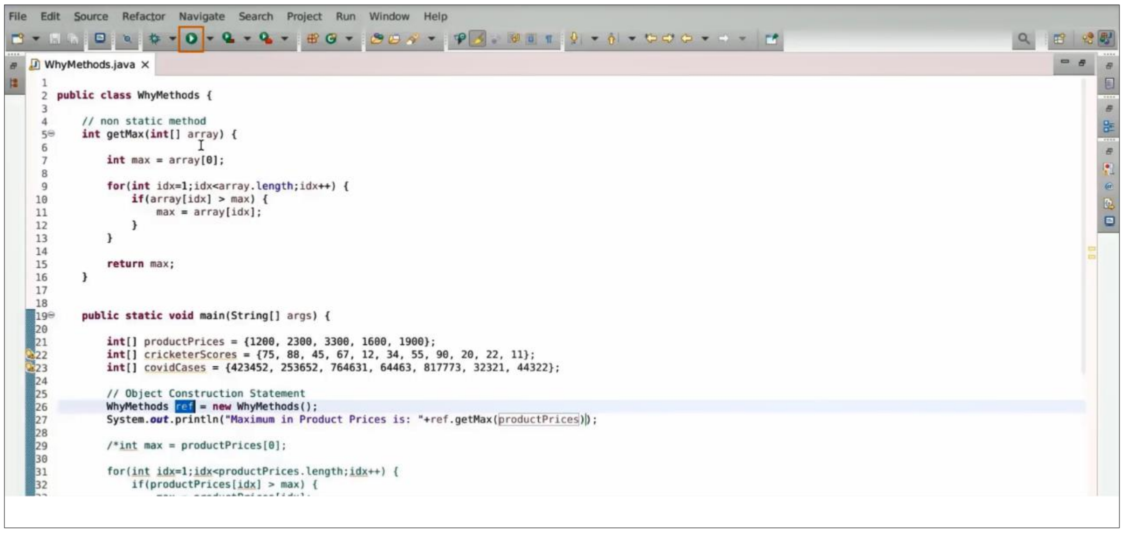
```
<terminated> WhyMethods [Java Application] /usr/eclipse/plugins/org.eclipse.justj.openjdk
Maximum in Product Prices is: 3300
Maximum in Cricketer Scores is: 90
Maximum in Covid Cases is: 817773
```

3.6 When a certain task is repeatedly executed, why waste time writing the same code again? This is where methods are useful to save development time. In the same class, create a method with an integer return type, named getMax, and take one array as input. Then, run this entire algorithm on the array instead of on the product prices. Finally, return the maximum value. This is known as a non-static method


File Edit Source Refactor Navigate Search Project Run Window Help
WhyMethods.java

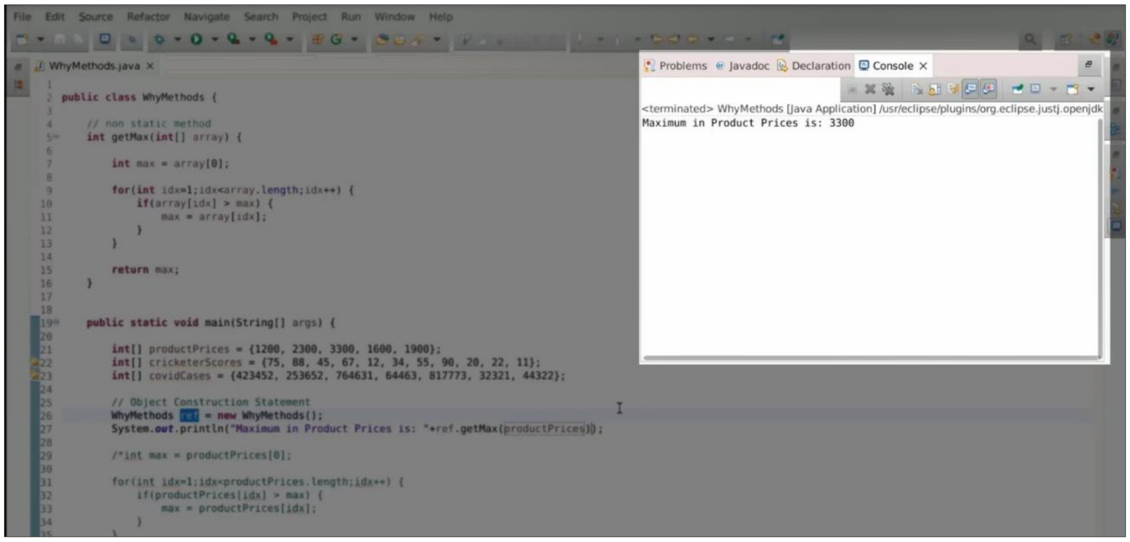
```java

public class WhyMethods {

    // non static method
    int getMax(int[] array) {

        int max = array[0];

        for(int idx=1;idx<array.length;idx++) {
            if(array[idx] > max) {
                max = array[idx];
            }
        }

        return max;
    }


    public static void main(String[] args) {

        int[] productPrices = {1200, 2300, 3300, 1600, 1900};
        int[] cricketerScores = {75, 88, 45, 67, 12, 34, 55, 90, 20, 22, 11};
        int[] covidCases = {423452, 253652, 764631, 64463, 817773, 32321, 44322};

        int max = productPrices[0];

        for(int idx=1;idx<productPrices.length;idx++) {
            if(productPrices[idx] > max) {
                max = productPrices[idx];
            }
        }

        System.out.println("Maximum in Product Prices is: "+max);

        max = cricketerScores[0];
```

3.7 To use this method, comment out the entire previous code. Now, create an object of the class. Write **WhyMethods** and create a reference variable. Use the new operator, followed by your class name with parentheses; this is known as the object construction statement. Write **System.out.println("Maximum in product prices is " + referenceVariable.getMax(productPrices));**. This will execute the **getMax** method using the reference variable and pass the **productPrices** array


File Edit Source Refactor Navigate Search Project Run Window Help
WhyMethods.java

```java

public class WhyMethods {

    // non static method
    int getMax(int[] array) {

        int max = array[0];

        for(int idx=1;idx<array.length;idx++) {
            if(array[idx] > max) {
                max = array[idx];
            }
        }

        return max;
    }


    public static void main(String[] args) {

        int[] productPrices = {1200, 2300, 3300, 1600, 1900};
        int[] cricketerScores = {75, 88, 45, 67, 12, 34, 55, 90, 20, 22, 11};
        int[] covidCases = {423452, 253652, 764631, 64463, 817773, 32321, 44322};

        // Object Construction Statement
        WhyMethods ref = new WhyMethods();
        System.out.println("Maximum in Product Prices is: "+ref.getMax(productPrices));

        /*int max = productPrices[0];

        for(int idx=1;idx<productPrices.length;idx++) {
            if(productPrices[idx] > max) {
```

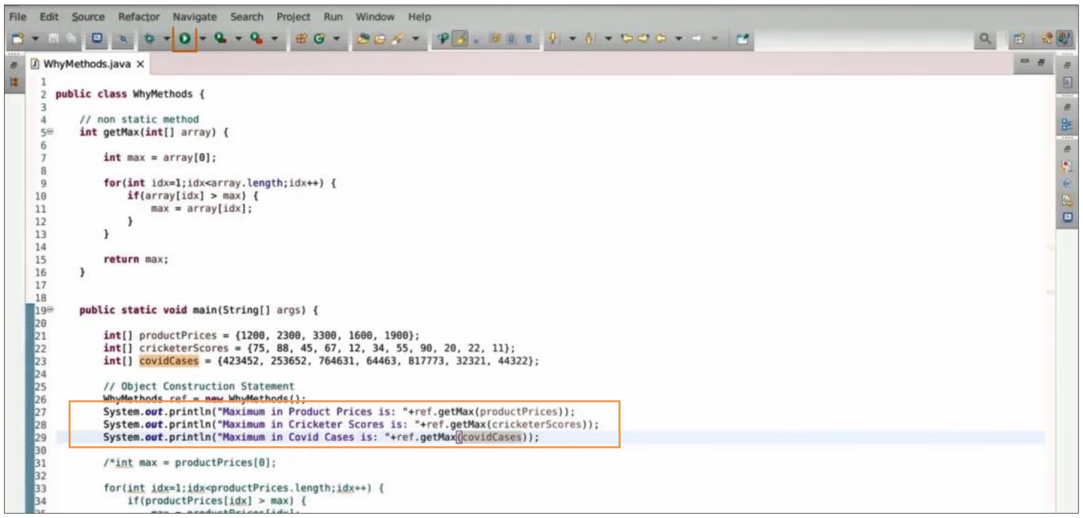
3.8 Run this code, and now you can see the maximum in product prices is 3300



```java

public class WhyMethods {

    // non static method
    int getMax(int[] array) {

        int max = array[0];

        for(int idx=1;idx<array.length;idx++) {
            if(array[idx] > max) {
                max = array[idx];
            }
        }

        return max;
    }


    public static void main(String[] args) {

        int[] productPrices = {1200, 2300, 3300, 1600, 1900};
        int[] cricketerScores = {75, 88, 45, 67, 12, 34, 55, 90, 20, 22, 11};
        int[] covidCases = {423452, 253652, 764631, 64463, 817773, 32321, 44322};

        // Object Construction Statement
        WhyMethods ref = new WhyMethods();
        System.out.println("Maximum in Product Prices is: "+ref.getMax(productPrices));

        /*int max = productPrices[0];

        for(int idx=1;idx<productPrices.length;idx++) {
            if(productPrices[idx] > max) {
                max = productPrices[idx];
            }
```

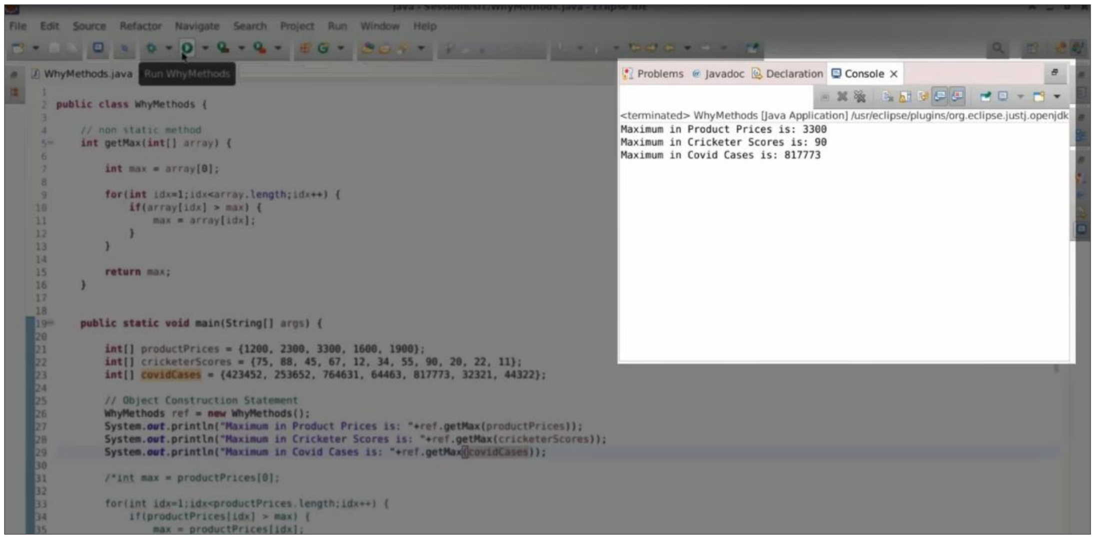
```
<terminated> WhyMethods [Java Application] /usr/eclipse/plugins/org.eclipse.justj.openjdk
Maximum in Product Prices is: 3300
```

3.9 Now you will execute the same method repeatedly, rather than writing the same code repeatedly, and you can even use the same method on different arrays to get the maximum out of it



```java

public class WhyMethods {

    // non static method
    int getMax(int[] array) {

        int max = array[0];

        for(int idx=1;idx<array.length;idx++) {
            if(array[idx] > max) {
                max = array[idx];
            }
        }

        return max;
    }


    public static void main(String[] args) {

        int[] productPrices = {1200, 2300, 3300, 1600, 1900};
        int[] cricketerScores = {75, 88, 45, 67, 12, 34, 55, 90, 20, 22, 11};
        int[] covidCases = {423452, 253652, 764631, 64463, 817773, 32321, 44322};

        // Object Construction Statement
        WhyMethods ref = new WhyMethods();
        System.out.println("Maximum in Product Prices is: "+ref.getMax(productPrices));
        System.out.println("Maximum in Cricketer Scores is: "+ref.getMax(cricketerScores));
        System.out.println("Maximum in Covid Cases is: "+ref.getMax(covidCases));

        /*int max = productPrices[0];

        for(int idx=1;idx<productPrices.length;idx++) {
            if(productPrices[idx] > max) {
```

3.10 Run the code and you will get the same output

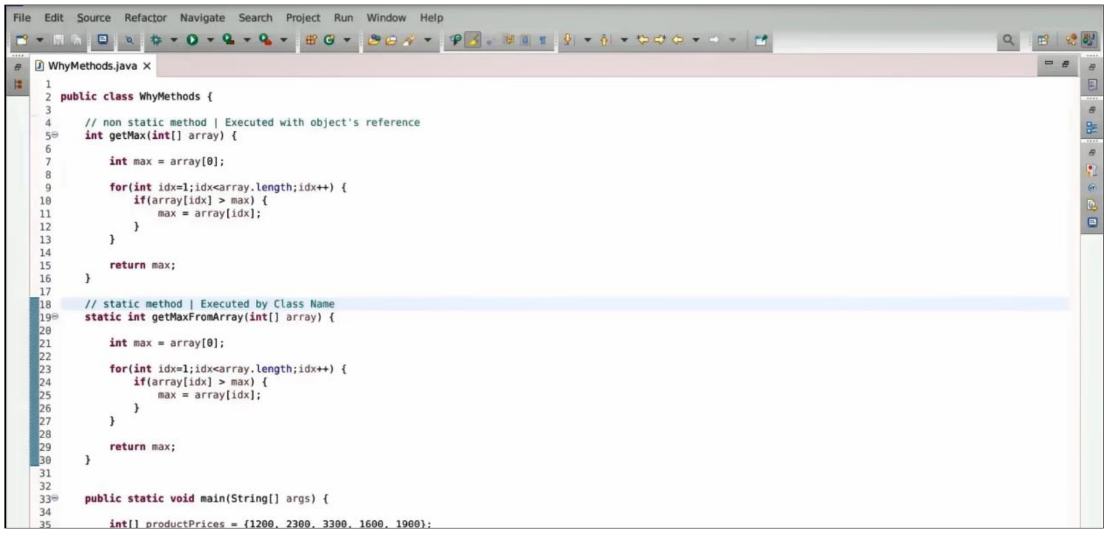


```
public class WhyMethods {

    // non static method
    int getMax(int[] array) {

        int max = array[0];

        for(int idx=1;idx<array.length;idx++) {
            if(array[idx] > max) {
                max = array[idx];
            }
        }

        return max;
    }


    public static void main(String[] args) {

        int[] productPrices = {1200, 2300, 3300, 1600, 1900};
        int[] cricketerScores = {75, 88, 45, 67, 12, 34, 55, 90, 20, 22, 11};
        int[] covidCases = {423452, 253652, 764631, 64463, 817773, 32321, 44322};

        // Object Construction Statement
        WhyMethods ref = new WhyMethods();
        System.out.println("Maximum in Product Prices is: "+ref.getMax(productPrices));
        System.out.println("Maximum in Cricketer Scores is: "+ref.getMax(cricketerScores));
        System.out.println("Maximum in Covid Cases is: "+ref.getMax(covidCases));

        /*int max = productPrices[0];

        for(int idx=1;idx<productPrices.length;idx++) {
            if(productPrices[idx] > max) {
                max = productPrices[idx];
```

```
<terminated> WhyMethods [Java Application] /usr/eclipse/plugins/org.eclipse.justj.openjdk
Maximum in Product Prices is: 3300
Maximum in Cricketer Scores is: 90
Maximum in Covid Cases is: 817773
```

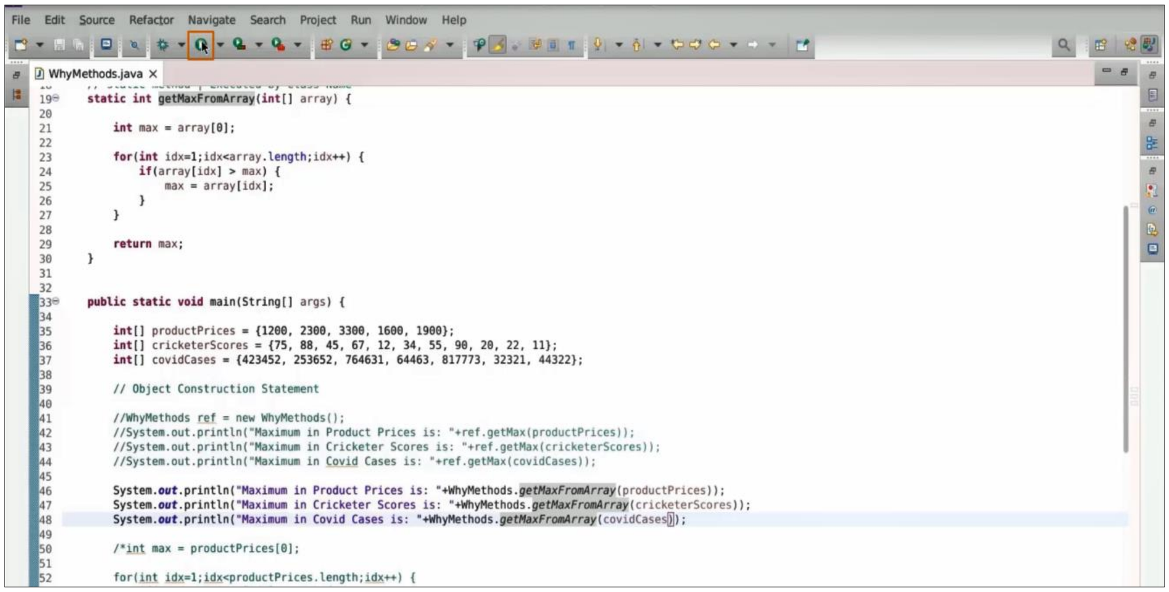
Step 4: Create and use a non-static method

4.1 The same method can also be rewritten, write **getMaxFromArray**, here you have changed the name since you cannot have two methods with the This time you can add a static keyword in front of it. So this method which is marked as static is known as a static method



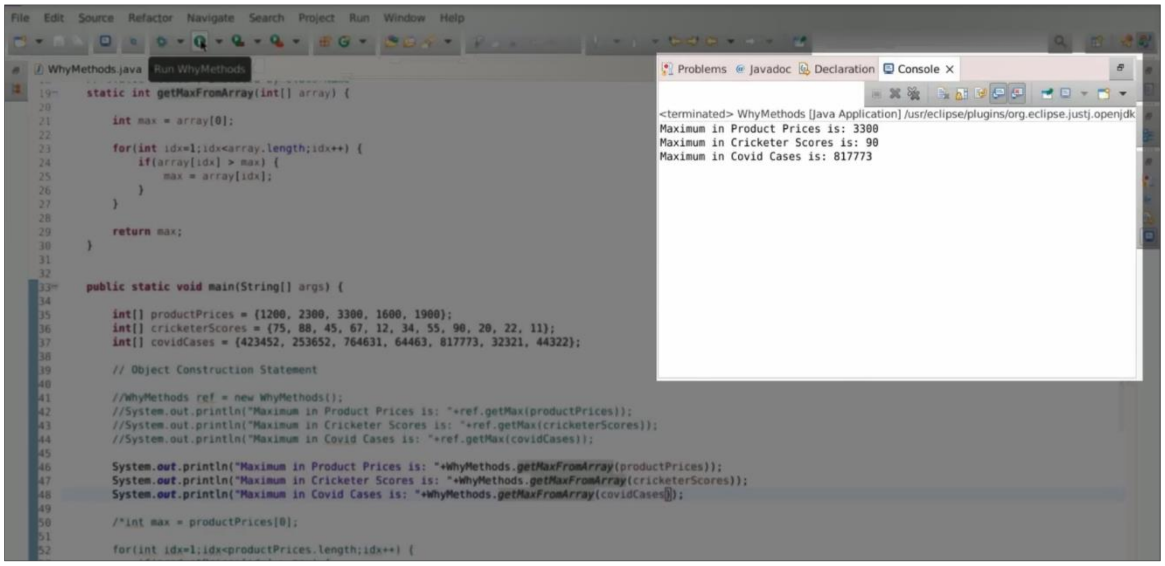
```
public class WhyMethods {

    // non static method | Executed with object's reference
    int getMax(int[] array) {

        int max = array[0];

        for(int idx=1;idx<array.length;idx++) {
            if(array[idx] > max) {
                max = array[idx];
            }
        }

        return max;
    }

    // static method | Executed by Class Name
    static int getMaxFromArray(int[] array) {

        int max = array[0];

        for(int idx=1;idx<array.length;idx++) {
            if(array[idx] > max) {
                max = array[idx];
            }
        }

        return max;
    }


    public static void main(String[] args) {

        int[] productPrices = {1200, 2300, 3300, 1600, 1900};
```

4.2 Now, instead of using the object's reference, write the class name followed by **.getMaxFromArray**



```java
    static int getMaxFromArray(int[] array) {

        int max = array[0];

        for(int idx=1;idx<array.length;idx++) {
            if(array[idx] > max) {
                max = array[idx];
            }
        }

        return max;
    }


    public static void main(String[] args) {

        int[] productPrices = {1200, 2300, 3300, 1600, 1900};
        int[] cricketerScores = {75, 88, 45, 67, 12, 34, 55, 90, 20, 22, 11};
        int[] covidCases = {423452, 253652, 764631, 64463, 817773, 32321, 44322};

        // Object Construction Statement

        //WhyMethods ref = new WhyMethods();
        //System.out.println("Maximum in Product Prices is: "+ref.getMax(productPrices));
        //System.out.println("Maximum in Cricketer Scores is: "+ref.getMax(cricketerScores));
        //System.out.println("Maximum in Covid Cases is: "+ref.getMax(covidCases));

        System.out.println("Maximum in Product Prices is: "+WhyMethods.getMaxFromArray(productPrices));
        System.out.println("Maximum in Cricketer Scores is: "+WhyMethods.getMaxFromArray(cricketerScores));
        System.out.println("Maximum in Covid Cases is: "+WhyMethods.getMaxFromArray(covidCases));

        /*int max = productPrices[0];

        for(int idx=1;idx<productPrices.length;idx++) {
```

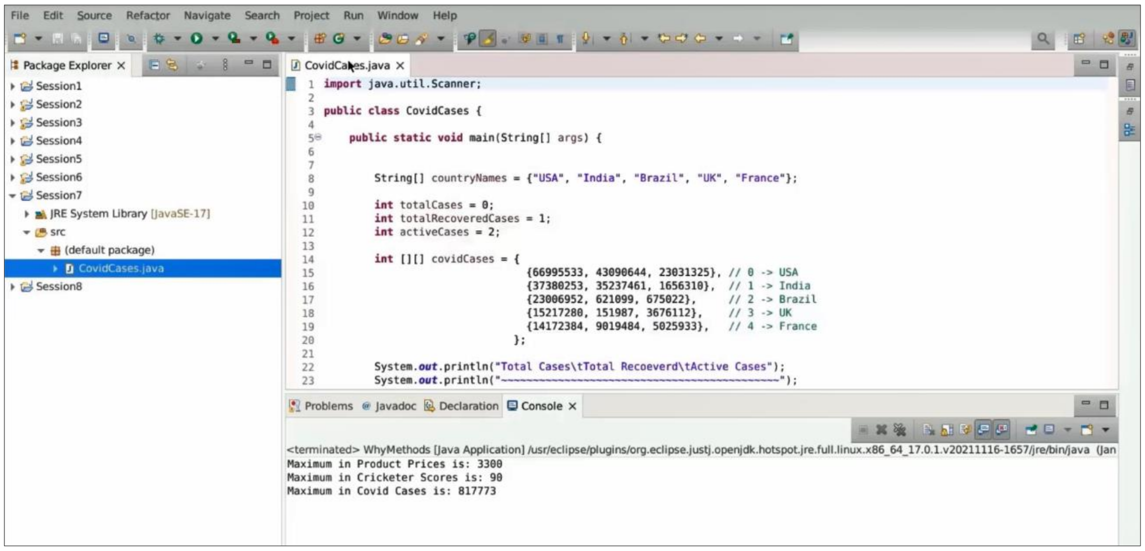
4.3 Run the code and you will get the same output



```
<terminated> WhyMethods [Java Application] /usr/eclipse/plugins/org.eclipse.justj.openjdk
Maximum in Product Prices is: 3300
Maximum in Cricketer Scores is: 90
Maximum in Covid Cases is: 817773
```

Step 5: Create an object with an object construction statement

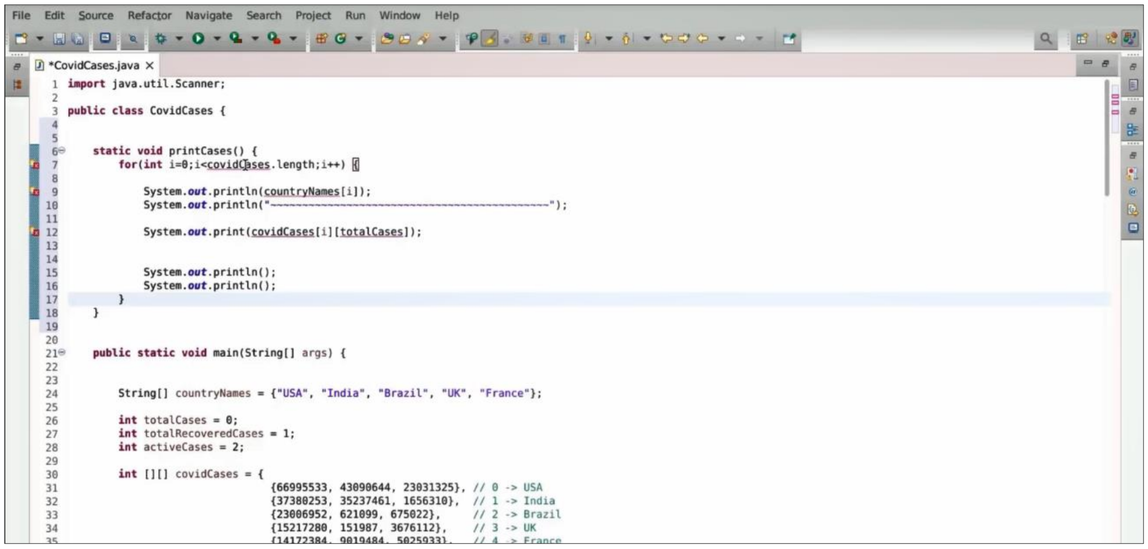
5.1 Return back to your Package explorer, open the Session number 7 folder, and here select the CovidCases dot java file. In this file, you can see that some statements are repeatedly used



```
import java.util.Scanner;

public class CovidCases {

    public static void main(String[] args) {


        String[] countryNames = {"USA", "India", "Brazil", "UK", "France"};

        int totalCases = 0;
        int totalRecoveredCases = 1;
        int activeCases = 2;

        int [][] covidCases = {
                        {66995533, 43090644, 23031325}, // 0 -> USA
                        {37380253, 35237461, 1656310},  // 1 -> India
                        {23006952, 621099, 675022},     // 2 -> Brazil
                        {15217280, 151987, 3676112},    // 3 -> UK
                        {14172384, 9019484, 5025933},   // 4 -> France
                    };

        System.out.println("Total Cases\tTotal Recoeverd\tActive Cases");
        System.out.println("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~");
```

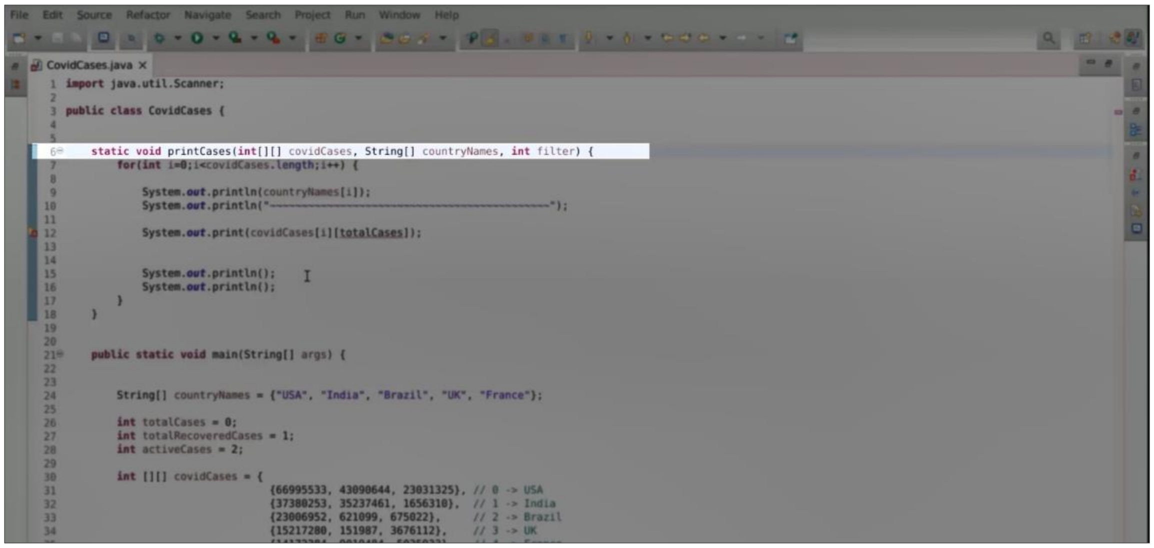
```
<terminated> WhyMethods [Java Application] /usr/eclipse/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.linux.x86_64_17.0.1.v20211116-1657/jre/bin/java (Jan
Maximum in Product Prices is: 3300
Maximum in Cricketer Scores is: 90
Maximum in Covid Cases is: 817773
```

5.2 Select the piece of logic which is printing the data, copy this, and inside your class, write Static void print cases, and paste this logic here:



```
import java.util.Scanner;

public class CovidCases {


    static void printCases() {
        for(int i=0;i<covidCases.length;i++) {

            System.out.println(countryNames[i]);
            System.out.println("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~");

            System.out.print(covidCases[i][totalCases]);


            System.out.println();
            System.out.println();
        }
    }


    public static void main(String[] args) {


        String[] countryNames = {"USA", "India", "Brazil", "UK", "France"};

        int totalCases = 0;
        int totalRecoveredCases = 1;
        int activeCases = 2;

        int [][] covidCases = {
                        {66995533, 43090644, 23031325}, // 0 -> USA
                        {37380253, 35237461, 1656310},  // 1 -> India
                        {23006952, 621099, 675022},     // 2 -> Brazil
                        {15217280, 151987, 3676112},    // 3 -> UK
                        {14172384, 9019484, 5025933},   // 4 -> France
```

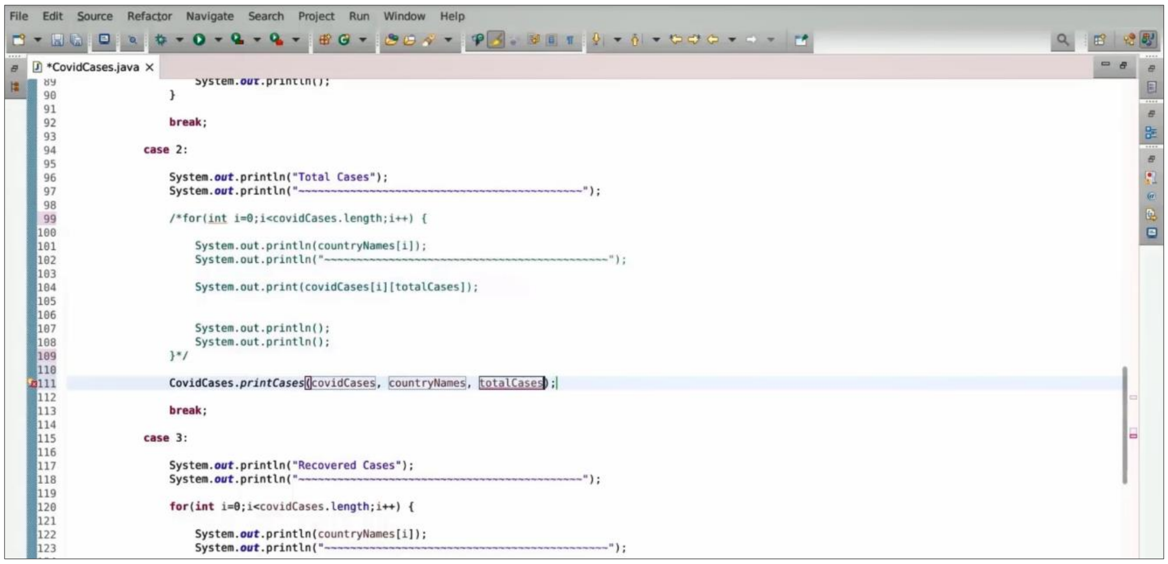
5.3 You can even take one more array, like the array of strings which is the country names, instead of these total cases, you can pass down this filter now. You have a method created as a static method, which will take the array of COVID cases, an array of country names, and a filter as input



```java
import java.util.Scanner;

public class CovidCases {


    static void printCases(int[][] covidCases, String[] countryNames, int filter) {
        for(int i=0;i<covidCases.length;i++) {

            System.out.println(countryNames[i]);
            System.out.println("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~");

            System.out.print(covidCases[i][totalCases]);


            System.out.println();
            System.out.println();
        }
    }


    public static void main(String[] args) {


        String[] countryNames = {"USA", "India", "Brazil", "UK", "France"};

        int totalCases = 0;
        int totalRecoveredCases = 1;
        int activeCases = 2;

        int [][] covidCases = {
                                {66995533, 43090644, 23031325}, // 0 -> USA
                                {37380253, 35237461, 1656310},  // 1 -> India
                                {23006952, 621099, 675022},     // 2 -> Brazil
                                {15217280, 151987, 3676112},    // 3 -> UK
```

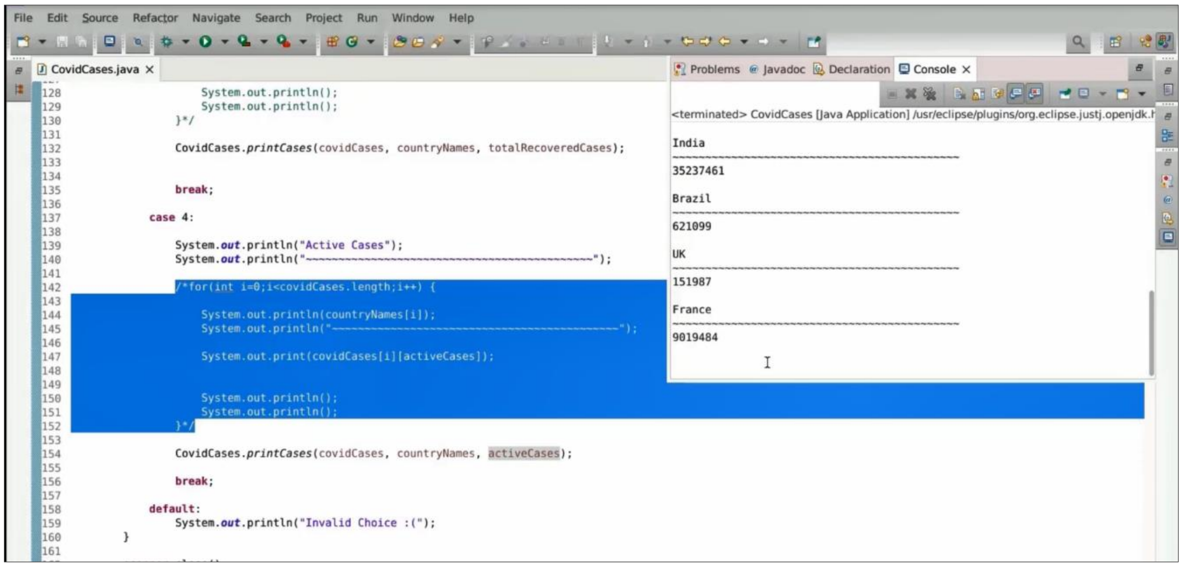
Step 6: Differentiate between running a static and a non-static method

6.1 Scroll back to your case number 2, where you want to print the total cases. Now, you can replace this logic with a single line of code: **CovidCases.printCases(covidCases, countryNames, totalCases);**. Do the same for cases 3 and 4



```java
                System.out.println();
            }

            break;

        case 2:

            System.out.println("Total Cases");
            System.out.println("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~");

            /*for(int i=0;i<covidCases.length;i++) {

                System.out.println(countryNames[i]);
                System.out.println("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~");

                System.out.print(covidCases[i][totalCases]);


                System.out.println();
                System.out.println();
            }*/

            CovidCases.printCases(covidCases, countryNames, totalCases);

            break;

        case 3:

            System.out.println("Recovered Cases");
            System.out.println("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~");

            for(int i=0;i<covidCases.length;i++) {

                System.out.println(countryNames[i]);
                System.out.println("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~");
```

6.2 Run the code. Now, if you try to filter based on the total cases, it will work the same way. If you try to filter based on recovered cases, it will give you the recovered cases. Similarly, if you filter based on active cases, it will give you active cases



```java
                System.out.println();
                System.out.println();
            }*/

            CovidCases.printCases(covidCases, countryNames, totalRecoveredCases);


            break;

        case 4:

            System.out.println("Active Cases");
            System.out.println("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~");

            /*for(int i=0;i<covidCases.length;i++) {

                System.out.println(countryNames[i]);
                System.out.println("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~");

                System.out.print(covidCases[i][activeCases]);


                System.out.println();
                System.out.println();
            }*/

            CovidCases.printCases(covidCases, countryNames, activeCases);

            break;

        default:
            System.out.println("Invalid Choice :(");
    }
```

```
<terminated> CovidCases [Java Application] /usr/eclipse/plugins/org.eclipse.justj.openjdk.h

India
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
35237461

Brazil
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
621099

UK
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
151987

France
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
9019484
```

By following the above steps, you have successfully depicted how methods are used and implemented in Java.