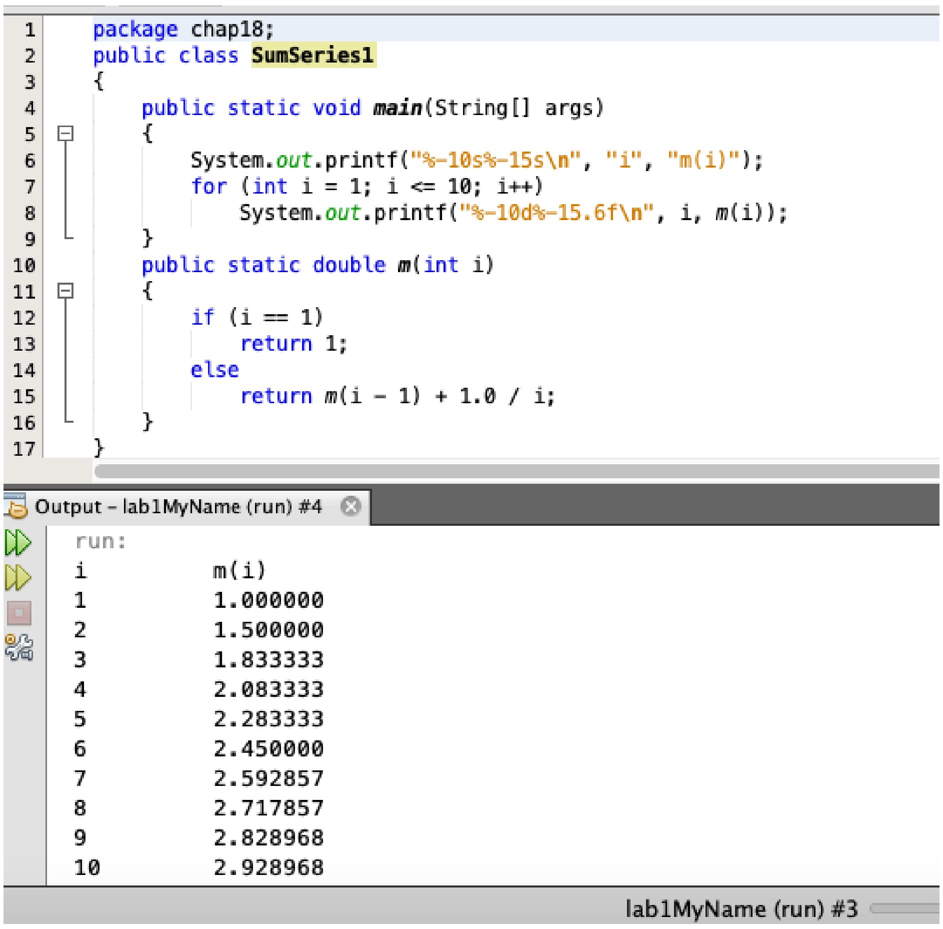
Recursion Lab

1. Implement class SumSeries1.

Write a recursive method to compute the following series:
m(i) = 1 + 1 / 2   + 1/3 +..... + 1 / i

Test for s m(i) for i = 1, 2, . . ., 10.



```java
package chap18;
public class SumSeries1
{
    public static void main(String[] args)
    {
        System.out.printf("%-10s%-15s\n", "i", "m(i)");
        for (int i = 1; i <= 10; i++)
            System.out.printf("%-10d%-15.6f\n", i, m(i));
    }
    public static double m(int i)
    {
        if (i == 1)
            return 1;
        else
            return m(i - 1) + 1.0 / i;
    }
}
```

Output - lab1MyName (run) #4

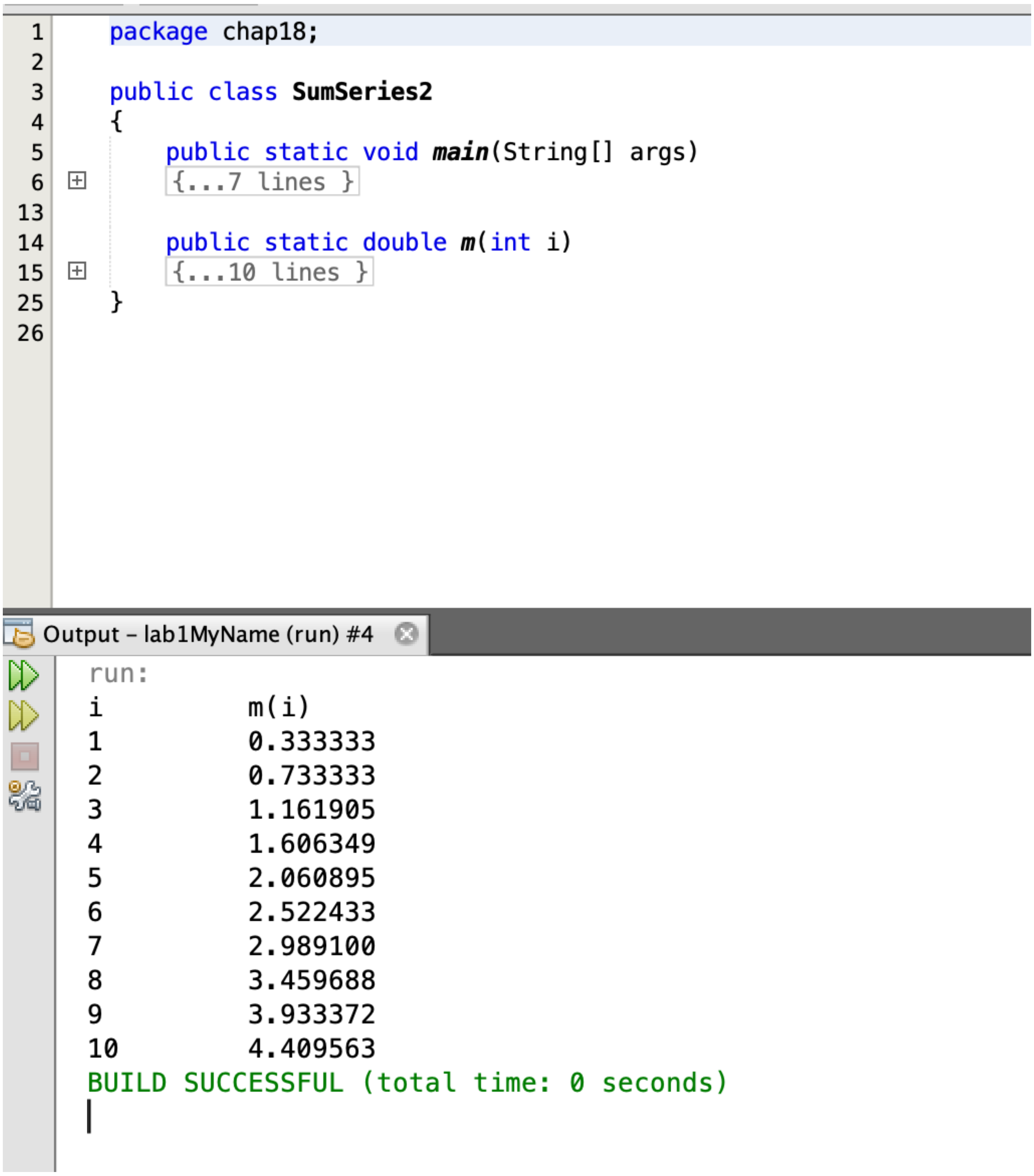
```
run:
i         m(i)
1         1.000000
2         1.500000
3         1.833333
4         2.083333
5         2.283333
6         2.450000
7         2.592857
8         2.717857
9         2.828968
10        2.928968
```

lab1MyName (run) #3

2. Implement class SumSeries2.

$$m(i) = \frac{1}{3} + \frac{2}{5} + \frac{3}{7} + \frac{4}{9} + \frac{5}{11} + \frac{6}{13} + \ldots + \frac{i}{2i+1}$$

Write a test program that displays `m(i)` for `i` = `1`, `2`, . . ., `10`.

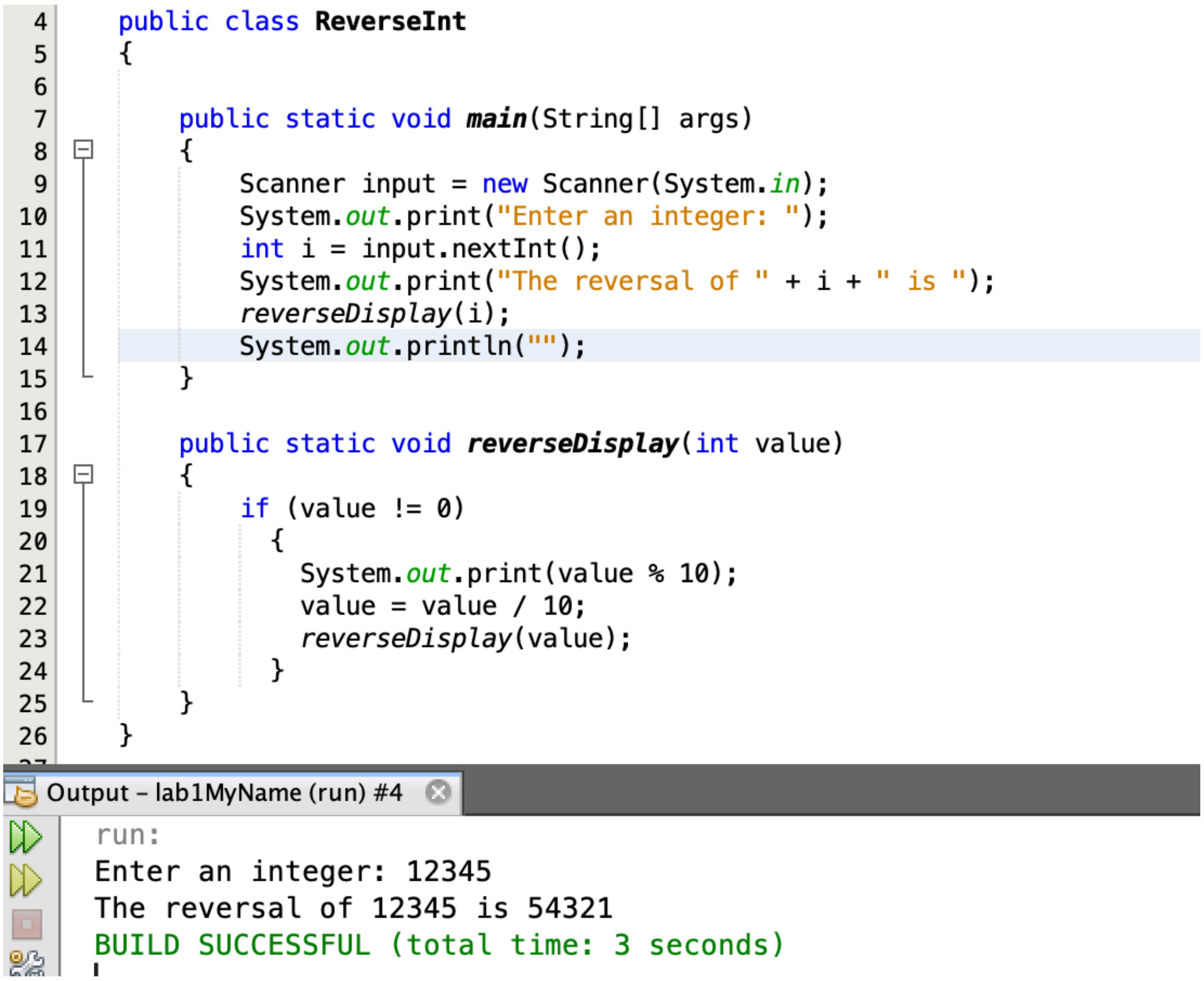


```
package chap18;

public class SumSeries2
{
    public static void main(String[] args)
    {...7 lines }

    public static double m(int i)
    {...10 lines }
}
```

Output – lab1MyName (run) #4

```
run:
i          m(i)
1          0.333333
2          0.733333
3          1.161905
4          1.606349
5          2.060895
6          2.522433
7          2.989100
8          3.459688
9          3.933372
10         4.409563
BUILD SUCCESSFUL (total time: 0 seconds)
```

3. Implement ReverseInt that displays an int value reversely.

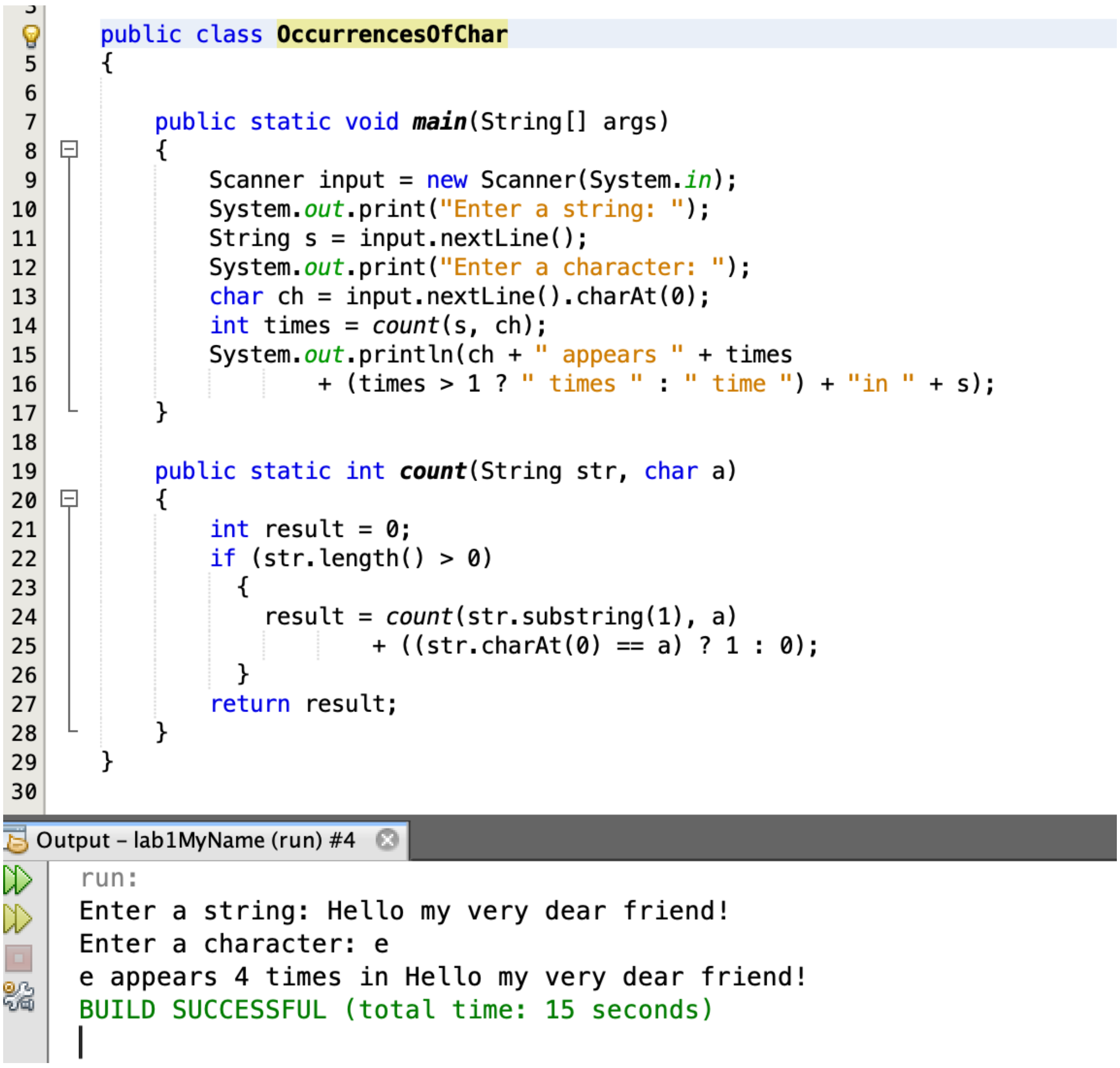


```java
public class ReverseInt
{

    public static void main(String[] args)
    {
        Scanner input = new Scanner(System.in);
        System.out.print("Enter an integer: ");
        int i = input.nextInt();
        System.out.print("The reversal of " + i + " is ");
        reverseDisplay(i);
        System.out.println("");
    }

    public static void reverseDisplay(int value)
    {
        if (value != 0)
          {
            System.out.print(value % 10);
            value = value / 10;
            reverseDisplay(value);
          }
    }
}
```

Output – lab1MyName (run) #4

```
run:
Enter an integer: 12345
The reversal of 12345 is 54321
BUILD SUCCESSFUL (total time: 3 seconds)
```

4. Implement OccurrencesOfChar that finds the number of occurrences of a specified letter in a string.

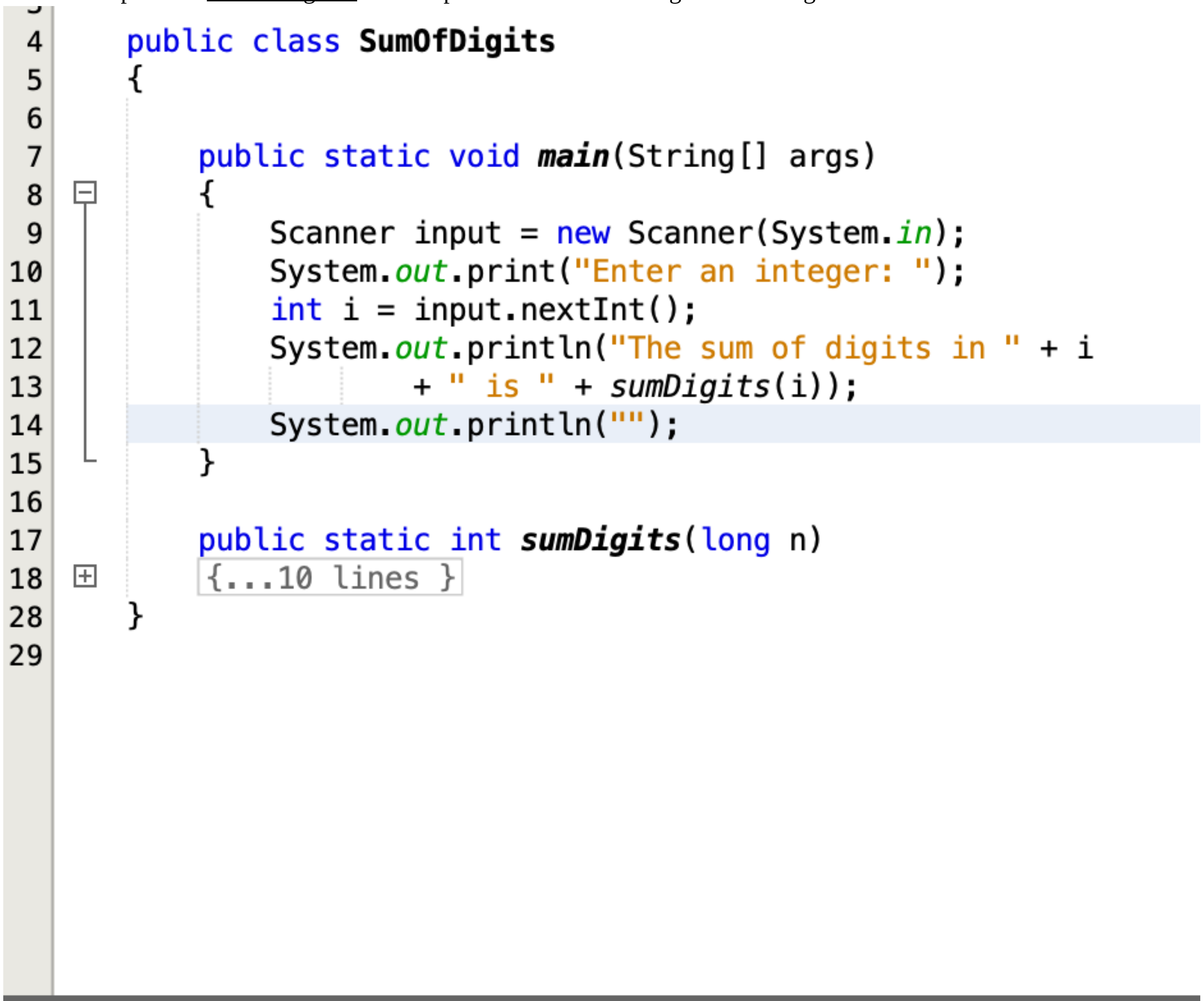


```java
public class OccurrencesOfChar
{

    public static void main(String[] args)
    {
        Scanner input = new Scanner(System.in);
        System.out.print("Enter a string: ");
        String s = input.nextLine();
        System.out.print("Enter a character: ");
        char ch = input.nextLine().charAt(0);
        int times = count(s, ch);
        System.out.println(ch + " appears " + times
                + (times > 1 ? " times " : " time ") + "in " + s);
    }

    public static int count(String str, char a)
    {
        int result = 0;
        if (str.length() > 0)
        {
            result = count(str.substring(1), a)
                    + ((str.charAt(0) == a) ? 1 : 0);
        }
        return result;
    }
}
```

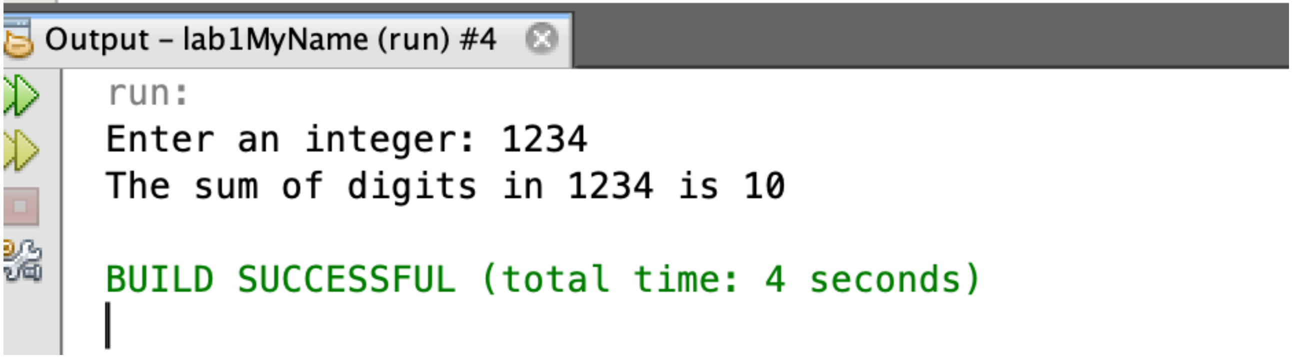
Output – lab1MyName (run) #4

```
run:
Enter a string: Hello my very dear friend!
Enter a character: e
e appears 4 times in Hello my very dear friend!
BUILD SUCCESSFUL (total time: 15 seconds)
```

5. Implement SumOfDigitsUthat computes the sum of the digits in an integer.



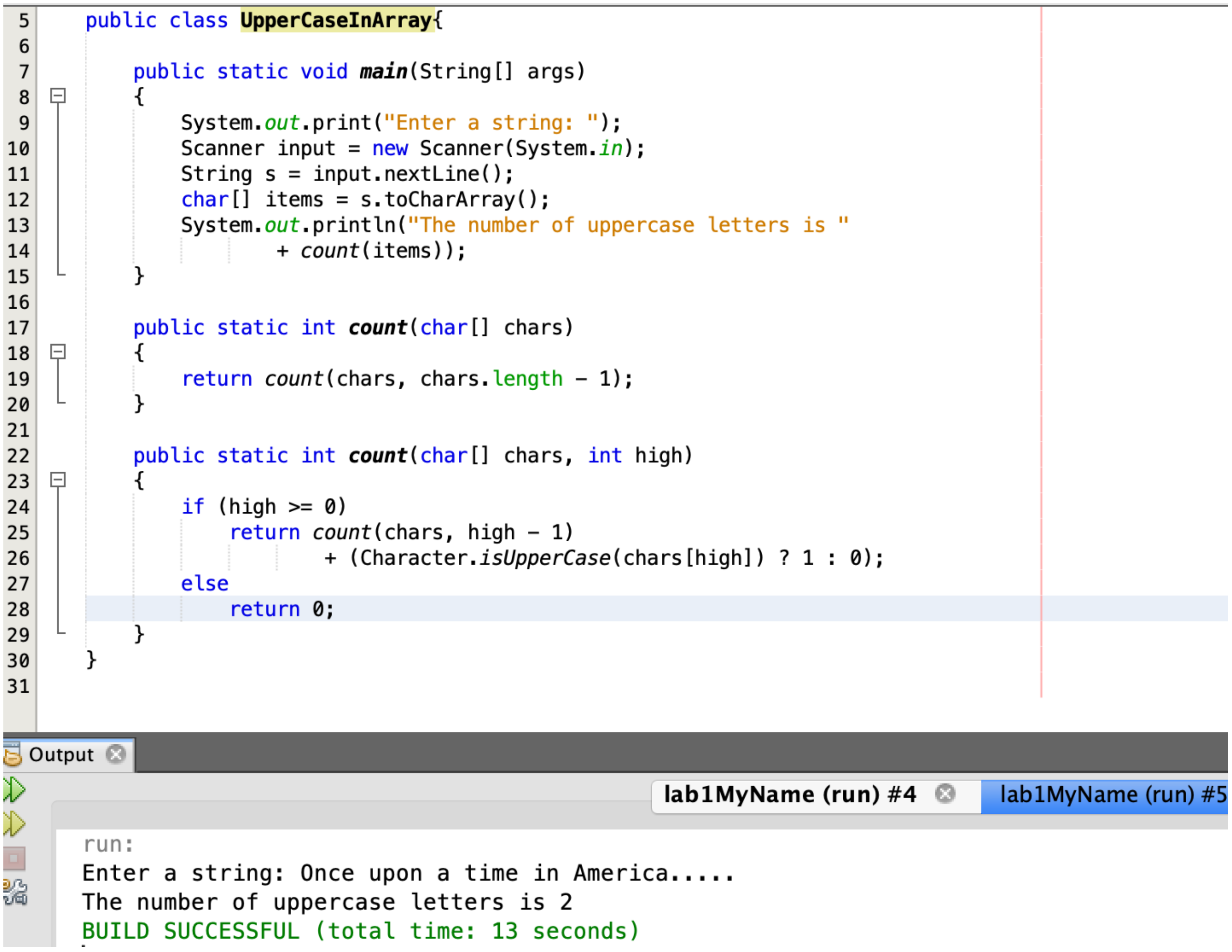
```
public class SumOfDigits
{

    public static void main(String[] args)
    {
        Scanner input = new Scanner(System.in);
        System.out.print("Enter an integer: ");
        int i = input.nextInt();
        System.out.println("The sum of digits in " + i
                + " is " + sumDigits(i));
        System.out.println("");
    }

    public static int sumDigits(long n)
    {...10 lines }
}
```



Output – lab1MyName (run) #4

```
run:
Enter an integer: 1234
The sum of digits in 1234 is 10

BUILD SUCCESSFUL (total time: 4 seconds)
```

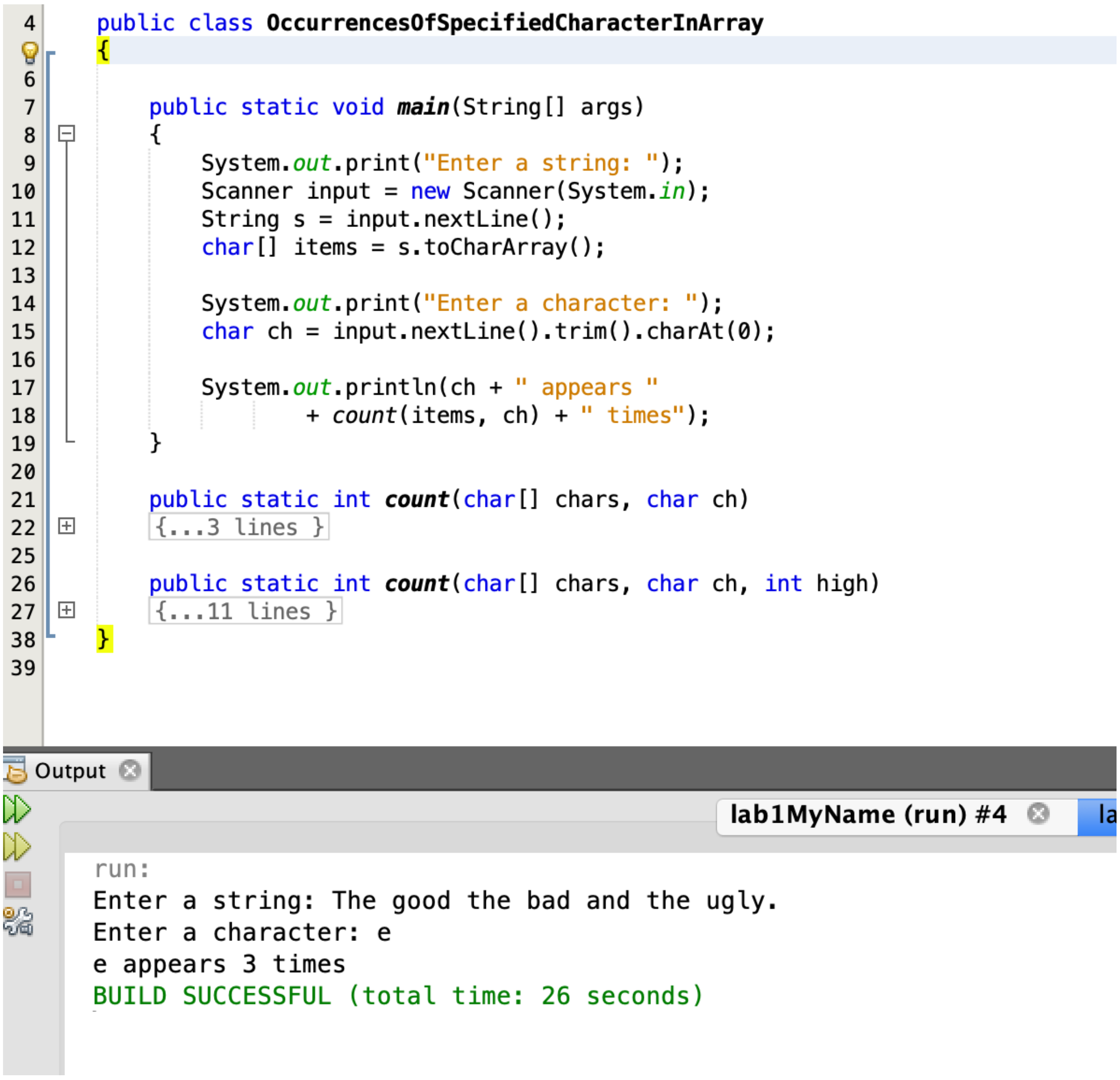
6. Implement UpperCaseInArray that returns the number of uppercase letters in an array of characters. You need to define two methods. The second one is a recursive helper method.



```java
public class UpperCaseInArray{

    public static void main(String[] args)
    {
        System.out.print("Enter a string: ");
        Scanner input = new Scanner(System.in);
        String s = input.nextLine();
        char[] items = s.toCharArray();
        System.out.println("The number of uppercase letters is "
                + count(items));
    }

    public static int count(char[] chars)
    {
        return count(chars, chars.length - 1);
    }

    public static int count(char[] chars, int high)
    {
        if (high >= 0)
            return count(chars, high - 1)
                    + (Character.isUpperCase(chars[high]) ? 1 : 0);
        else
            return 0;
    }
}
```

```
run:
Enter a string: Once upon a time in America.....
The number of uppercase letters is 2
BUILD SUCCESSFUL (total time: 13 seconds)
```

7. Implement OccurrencesOfSpecifiedCharacterInArray that finds the number of occurrences of a specified character in an array. You need to define two methods. The second one is a recursive helper method.



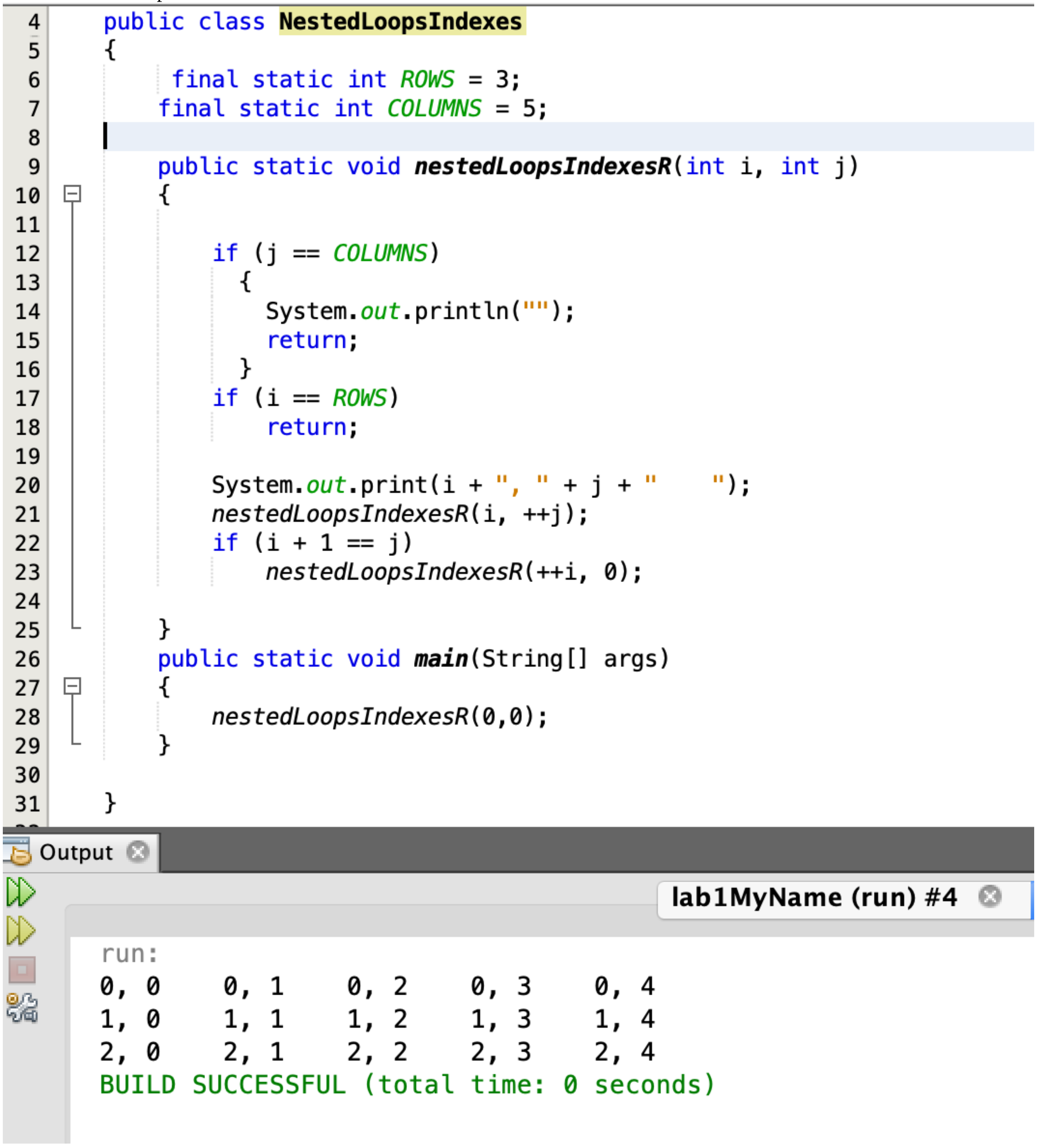
```
public class OccurrencesOfSpecifiedCharacterInArray
{

    public static void main(String[] args)
    {
        System.out.print("Enter a string: ");
        Scanner input = new Scanner(System.in);
        String s = input.nextLine();
        char[] items = s.toCharArray();

        System.out.print("Enter a character: ");
        char ch = input.nextLine().trim().charAt(0);

        System.out.println(ch + " appears "
                + count(items, ch) + " times");
    }

    public static int count(char[] chars, char ch)
    {...3 lines }

    public static int count(char[] chars, char ch, int high)
    {...11 lines }
}
```

Output

lab1MyName (run) #4

```
run:
Enter a string: The good the bad and the ugly.
Enter a character: e
e appears 3 times
BUILD SUCCESSFUL (total time: 26 seconds)
```

8. Implement NestedLoopsIndexes that prints the indexes of 2 for-nested loops:
outer loop 0 to ROWS
inner loop 0 to COLUMS



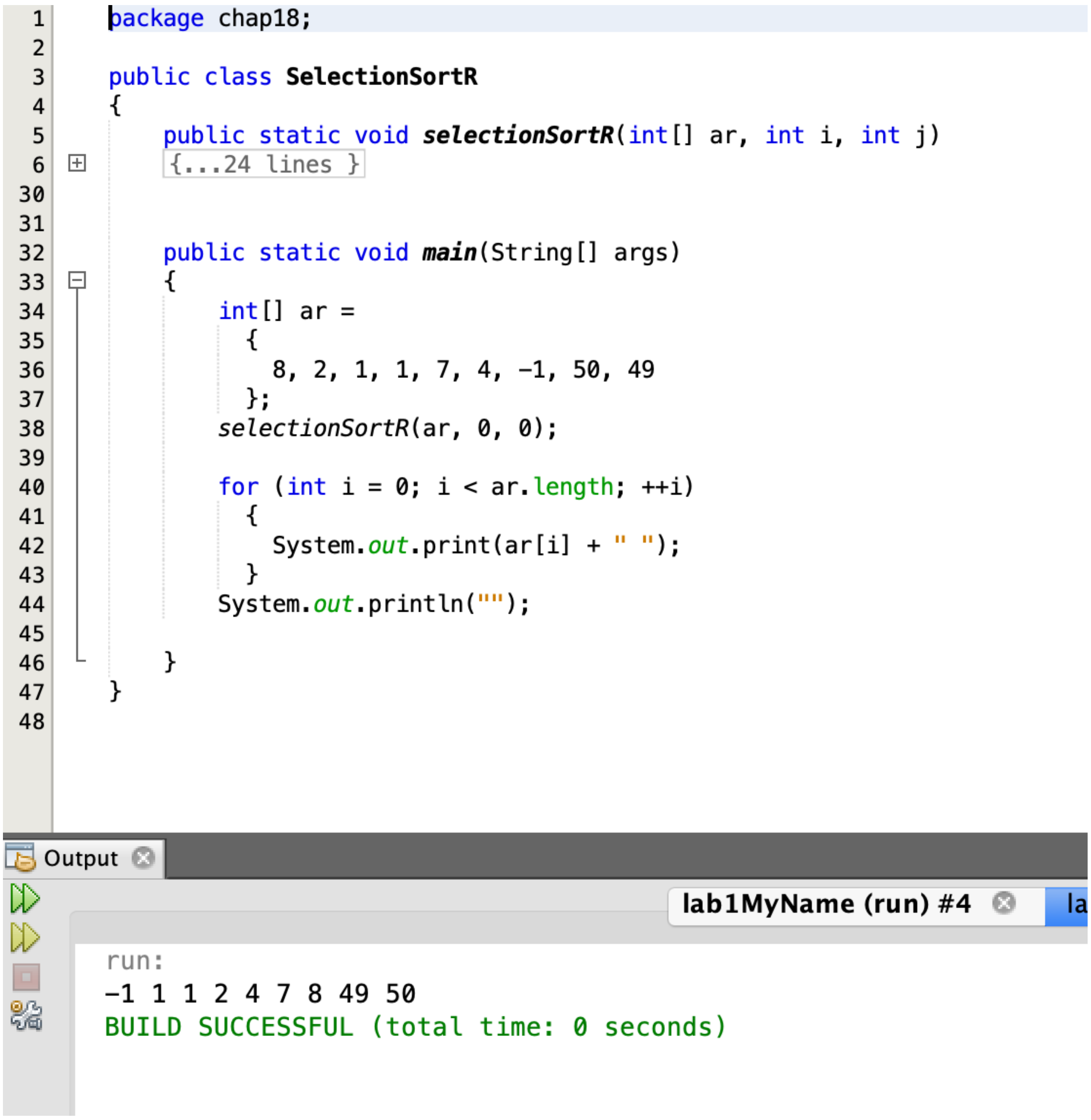
```
public class NestedLoopsIndexes
{
     final static int ROWS = 3;
    final static int COLUMNS = 5;

    public static void nestedLoopsIndexesR(int i, int j)
    {

        if (j == COLUMNS)
          {
            System.out.println("");
            return;
          }
        if (i == ROWS)
            return;

        System.out.print(i + ", " + j + "    ");
        nestedLoopsIndexesR(i, ++j);
        if (i + 1 == j)
            nestedLoopsIndexesR(++i, 0);

    }
    public static void main(String[] args)
    {
        nestedLoopsIndexesR(0,0);
    }

}
```

Output

lab1MyName (run) #4

```
run:
0, 0    0, 1    0, 2    0, 3    0, 4
1, 0    1, 1    1, 2    1, 3    1, 4
2, 0    2, 1    2, 2    2, 3    2, 4
BUILD SUCCESSFUL (total time: 0 seconds)
```

9. Implement SelectionSortR that sort recursively any array of integers. Test it for the array given in main



```java
package chap18;

public class SelectionSortR
{
    public static void selectionSortR(int[] ar, int i, int j)
    {...24 lines }


    public static void main(String[] args)
    {
        int[] ar =
          {
            8, 2, 1, 1, 7, 4, -1, 50, 49
          };
        selectionSortR(ar, 0, 0);

        for (int i = 0; i < ar.length; ++i)
          {
            System.out.print(ar[i] + " ");
          }
        System.out.println("");

    }
}
```

```
run:
-1 1 1 2 4 7 8 49 50
BUILD SUCCESSFUL (total time: 0 seconds)
```