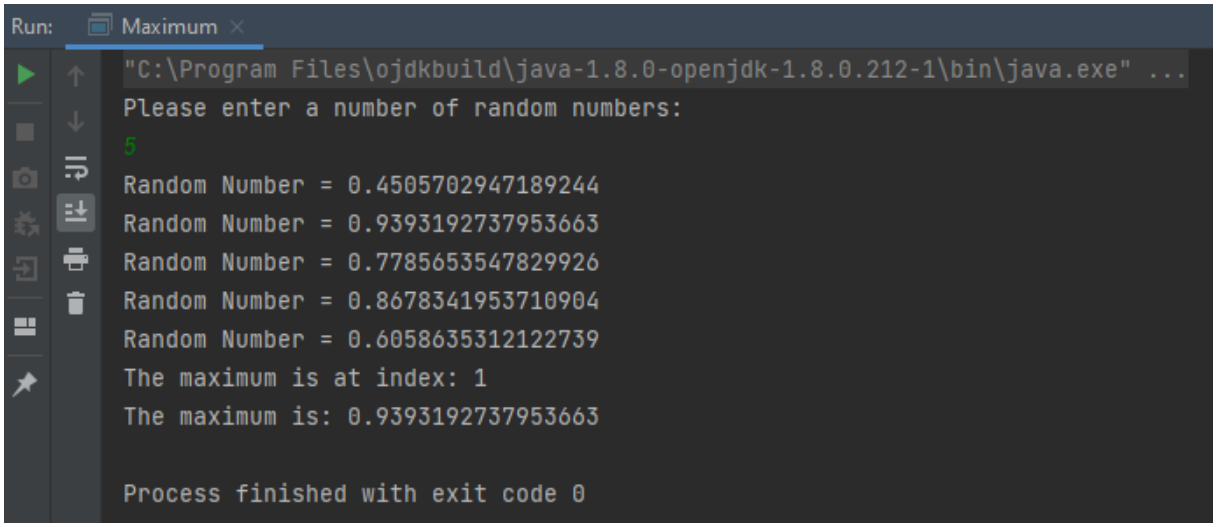
Exercise sheet 4 – Methods, fields and control structures

Task 1 – Find the maximum

Create a class `Maximum` and add a method `max` to the class, which takes a `double` array as an argument and returns the array index at whose position the largest number is located.
Also write a `main` method to test your `max` method. To do so, read from the console how many random numbers should be generated, and then create a corresponding array of random numbers. Then use the `max` method to find both the index and the value of the maximum.



Note:

- You need a main() and a max() method
- main():
  - Scanner for reading the array size
  - Create an array of the size entered by the user
  - Populate the array with random values using Math.random()
  - Call max() whereby the array is passed
  - Output the array and the index where the maximum value is located.
- max():
  - Assume that the first array value at index 0 is the maximum and temporarily store the values
  - Use a for loop, iterate through the array and compare each array position with the temporarily stored value.
  - If you find a higher value, temporarily store this and the index in the variables you have declared, and use the new maximum value in the next loop iteration.
  - max() returns the index as return parameter
  - Use the method `Math.random()` to generate a random number.

Task 2 - Intersection

Create a class `Intersection` and add a method `intersection` to the class, which takes two `int` arrays as arguments, and returns an `int` array which contains the numbers that occur in both arrays (from Mathematics - set theory: intersection).
Also write a `main` method in which you test the `intersection` method using some (specified) examples:

- {0, 1, 2, 3, 4, 5} and {3, 4, 5, 6, 7, 8} → {3, 4, 5} (set1a, set1b)
- {0, 1, 2, 3} and {4, 5, 6} → {} (set2a, set2b)
- {0, 1, 2}, {0, 1, 2} → {0, 1, 2} (set3a, set3b)

Note:

- You need a main() and a max() method
- main():
  - In this task, you don't have to read values from the keyboard, a "static" test is sufficient.
  - You should generate 6 arrays as specified above.
  - Here you call the method intersection() and pass the arrays you want to check for an intersection.
  - You can use the following printouts for output:

```
System.out.println(Arrays.toString(intersection(set1a, set1b)));
System.out.println(Arrays.toString(intersection(set2a, set2b)));
System.out.println(Arrays.toString(intersection(set3a, set3b)));
```

- intersection():
  - Because the size of arrays is fixed after creation, you must determine the intersection twice: once to find the size (and create the return array), and once to insert the values.
  - 1st for-loop: find the number of intersection elements
  - Create the array of the size determined in the 1st loop.
  - 2nd for-loop: populate the array created with the intersection elements.

Task 3 - Change

Create a new class `Change` and add a method `change` to the class, which takes as arguments a price (or set value) and the amount paid by the customer, and outputs the correct change to the console (in 100, 50, 20, 10, 5 Euro notes, and 2, 1, 0.50, 0.20, 0.10, 0.05, 0.02, 0.01 Euro coins) that the customer receives. The return value of the method should be `false` if the customer paid too little, otherwise `true`.
Also write a `main` method to test your `change` method. To do so, read both the price and the amount paid from the keyboard, and output to the console whether enough has been paid ("Thank you very much") or not ("Unfortunately that's not enough").

```
Run:  Change
"C:\Program Files\ojdkbuild\java-1.8.0-openjdk-1.8.0.212-1\bin\java.exe" ...
Please enter the price 2.60
Please enter the amount: 3.00
Change:
2x 20ct
Thank you very much!

Process finished with exit code 0
```

Note:

- You need a `main()` and a `change()` method
- `main()`:
  - Scanner to read the price and amount input by the user; Attention: the user inputs the data type double, and it is helpful to store the values as CENT amounts and pass them as `int` to the method `change()`.
  - Call the function `change(price, amount)`
  - `Printout` as specified above
    - Use the return value of the `change` method.
- `change()`:
  - Since the smallest denomination required is in cent amounts, it is advisable to calculate the change denomination in whole cents (`int`).
  - Calculation of the return value
  - for loop through the centBreakdown array to determine the amount of change.