Tutorial of Array

Experimental Objective

- Learn array initializer (Declare, create, and initialize).
- Learn how to copy and print array by for loop
- Learn how to using array to realize some simple algorithms

Before Exercise

(1) Type following code, try to create two arrays and use two different ways to print them.

```java
int[] array1 = {1,2,3,4,5};
int[] array2 = new int[5];
array2[0] = 6;
array2[1] = 7;
array2[2] = 8;
array2[3] = 9;
array2[4] = 10;

for(int i = 0; i<array1.length; i++){
  System.out.print(array1[i] + "\t");
}
System.out.println();
for(int e:array2){
    System.out.print(e + "\t");
}
System.out.println();
```

(2) Continue typing, create another array without giving it an address, and then finding what is the difference between two arrays.

```java
int[] array3 = null;
System.out.println(array3);

array3=array2;
System.out.println(array3);
```

(3) Why the first loop cannot change the value of array3? The second loop can change the value.

```java
for(int e:array3){
    e=1;
}
System.out.println("array3: " + Arrays.toString(array3));
for(int i = 0; i<array3.length; i++){
  array3[i] = 1;
}
System.out.println("array3: "+Arrays.toString(array3));
```

(4) We change the value of elements in array3, why are the elements in array2 changed accordingly?

```java
System.out.println("array2: " + Arrays.toString(array2));
```

[Advanced]Try following code:

```java
char[] Array4 = {'a', 'b', 'c'};
System.out.println(Array4);
```

Exercise

Exercise 1:

Practice basic operations of an array:

- Declare and create an array named `myList1` with n (0<n<20) elements of double type.
- Initialize `myList1` with input values and make a copy of `myList1` named `myList2`.
- Shift the elements in `myList1` to the left by one position.
- Print the elements in `myList1` and `myList2`.

```
Enter the length of myList1:8
Enter 8 values: 2.5 5.5 3.4 6.4 7.7 2.2 8.9 0.2
myList1:5.5 3.4 6.4 7.7 2.2 8.9 0.2 2.5
myList2:2.5 5.5 3.4 6.4 7.7 2.2 8.9 0.2
```

Exercise 2：

Suppose there are 10 students in a class, and we want the average score of these 10 students. Input 10 scores ([0, 100]) from the keyboard. Then after removing the highest score and the smallest score, please find the average score of the other 8 scores.

```
Please input 10 scores of these students: 88.3 99 45 78 67.5 98.4 23.5 65.5 82
85.4 Average score is 76.26
```

Exercise 3:

Write a program to compare two arrays with same size. Let the user inputs the array size and every elements of the two arrays. Two arrays are considered equal if both arrays contain the same number of elements, and all corresponding pairs of elements in the two arrays are equal.

Sample input and output:

```
Enter the length of array:4
Enter the 1st integer array of size 4:1 2 3 4
Enter the 2nd integer array of size 4:1 2 3 4
The two arrays are equal.
```

Sample input and output:

```
Enter the length of array:3
Enter the 1st integer array of size 4:1 2 3
Enter the 2nd integer array of size 4:3 2 1
The two arrays are not equal.
```

Exercise 4:

Write a program that reads the integers between 1 and 100 and counts the occurrences of each. Assume the input ends with 0.

Here is a sample run of the program

```
Enter the integers between 1 and 100: 22 33 35 34 99 87 45 34 23 78 45 33 11 23
87 34 76 0
11 occurs 1 time
22 occurs 1 time
23 occurs 2 times
33 occurs 2 times
34 occurs 3 times
35 occurs 1 time
45 occurs 2 times
76 occurs 1 time
78 occurs 1 time
87 occurs 2 times
99 occurs 1 time
```

Exercise 5:

Write a program to sort an integer sequence in ascending order. The user will first input the size of the sequence, then the numbers. The program will print the sorted sequence.

Sample input and output:

```
How many numbers you will input: 10
3 5 2 99 44 54 23 46 87 56
2 3 5 23 44 46 54 56 87 99
```

Further Reading - Bubble sort

Original sequence: 35,64,18,22,51,7 For first outer loop: make sure to move the largest number to the end of array Compare:

```
if( array[i] >array[i+1]){
   int temp = array[i];
   array[i] = array[i+1];
   array[i+1] = temp;
  }
```

First Loop:

loop1: 35,**64**,18,22,51,7 (no change)
loop2: 35,18,**64**,22,51,7 (change)
loop3: 35,18,22,**64**,51,7 (change)
loop4: 35,18,22,51,**64**,7 (change)
loop5: 35,18,22,51,7,**64** (change)

For second outer loop: make sure to move the second largest number to the second end position of array

loop1: 18,**35**,22,51,7,64 (change)
loop2: 18,22,**35**,51,7,64 (change)
loop3: 18,22,35,**51**,7,64 (no change)
loop4: 18,22,35,7,**51**,64 (change)

Third outer loop:

loop1: 18,**22**,35,7,51,64 (no change)
loop2: 18,22,**35**,7,51,64 (no change)
loop3: 18,22,7,**35**,51,64 (change)

Exercise 6:

Write a program that prompts the user to input n integers from 1 to 1000 in ascending order.

Let μ be the average value of all the integers.

Count how many pairs of integers whose average value is greater than μ. (Please try to design your program to accomplish the task as fast as possible)

You can use the following code `(current2-current1)` to estimate the running time of your algorithm.

```
long current1=System.currentTimeMillis();
/* your algorithm */
long current2=System.currentTimeMillis();
System.out.printf("your program using %.3f
second",(current2-current1)/1000.0d);
```

Sample input and output:

```
Enter how many numbers: 5
Enter 5 numbers: 1 2 3 4 5
average=3.0
The number of pairs of integer is 4
The running time is 0.004 second
```

Sample input and output:

```
Enter how many numbers: 30
Enter 30 numbers: 2 3 5 6 9 10 12 13 15 16 23 55 66 77 89 101 220 221 222 255
277 280 290 300 303 400 420 455 500 520
average=172.16666666666
The number of pairs of integer is 194
The running time is 0.004 second
```