CS162

ASSIGNMENT 3

NAME:

ARCHIT AGRAWAL

ROLL NO. :

202052307

SECTION:

A

Question 1

1. **Code and compare the running time of Bubble and Selection Sort.**
   Do as directed:
   a. Take an array of at least 100 numbers in the range of 1000 – 10000.
   b. Implement Bubble Sort and Selection Sort.
   c. Make the following measurements:
      i. Time for sorting when numbers are in **random order**.
      ii. Time for sorting when numbers are in **ascending order**.
      iii. Time for sorting when numbers are in **descending order**.
   d. Repeat these measurements for 10 times and take out the average value.
   e. Fill the table below with your readings and attach it with your submission file.

Note: Do not use any built in sorting libraries.

S.No.	Sorting Algorithm	Number of elements in the Array	Average Time (in ms) Random Order	Average Time (in ms) Ascending Order	Average Time (in ms) Descending Order
1.	Bubble Sort				
2.	Selection Sort				

Hint:

A. Use the snippet below to calculate the time.

```
long start = System.nanoTime();
// your code goes here
long end = System.nanoTime();
long elapsedTime = end - start;
```

Note: This code calculates time in nanoseconds. You need to report the time in milliseconds upto 3 decimal places

B. You can also use the random number generator to generate 100 numbers instead of hard coding them

---

Description -: The following program asks the user to input the size of array and asks to decide in what order(ascending, descending or random) the user wants to pass the array to sorting methods (bubble sort, selection sort).

The array is generated using random function, hence it is not sorted.

- If the user wants a random sorted to be passed to the sorting methods(bubble sort, selection sort), this array can directly be passed.

- If the user wants an ascending sorted array, this array will be first sorted in ascending order and then it will be passed to the sorting methods.
- If the user wants a descending sorted array, this array will be first sorted in descending order and then it will be passed to the sorting methods.

The program will calculate time taken to sort the array for both bubble and selection sort 10 times and gives the output as average time taken to do so for both the sorting methods.

CODE

```
/* This code is written to compare the average time between selection sort
and bubble sort algorithms
   You will be asked to enter the size of array as input
   You will be asked to decide the order of input array
   an array will be generated randomly of that size and time will be
calculated for both the algorithms (this will take place 10 times)
   the average of the time for both the methods will be calculated and
printed
 */
package com.company;
import java.util.*;
import java.util.Random;
import java.util.Arrays;
import java.util.Collections;

public class SortingMethods{

    public static double bubbleSort(Integer []array, int n){
        double start = System.nanoTime();
        //Bubble Sort Algorithm
        for(int i = 0; i < n ; i++){
            for(int j = 0; j < n - i - 1; j++){
                if(array[j] > array[j + 1]){
                    int temp = array[j];
                    array[j] = array[j + 1];
                    array[j + 1] = temp ;
                }
            }
        }
        double end = System.nanoTime();
        return (end - start)/1000000.0;
    }

    public static double selectionSort(Integer []array, int n){
        double start = System.nanoTime();
```

```
        //Selection Sort Algorithm
        for(int i = 1; i <= n; i++){
            int greatest = array[0];
            int tempJ = 0;
            for(int j = 1; j <= n - i ; j++){
                if(array[j] > greatest ){
                    greatest = array[j] ;
                    tempJ = j ;
                }
            }
            int tempA = array[n - i] ;
            array[n - i] = greatest;
            array[tempJ] = tempA;
        }
        double end = System.nanoTime();
        return (end - start)/1000000.0;
    }

    public static void main(String []args){
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the size of array");
        int n = sc.nextInt();

        System.out.println("1. If you want the input array to be randomly
arranged, enter 1");
        System.out.println("2. If you want the input array to be arranged
in increasing order, enter 2");
        System.out.println("3. If you want the input array to be arranged
in descending order, enter 3");
        int order = sc.nextInt();

        double avg_time_bubble = 0.0;
        double avg_time_selection = 0.0;

        int t = 10;
        while(t > 0) {  //while loop is used to run bubble/selection sort
for different arrays and compute the average.
            Integer[] a = new Integer[n];
            Integer[] b = new Integer[n]; //a copy of a[]
            Random rand = new Random();

            for (int i = 0; i < n; i++) {
                a[i] = rand.nextInt(9000) + 1000;
                //System.out.print(a[i] + " ");
            }
            for(int i = 0; i < n; i++){
                b[i] = a[i];
            }
            //a[] will be bubble sorted
            //b[] will be selection sorted
            // as random will give a new number everytime that is why
            //a copy of a[] is created to ensure that both the sorting
            //methods gets the same array

            if (order == 2) {
                Arrays.sort(a);
                Arrays.sort(b);
            } else if (order == 3) {
                Arrays.sort(a, Collections.reverseOrder());
                Arrays.sort(b, Collections.reverseOrder());
            }
```

```
            double time_in_bubble = bubbleSort(a, n);
            double time_in_selection = selectionSort(b, n);
            avg_time_bubble = avg_time_bubble + time_in_bubble;
            avg_time_selection = avg_time_selection + time_in_selection;

            t = t - 1;
        }


System.out.println("**********************************************************
**************************************************************************
***************");
        System.out.printf("Average time taken in bubble sort in
milliseconds : %.3f ",(avg_time_bubble/10.0));
        System.out.println();
        System.out.printf("Average time taken in selection sort
milliseconds : %.3f ",(avg_time_selection/10.0));
    }
}
```

OUTPUT

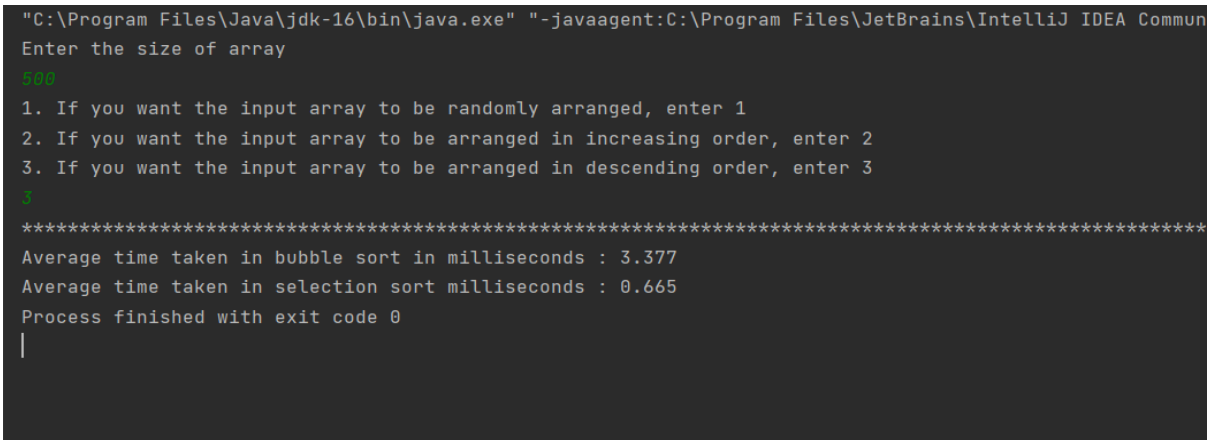
- ***For random ordered input array***

```
Enter the size of array
500
1. If you want the input array to be randomly arranged, enter 1
2. If you want the input array to be arranged in increasing order, enter 2
3. If you want the input array to be arranged in descending order, enter 3
1
**************************************************************************************************************
Average time taken in bubble sort in milliseconds : 2.923
Average time taken in selection sort milliseconds : 0.721
Process finished with exit code 0
```

- ***For ascending ordered input array***

```
"C:\Program Files\Java\jdk-16\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community E
Enter the size of array
500
1. If you want the input array to be randomly arranged, enter 1
2. If you want the input array to be arranged in increasing order, enter 2
3. If you want the input array to be arranged in descending order, enter 3
2
**************************************************************************************************************
Average time taken in bubble sort in milliseconds : 0.804
Average time taken in selection sort milliseconds : 0.998
Process finished with exit code 0
```

- ***For descending ordered input array***



The above data is tabularized below.

S.No.	Sorting Algorithm	Number of elements in the array	Average Time (in ms) for Random Order Input	Average Time (in ms) for Ascending Order Input	Average Time (in ms) for Descending Order Input
1.	Bubble Sort	500	2.923	0.804	3.377
2.	Selection Sort	500	0.721	0.998	0.665

Question 2

2. Pattern Printing:

Write a program to print the hourglass pattern using loops:

Note: Ask the user for the number of rows as an input.

Eg: if number of rows = 4, the pattern printed as follows

```
* * * *
 * * *
  * *
   *
  * *
 * * *
* * * *
```

CODE

```
package com.company;
import java.util.*;

public class Pattern {

    public static void hourglass(int rows){
        int n = 2 * rows - 1;
        int s;
        for(int i = 0; i < n; i++){
            if(i < rows) s = i;
            else s = n - i - 1;


            for(int j = 0; j < s; j++){
                System.out.print(" ");
            }
            for(int j = 0; j < rows - s; j++){
                System.out.print("* ");
            }

            System.out.println();
        }
```

```
    }

    public static void main(String []args){
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the number of rows");
        int n = sc.nextInt();
        hourglass(n);
    }
}
```

OUTPUT

```
"C:\Program Files\Java\jdk-16\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2
Enter the number of rows
7
* * * * * * *
 * * * * * *
  * * * * *
   * * * *
    * * *
     * *
      *
     * *
    * * *
   * * * *
  * * * * *
 * * * * * *
* * * * * * *

Process finished with exit code 0
```

Question 3

3. **Problem based on array**
Given an integer array, find whether it is pretty or not. We define an array is a pretty array if it begins with a 0 and all the rest of its elements are divisible by their respective indices.

Ex - Input: 0 2 6 21 16 15
Output: True
Explanation: each value is divisible by it's index.
Ex - Input: 0 2 3 21 16 15
Output: False
Explanation: value at index 2 (i.e., 3) is not divisible by 2.

CODE

```
package com.company;
import java.util.*;

public class PrettyArray{

    public static boolean prettyArray(int []array){

        if(array[0] != 0) return false;
        for(int i = 1; i < array.length; i++){
            if(array[i] % i != 0) return false;
        }
        return true;
    }

    public static void main(String [] args){
        Scanner sc = new Scanner(System.in);
        System.out.println("An array is a pretty array if it begins with a
0 and all the rest of its elements are divisible by their respective
indices(0 - based).");
        System.out.println();
        System.out.println("Enter the size of the array");
        int n = sc.nextInt();
        System.out.println("Enter array elements one-by-one with spaces");
        int[] array = new int[n];
        for(int i = 0; i < n; i++){
            array[i] = sc.nextInt();
        }
        if(prettyArray(array)) System.out.println("The given array is a
```

```
pretty array.");
        else System.out.println("The given array is not a pretty array.");
    }
}
```

OUTPUT

```
"C:\Program Files\Java\jdk-16\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2021.1\lib\idea_rt.jar=6
An array is a pretty array if it begins with a 0 and all the rest of its elements are divisible by their respective indices(0 - based).

Enter the size of the array
12
Enter array elements one-by-one with spaces
0 1 4 3 8 10 18 7 16 9 20 11 36
The given array is a pretty array.

Process finished with exit code 0
```

```
"C:\Program Files\Java\jdk-16\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2021.1\lib\idea_rt.jar=60134
An array is a pretty array if it begins with a 0 and all the rest of its elements are divisible by their respective indices(0 - based).

Enter the size of the array
12
Enter array elements one-by-one with spaces
0 1 3 3 8 10 18 7 16 9 20 11 36
The given array is not a pretty array.

Process finished with exit code 0
```

```
"C:\Program Files\Java\jdk-16\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2021.1\lib\idea_rt.jar
An array is a pretty array if it begins with a 0 and all the rest of its elements are divisible by their respective indices(0 - based).

Enter the size of the array
12
Enter array elements one-by-one with spaces
1 1 2 6 8 5 18 7 8 18 20 11 24
The given array is not a pretty array.

Process finished with exit code 0
```