Lab Assignment 8

Arrays

CSE110: Programming Language I

No of Tasks			Points to Score
Classwork	**Evaluation**	**Homework**	**Homework** **9*10 = 90** **Assessment** **2*10 = 20**
4	**2**	**9**	

The students must complete the classwork tasks in the lab class to obtain the lab performance marks. They will also be marked based on the assessment tasks. The lab instructors may show/explain a few of the classwork tasks to the students if necessary. Any plagiarism in classwork or homework will lead to the student getting zero in the entire assignment. A random viva may take place.

Classwork

1. Write a Java program that will take N integer numbers from the user and create an array of length N.
    a. Print the elements of the array with their indices.
    b. Take another integer input from the user, resize the array by length 1, and add the new integer value to the array. Print the resized array.

Sample Input	Sample Output
N = 5 Enter a number: 11 Enter a number: 22 Enter a number: 33 Enter a number: 44 Enter a number: 55 Enter another number: 101	The elements of the array are: 0: 11 1: 22 2: 33 3: 44 4: 55 After resizing the array: 11 22 33 44 55 101

2. You are given an integer array with duplicate values. Write a Java program to update the array by replacing the duplicate values of the array with zero. Then print the updated array. [Your code should work for any given integer array]

Given Array	Sample Output
int arr [] = {9, -5, 7, 9, -5, 5, 7};	Before removing duplicates: 9 -5 7 9 -5 5 7 After replacing duplicates with 0: 9 -5 7 0 0 5 0

3. Write a Java program that asks the user for the length of an array and then creates an integer array of that length by taking inputs from the user. Then,

a. Reverse the array by creating a new array of the same length and print it. (Out-of Place)

b. Reverse the array **without** creating any new arrays and print it. (In-Place)

Sample Input	Sample Output
Enter the length of the array: 5 Enter a number: 7 Enter a number: -31 Enter a number: 344 Enter a number: 97 Enter a number: 100	Reversed using a new array: 100 97 344 -31 7 Reversed the original array: 100 97 344 -31 7

4. Trace the following code, create a tracing table, and write the outputs.

```
1  class TraceA{
2      public static void main(String args[]) {
3          int [] myArray = new int[5];
4          int [] b;
5          int idx1 = 0, idx2 = 0;
6          b =  myArray;
7          while (idx1 < 5){
8              myArray[idx1] = idx1 + 11;
9              idx2 = 1;
10             while (idx2 < idx1 ){
11                 myArray[idx1] = b[idx2 - 1] + myArray[idx2] - idx1;
12                 idx2 = idx2 + 1;
13             }
14             System.out.println(myArray[idx2]);
15             idx1 = idx1 + 1;
16         }
17     }
18 }
```

Evaluation tasks

1. Take an integer N input from the user and create an integer array of N numbers by taking inputs from the user. Then, print the array. Next, **modify the array** by changing the positive numbers by 1 and the negative numbers by 0. If the element is zero, then it will be unchanged. Lastly, print the modified array.

Sample Input	Sample Output
N = 4 3 4 -2 1	Original array: 3 4 -2 1 After modifying: 1 1 0 1
N = 3 -4 0 2	Original array: -4 0 2 After modifying: 0 0 1

2. Write a Java program that will take N integer numbers from the user and create an array of length N. Take another number from the user and print the index of the number where it is found first. If not found then print 'Element not found'.
   Note: Think about how to apply the concept of **flag** and **break** in this task.

Sample Input 1	**Sample Input 2**
N = 7 Enter a number: 45 Enter a number: 0 Enter a number: 17 Enter a number: 11 Enter a number: -34 Enter a number: -100 Enter a number: 17 17	N = 5 Enter a number: 4 Enter a number: 99 Enter a number: 23 Enter a number: -67 Enter a number: 34 55
Sample Output 1 17 is at index 2	**Sample Output 2** Element not found

Homework

1. Write a Java program that asks the user for the length of an array then creates a double datatype array of that length by taking inputs from the user.
   Then do the following:
   a. Show the maximum element and its index from the array.
   b. Show the minimum element and its index from the array.
   c. Show the summation of all the elements from the array.
   d. Show the average of all the elements from the array.

Sample Input	Sample Output
Enter the length of the array: 5 Enter a number: 7.5 Enter a number: -31.2 Enter a number: 344.0 Enter a number: 97.1 Enter a number: 100.4	Maximum element 344.0 found at index 2 Minimum element -31.2 found at index 1 Summation: 517.8 Average: 103.56

2. You are given an integer array. You need to create a new array that will contain only the unique elements of the given array. Finally, print the new array.

Sample Input	Sample Output
int arr [] = {23,100,23,56,100};	Input array: 23 100 23 56 100 New array: 23 100 56
int arr [] = {-5,10,-7,-5};	Input array: -5 10 -7 -5 New array: -5 10 -7

3. Write a Java program that will take input of two arrays and elements from the user and check whether the second array is a subset of the first array. A subset is a set that contains only elements found in the original set.

Sample Input - 1	Sample Output - 1
Please enter the length of array 1: 5 Please enter the elements of the arr1: 5 3 2 72 8 Please enter the length of array 2: 3 Please enter the elements of the arr2: 5 3 72	Array 2 is a subset of Array 1.

Sample Input - 2	Sample Output - 2
Please enter the length of array 1: 5 Please enter the elements of the arr1: 7 2 33 1 6 Please enter the length of array 2: 3 Please enter the elements of the arr2: 1 8 2	Array 2 is not a subset of Array 1.

4. Write a Java program that asks the user the length of an array then takes that many integer numbers as elements for the array as input. Then the program asks the user to enter the target value. The program should add exactly two elements of the array to find the target value and their corresponding indices. The program should print the combination of elements that add up to give the target value and its corresponding index. If the target value does not exist, print "Target value not found". [Only consider the first pair if multiple pairs exist.]

Sample Input - 1	Sample Output - 1
Please enter the length of array: 5 Please enter the elements of the array: 3 8 5 4 1 Please enter the target value: 9	Elements need to be added: 8 1 Index of the elements: 1 and 4

Sample Input - 2	Sample Output - 2
Please enter the length of array: 4 Please enter the elements of the array: 3 6 4 1 Please enter the target value: 6	Target value not found

5. Take an integer N input from the user and create an array of length N by taking the elements as user input. Then, print the array. Next, sort the array in **descending** order using **Selection Sort** technique. Lastly, print the sorted array.

Sample Input	Sample Output
N = 6 Enter a number: 3 Enter a number: 6 Enter a number: 1 Enter a number: 2 Enter a number: 8 Enter a number: 5	Original Array: 3 6 1 2 8 5 Sorted Array: 8 6 5 3 2 1

6. You are given two arrays of the same length. The first array contains the marks of the students and the second array contains the name of the students. You need to sort the marks array in **ascending order** while maintaining the corresponding names of the students in the names array aligned with their respective marks. Use the **Bubble sort** technique to solve this problem. [Your code should work for any given arrays]

Given Array 1	Sample Output 1
int [] marks = {85, 90, 75, 44, 99}; String [] names = {"Bob", "Alice", "Max", "Marry", "Rosy"};	Sorted Array: 44 75 85 90 99 Marry Max Bob Alice Rosy

Given Array 2	Sample Output 2
int [] marks = {100, 47, 85, 94, 5, 50}; String [] names = {"Henry", "Mari", "Herry", "Jack", "Lily", "Oliver"};	Sorted Array: 5 47 50 85 94 100 Lily Mari Oliver Herry Jack Henry

7. Trace the following code, create a tracing table, and write the outputs.

```
1  class TraceB {
2    public static void main(String args[]){
3      int [] myArray = new int[10];
4      int index1 = 0, index2 =0;
5      while (index1 < 10){
6        myArray[index1] = index1 + 3;
7        index2 = 1;
8        while (index2 < index1 ){
9          myArray[index1] = myArray[index1] + myArray[index2] - index1;
10         index2 = index2 + 1;
11       }
12       System.out.println(myArray[index1]);
13       index1 = index 1 + 1;
14     }
15   }
16 }
```

8. Trace the following code, create a tracing table, and write the outputs.

```
1  class TraceC {
2    public static void main(String args[]){
3      int [] myArray = new int[10];
4      int [] b;
5      int index1 = 0, index2 =0;
6      index1 = 0;
7      b =  myArray;
8      while (index1 < 10){
9        myArray[index1] = index1 + 4;
10       index2 = 1;
11       while (index2 < index1 ){
12         myArray[index1] = b[index1] + myArray[index2] - index1;
13         index2 = index2 + 1;
14       }
15       System.out.println(myArray[index1]);
16       index1 = index1 + 1;
17     }
18   }
19 }
```

9. Trace the following code, create a tracing table, and write the outputs.

```
1  class TraceD{
2    public static void main(String args[]){
3       int[] arr1 = {3,2,0,1,5,6,7};
4       int[] arr2 = {30,20,40,11,55,-34,100};
5       int a1 = 0, a2 = 0;
6       while (a1<arr1.length-1){
7           arr2[a1] = arr1[a2]+ a1 - arr2[a2];
8           a2 = 1;
9           while (a2 < a1 ){
10             arr2[a1] = arr2[a1] + arr1[a2] - a2;
11             a2 = a2 + 1;
12          }
13          System.out.println(arr2[a1]);
14          a1 = a1 + 1;
15      }
16      System.out.println(arr2[arr1[a2]]);
17   }
18 }
```