Lesson 06 Demo 03

Converting Collections to Array in Java

Objective: To convert collections to arrays in Java 11 using efficient methods and proper data structures
Tools required: Eclipse IDE

Prerequisites: None

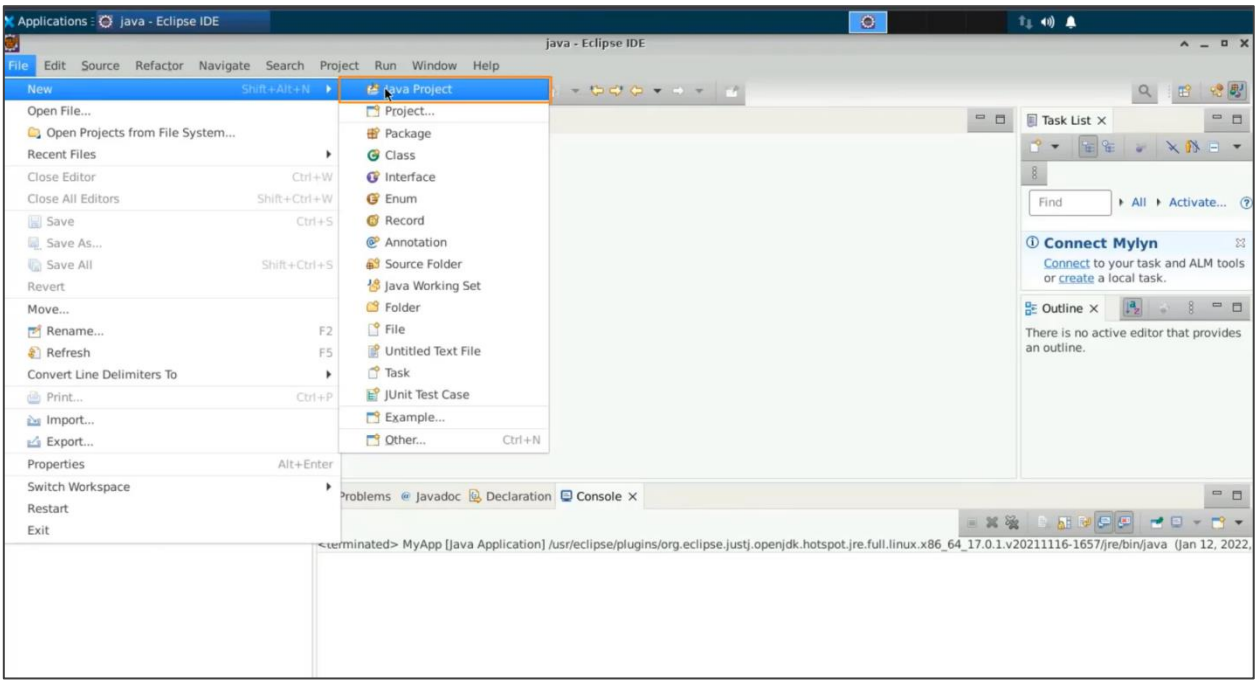
Steps to be followed:

1. Open the Eclipse IDE and create a new Java project
2. Write few emails as a new array list
3. Convert the array list back to an array
4. Print emails and the data coming in
5. Create an array of objects

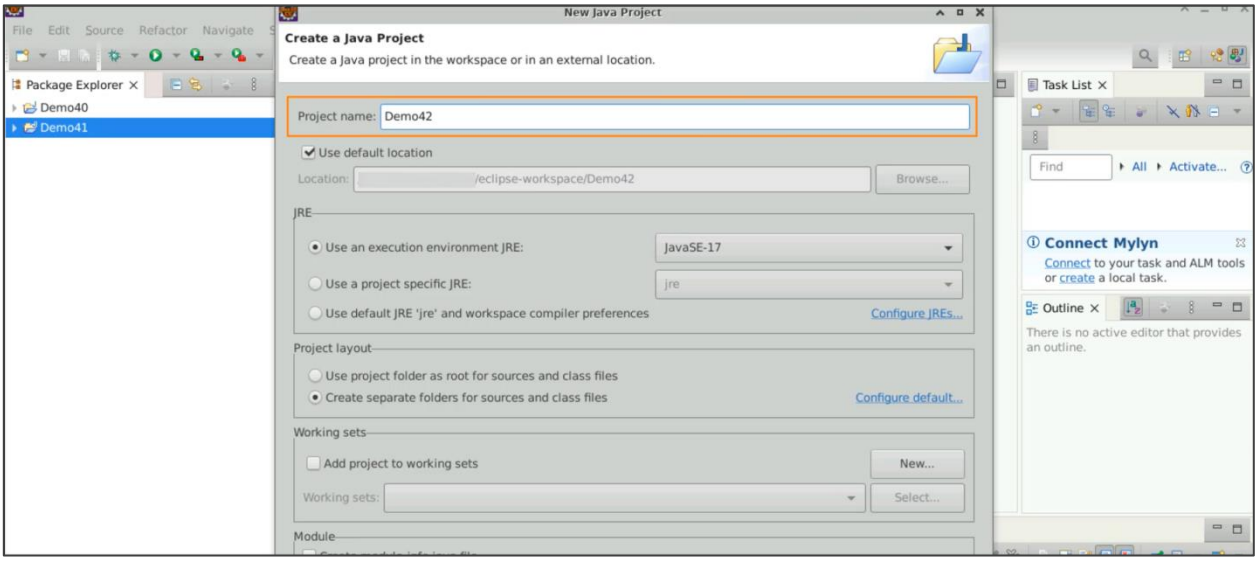
Step 1: Open the Eclipse IDE and create a new Java project

1.1 Open the **Eclipse IDE**

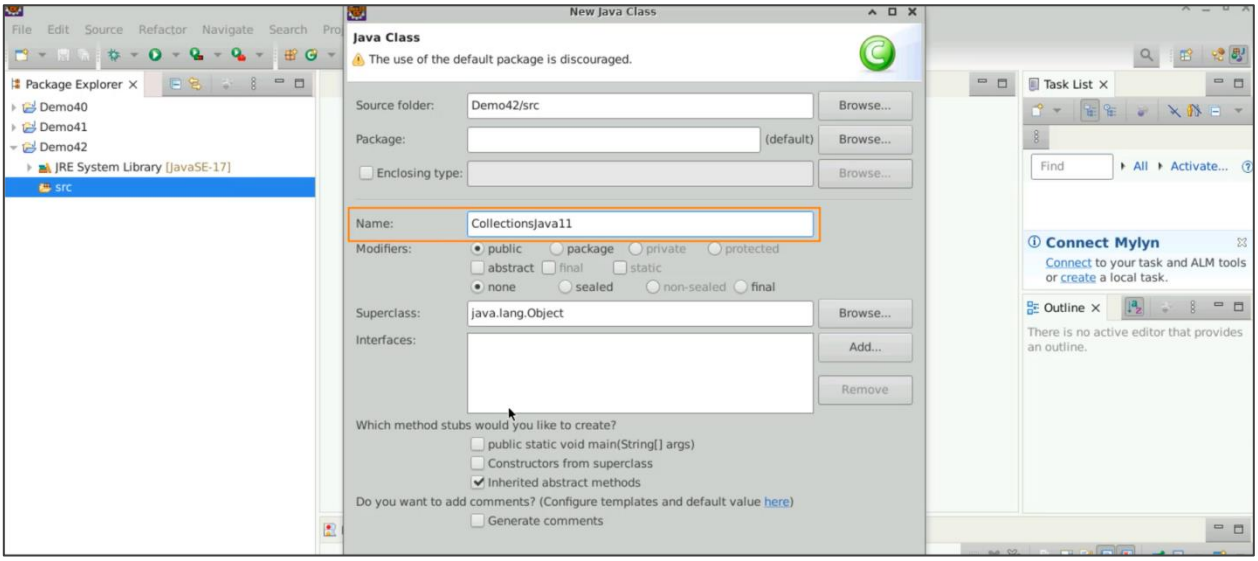
1.2 Select **File**, then **New,** and then **Java project**



1.3 Name the project **Demo42,** uncheck **Create a module-info.java file**, and press **Finish**

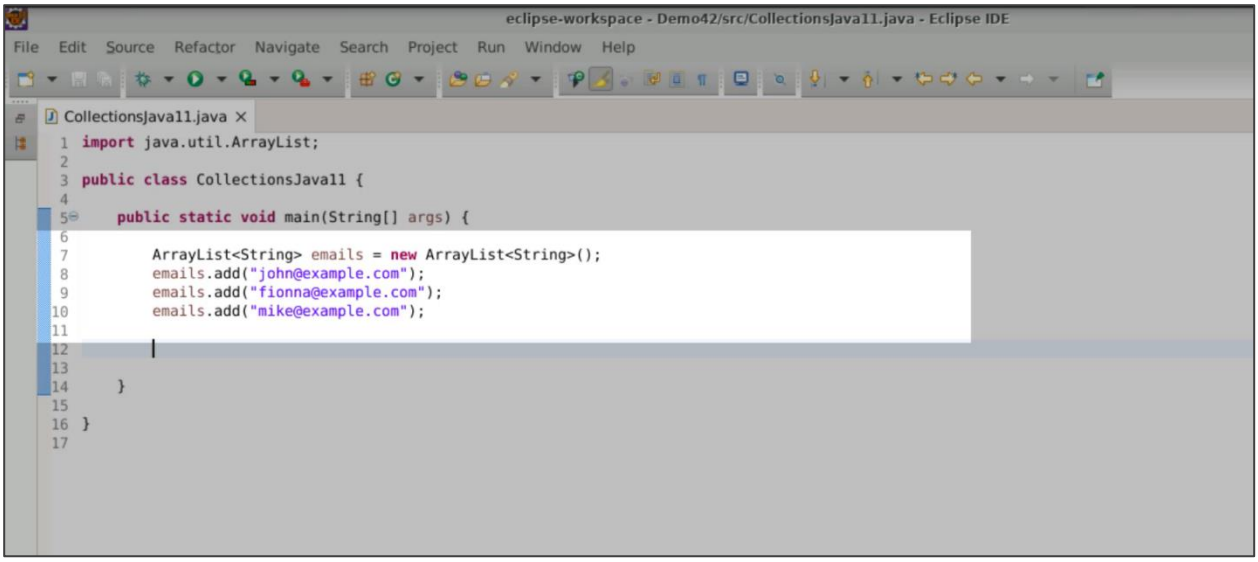


1.4 With **Demo42** selected in the **src** folder, right-click and create a new class. Name this class **CollectionsJava11**, then select the main method, and then select Finish

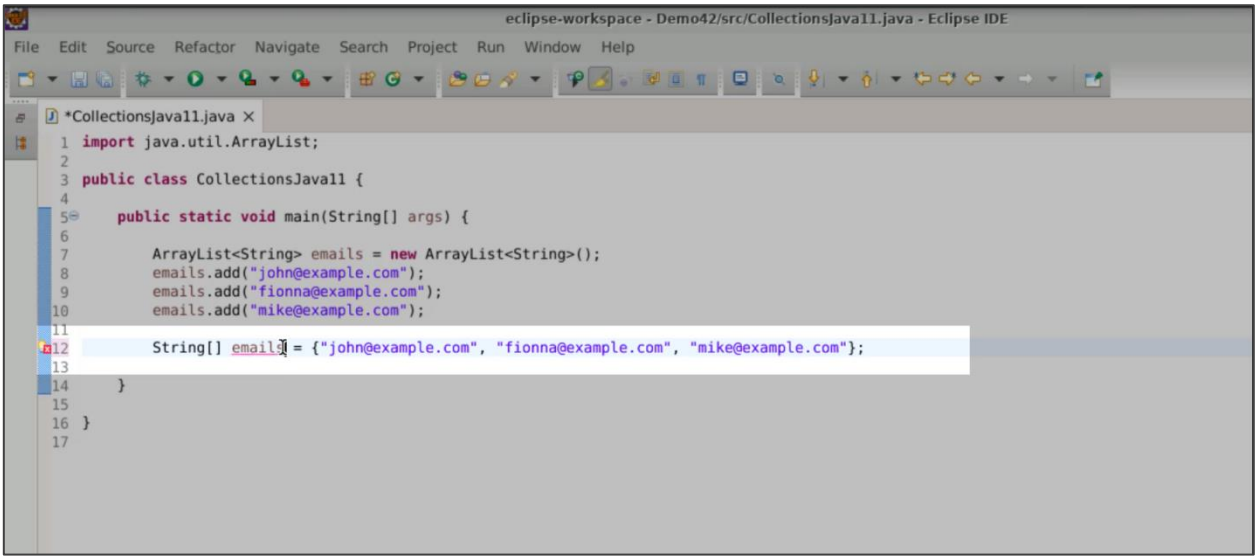


Step 2: Write few emails as a new array list

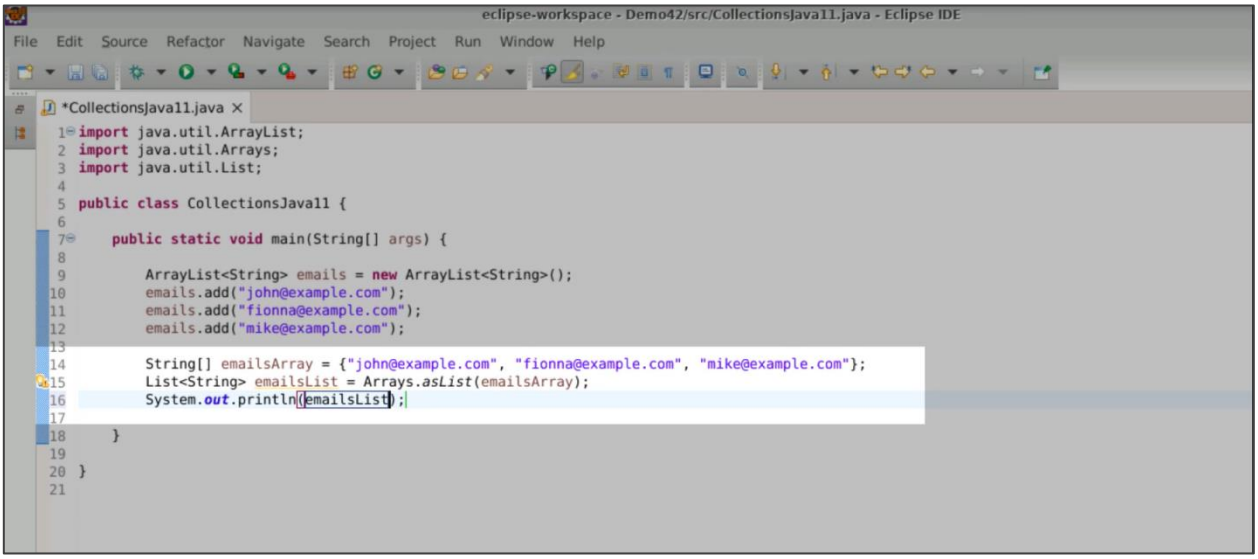
2.1 Let us write a type String and add a few emails as a new **ArrayList**



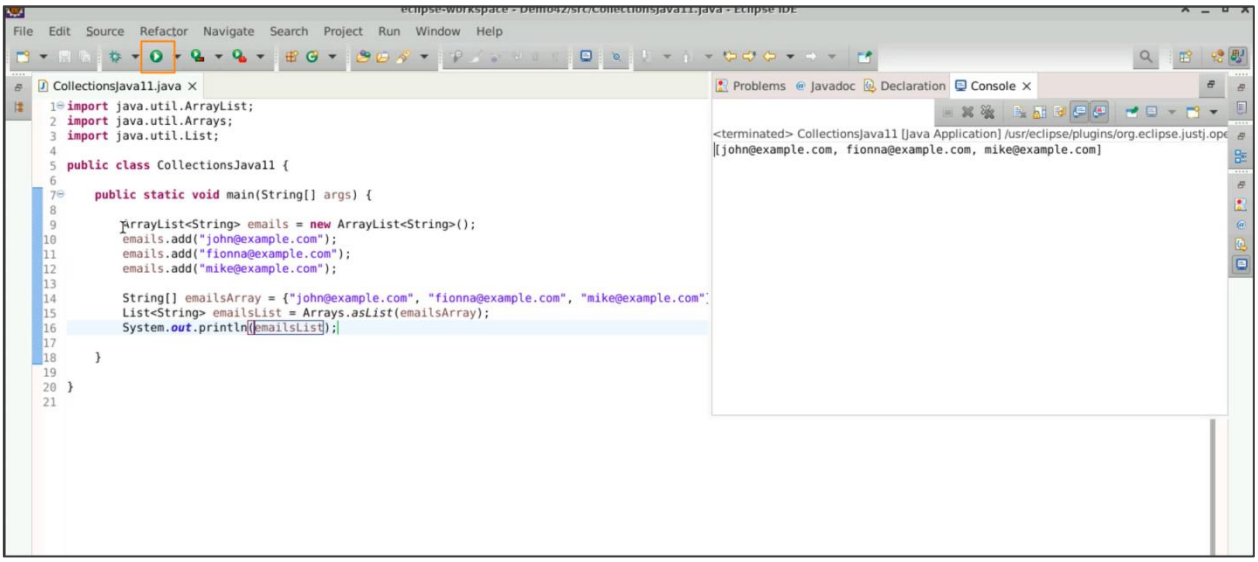
2.2 To simplify, you can create an array of emails and convert it into a list. This allows you to have a simple representation of the data using a homogeneous container



2.3 Next, you can use an email array and convert it to a list using **Arrays.asList()**. This will give you a list of emails which you can refer to as "**emails list**" from the **java.util** package

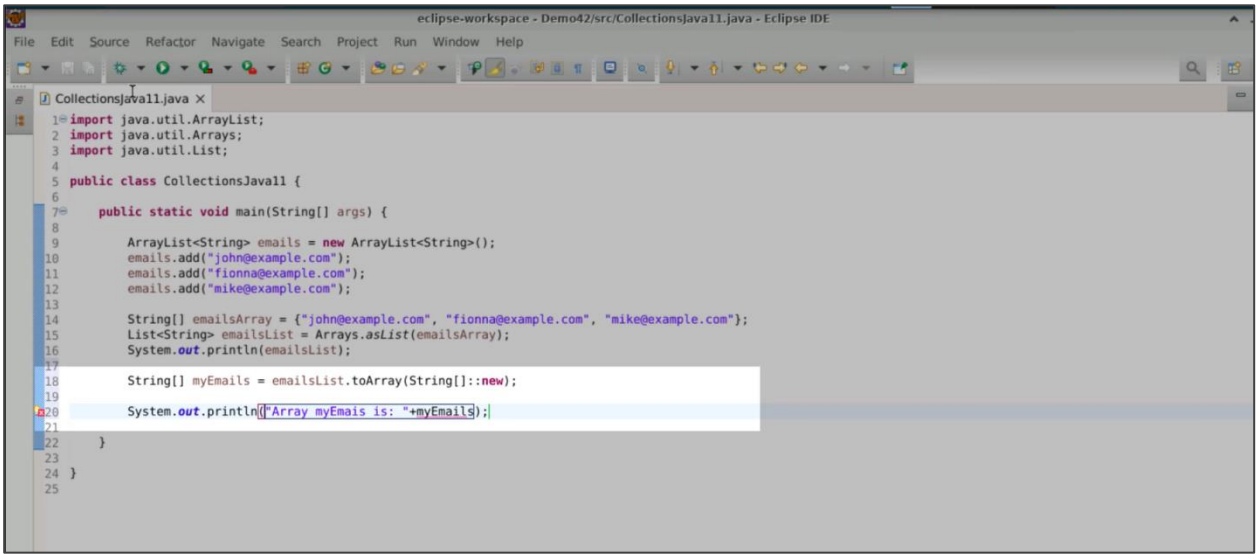


2.4 By converting the array to a list, you can process the emails list faster and more efficiently than adding emails one by one. It provides a quicker alternative to create and represent the list

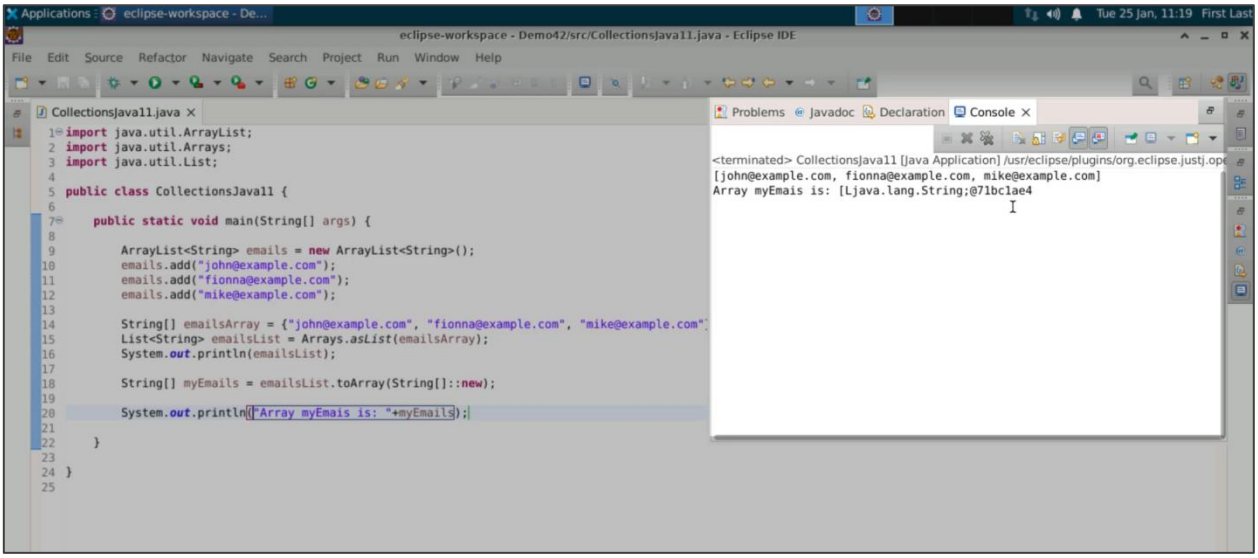


Step 3: Convert the array list back to an array

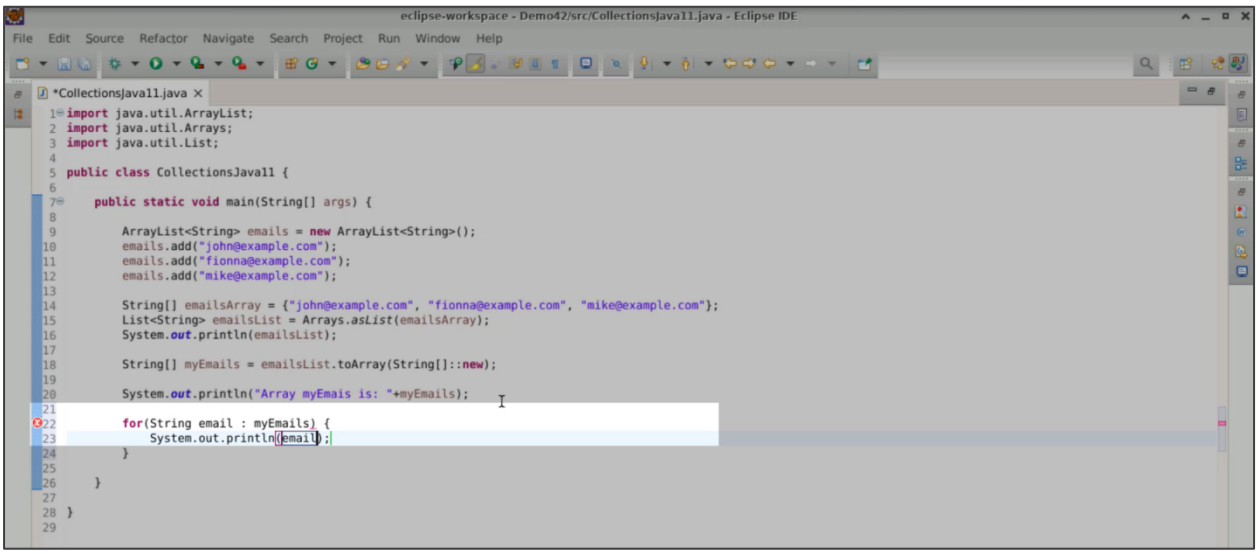
3.1 With the Java 11 feature, you can convert a list or ArrayList back to an array. For instance, use the **toArray** method on your **emails list** and specify the data type, such as creating a new string array using "**Array myEmails is : "+myEmails**



3.2 As you know, whenever you print the array, it will always give the hash code. This is how you can see the hash code coming in

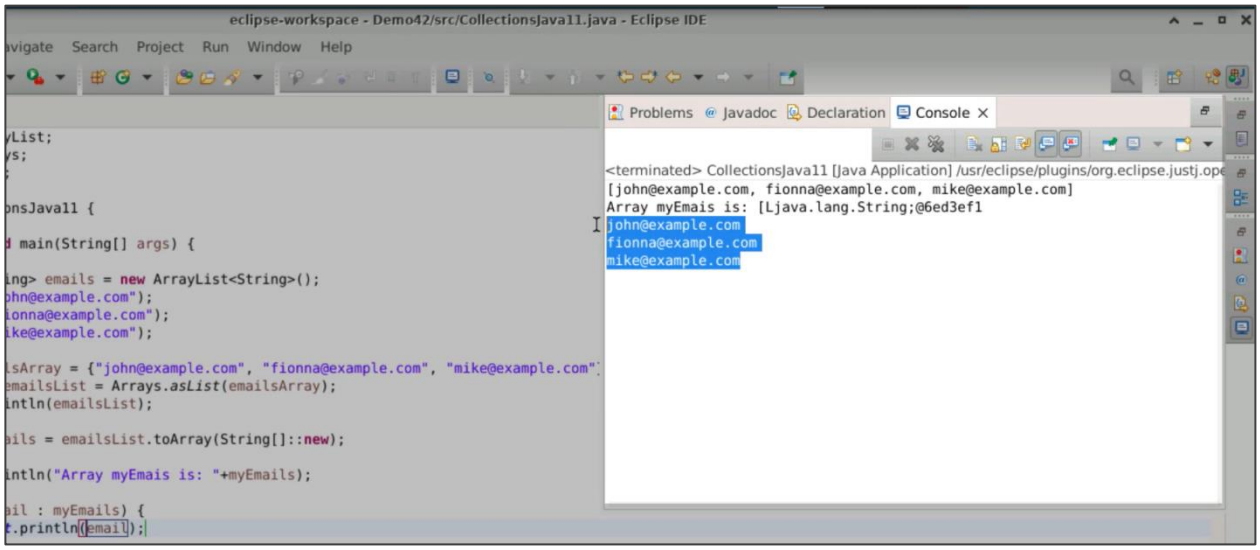


3.3 Now, let us write a loop to print each email in the **myEmails** array



Step 4: Print emails and the data coming in

4.1 Let us print email one by one and here is the data coming in



Step 5: Create an array of objects

5.1 If you have heterogeneous list, you can create an array of objects as per your the requirement. Currently, you have string array coming up

```
import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;

public class CollectionsJava11 {

    public static void main(String[] args) {

        ArrayList<String> emails = new ArrayList<String>();
        emails.add("john@example.com");
        emails.add("fionna@example.com");
        emails.add("mike@example.com");

        String[] emailsArray = {"john@example.com", "fionna@example.com", "mike@example.com"};
        List<String> emailsList = Arrays.asList(emailsArray);
        System.out.println(emailsList);

        String[] myEmails = emailsList.toArray(Object[]::new);

        System.out.println("Array myEmais is: "+myEmails);

        for(String email : myEmails) {
            System.out.println(email);
        }
```

By following these steps, you have successfully converted collections to arrays in Java 11 using efficient methods and proper data structures.