CSE 1001: Introduction to Computer Programming

Programming Assignment-VIII

(Single-Dimensional Arrays)

1. Write a java program to create an array of size *N* and store the random values in it and find the sum and average.

2. Write a java program using an array 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: 2 5 6 5 4 3 23 43 2 0
2 occurs 2 times
3 occurs 1 time
4 occurs 1 time
5 occurs 2 times
6 occurs 1 time
23 occurs 1 time
43 occurs 1 time
```

   Note that if a number occurs more than one time, the plural word "times" is used in the output.

3. Input 10 integers from the keyboard into an array. The number to be searched is entered through the keyboard by the user. Write a java program to find if the number to be searched is present in the array and if it is present, display the number of times it appears in the array.

4. Write a method that finds the smallest element in an array of double values using the following header:

   public static double min(double[] array)

   Write a java program that prompts the user to enter ten numbers, invokes this method to return the minimum value, and displays the minimum value. Here is a sample run of the program:

```
Enter ten numbers: 1.9 2.5 3.7 2 1.5 6 3 4 5 2
The minimum number is: 1.5
```

5. Write a java program to find the second largest value in an array of *n* elements.

6. Write a java program that implements the array reversal algorithm suggested in *Note 1*.

   Note 1: There is a simpler algorithm for array reversal that starts out with two indices, i=0 and j=n-1. With each iteration i is increased and j is decreased for i<j.

7. Write a java program to convert a decimal integer to its corresponding octal representation.

8. Design and develop a menu driven java program for the following array operations.

   a. Create an array of *N* integers
   b. Display the array elements
   c. Insert an element at specific position
   d. Delete an element at a given position
   e. Exit

9. You can compute the standard deviation with the following formula; you have to store the individual numbers using an array, so that they can be used after the mean is obtained.

$$mean = \frac{\sum_{i=1}^{n} x_i}{n} = \frac{x_1 + x_2 + x_3 + \cdots + x_n}{n}$$

$$deviation = \sqrt{\frac{\sum_{i=1}^{n}(x_i - mean)^2}{n-1}}$$

Your program should contain the following methods:

/** Compute the deviation of double values */

```
public static double deviation(double[] x)
```

/** Compute the mean of an array of double values */

```
public static double mean(double[] x)
```

Write a java program that prompts the user to enter ten numbers and displays the mean and standard deviation, as shown in the following sample run:

```
Enter ten numbers: 1.9 2.5 3.7 2 1 6 3 4 5 2
The mean is 3.11
The standard deviation is 1.55738
```

10. Write a method that returns a new array by eliminating the duplicate values in the array using the following method header:

```
public static int[] eliminateDuplicates(int[] list)
```

Write a java program that reads in ten integers, invokes the method, and displays the result. Here is the sample run of the program:

```
Enter ten numbers: 1 2 3 2 1 6 3 4 5 2
The distinct numbers are: 1 2 3 6 4 5
```

11. Write a sort method that uses the ***bubble-sort*** algorithm. The bubble sort algorithm makes several passes through the array. On each pass, successive neighbouring pairs are compared. If a pair is not in order, its values are swapped; otherwise, the values remain unchanged. The technique is called a *bubble sort* or *sinking sort* because the smaller values gradually "bubble" their way to the top and the larger values "sink" to the bottom. Write a java program that reads in ten double numbers, invokes the method, and displays the sorted numbers.

12. The ***selection-sort*** method repeatedly finds the smallest number in the current array and swaps it with the first. Write a java program that reads in ten integer values, invoke the method, and displays the sorted elements.

13. Write the following method that returns true if the list is already sorted in increasing order.

```
public static boolean isSorted(int[] list)
```

Write a java program that prompts the user to enter a list and displays whether the list is sorted or not. Here is a sample run. Note that the first number in the input indicates the number of the elements in the list. This number is not part of the list. Here is the sample run:

```
Enter list: 8 10 1 5 16 61 9 11 1
The list is not sorted
Enter list: 10 1 1 3 4 4 5 7 9 11 21
The list is already sorted
```

14. Write a java program that randomly generates an array of 100 integers and a key. Estimate the execution time of invoking the ***linearSearch*** method. Sort the array and estimate the execution time of invoking the ***binarySearch*** method. You can use the following code template to obtain the execution time:

```
long startTime = System.currentTimeMillis();

perform the task;
....
....
long endTime = System.currentTimeMillis();

long executionTime = endTime - startTime;
```

15. Write the following method that partitions the list using the first element, called a *pivot*.

```
public static int partition(int[] list)
```

After the partition, the elements in the list are rearranged so that all the elements before the Pivot are less than or equal to the pivot and the elements after the pivot are greater than the pivot. The method returns the index where the pivot is located in the new list. For example, suppose the list is {5, 2, 9, 3, 6, 8}. After the partition, the list becomes {3, 2, 5, 9, 6, 8}.

Implement the method in a way that takes at most *list.length* comparisons. Write a java program that prompts the user to enter a list and displays the list after the partition.

Here is a sample run. Note that the first number in the input indicates the number of the elements in the list. This number is not part of the list.

```
Enter list: 8 10 1 5 16 61 9 11 1
After the partition, the list is 9 1 5 1 10 61 11 16
```

****