JOBSHEET 8
ARRAY 1

1. Objective

- Students are able to understand one-dimensional Array creation and accessing its elements in Java
- Students are able to make programs using the concept of one-dimensional arrays

2. Laboratory

2.1 Experiment 1: Fill in Array Element

1. Create a new project
2. Create a new class, name it **myArray**
3. Write the basic structure of the Java programming language which contains the **main()** function
4. Create an array of integer type named **num** with a capacity of 4 elements

```
int[] num = new int[4];
```

5. Fill each element of the array with numbers 5, 12, 7, 20

```
num[0] = 5;
num[1] = 12;
num[2] = 7;
num[3] = 20;
```

6. Display all contents of the elements to the screen

```
System.out.println(num[0]);
System.out.println(num[1]);
System.out.println(num[2]);
System.out.println(num[3]);
```

7. Compile and run the program. Match the results of the running programs that you have created according to the following display

Edit with WPS Office

```
5
12
7
20
```

Questions!

1. In Experiment 1, what are the largest and smallest array indexes?
2. If the contents of each element of the array **num** are changed with numbers 5.0, 12867, 7.5, 2000000. What happens? How can it be like that?
3. Change the statement in step 6 to be like this

```
for (int i = 0; i < 4; i++) {
    System.out.println(num[i]);
}
```

What is the result? How can it be like that?

2.2Experiment 2: Requesting User Input to Fill in an Array Element

1. Create a new class, name it **arrayInputLoop**
2. Write the basic structure of the Java programming language which contains the **main()** function
3. Add the Scanner library
4. Make a **Scanner** declaration with the name **sc**
5. Create an array of integer type with the name **finalScore**, with a capacity of 6 elements

```
int[] finalScore = new int[6];
```

6. Using a loop, create an input to fill in the **finalScore** array element

```
for (int i = 0; i < 6; i++) {
    System.out.print("Enter the final score " + i + ": ");
    finalScore[i] = sc.nextInt();
}
```


Edit with WPS Office

7. Using a loop, display all the contents of the elements from the **finalScore** array

```
for (int i = 0; i < 6; i++) {
    System.out.println("Final score " + i + " is " + finalScore[i]);
}
```

8. Compile and run the program. Match the results of the running programs that you have created according to the following display

```
Enter the final score 0: 88
Enter the final score 1: 90
Enter the final score 2: 74
Enter the final score 3: 83
Enter the final score 4: 92
Enter the final score 5: 77
Final score 0 is 88
Final score 1 is 90
Final score 2 is 74
Final score 3 is 83
Final score 4 is 92
Final score 5 is 77
```

Questions!

1. Change the statement in step 6 to be like this

```
for (int i = 0; i < finalScore.length; i++) {
    System.out.print("Enter the final score " + i + ": ");
    finalScore[i] = sc.nextInt();
}
```

Run the program. Have there been any changes? How can it be like that?

Edit with WPS Office

2. What is the use of **finalScore.length**?
3. Change the statement in step 7 to be like this, so that the program only displays the grades of students who passed

```
for (int i = 0; i < finalScore.length; i++) {
    if (finalScore[i]>70) {
        System.out.println("Final score " + i + " is " + finalScore[i]);
    }
}
```

Run the program and describe the flow of the program!

4. Modify the program so that it displays all students, and marked which one passed and which did not!

```
Enter the final score 0: 82
Enter the final score 1: 78
Enter the final score 2: 65
Enter the final score 3: 88
Enter the final score 4: 70
Enter the final score 5: 90
Student 0 Passed
Student 1 Passed
Student 2 Failed
Student 3 Passed
Student 4 Failed
Student 5 Passed
```

2.3 Experiment 3: Perform Arithmetic Operations on Array Elements

1. This experiment is done to add array elements. The program will accept input of 10 student scores. Then the program will display the average score of 10 students.

Edit with WPS Office

2. Create a new class, name it **averageScore**
3. Write the basic structure of the Java programming language which contains the **main()** function
4. Add the Scanner library
5. Make a **Scanner** declaration with the name **sc**
6. Create an array of integer type with the name **score** with a capacity of 10. Then declare the variables total and average

```
int[] score = new int[10];
double total = 0;
double average;
```

7. Using a loop, create an input to fill in the **score** array element

```
for (int i = 0; i < score.length; i++) {
    System.out.print("Enter student score " + (i + 1) + ": ");
    score[i] = sc.nextInt();
}
```

8. Using a loop, calculate the total number of scores.

```
for (int i = 0; i < score.length; i++) {
    total += score[i];
}
```

9. Calculate the average value by dividing **total** by the number of elements of **score**

```
average = total / score.length;
System.out.println("The class average score is " + average);
```

10. Compile and run the program. Match the results of the running programs that you have created according to the following display

Edit with WPS Office

```
Enter student score 1: 98
Enter student score 2: 73
Enter student score 3: 86
Enter student score 4: 82
Enter student score 5: 95
Enter student score 6: 68
Enter student score 7: 90
Enter student score 8: 71
Enter student score 9: 78
Enter student score 10: 84
The class average score is 82.5
```

Questions!

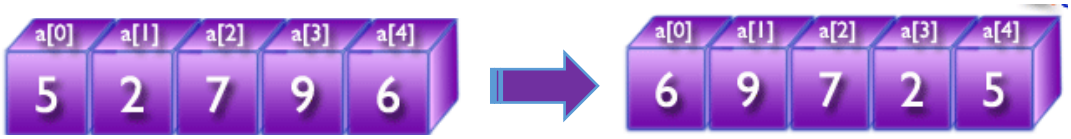
1. In step 9, why is the average calculation written outside the loop?
2. Modify the program in Experiment 3 so that it can produce output like the following display

```
Enter the number of students: 6
Enter student score 1: 75
Enter student score 2: 68
Enter student score 3: 83
Enter student score 4: 92
Enter student score 5: 88
Enter student score 6: 70
The class average score is 79.33333333333333
```

3. Assignment

Edit with WPS Office

1. Create a program that has an array of 5 elements. Then use the input to fill in the array elements, and display the contents of the array in reverse order as in the following illustration.



2. Create a program that accepts the number of array elements as input, also input the elements of array. Then displays the largest number of the array elements. Examples of program results are as follows:

```
Enter the number of array elements: 4
Enter the value of element 1: 27
Enter the value of element 2: 8
Enter the value of element 3: 33
Enter the value of element 4: 11
The largest number is 33
```

3. Create a program that accepts the number of array elements as input, also input the elements of array. Then displays which numbers are even and which are odd numbers. Examples of program results are as follows:

```
Enter the number of array elements: 6
Enter the value of element 1: 7
Enter the value of element 2: 3
Enter the value of element 3: 5
Enter the value of element 4: 8
Enter the value of element 5: 2
Enter the value of element 6: 1
Even number: 8
Even number: 2
Odd number: 7
Odd number: 3
Odd number: 5
Odd number: 1
```


Edit with WPS Office