

INFO 6205

Program Structures & Algorithms

Fall 2020

Assignment No: 2

- **Task:** To implement the class Timer.java and InsertionSort.java and also measure the running times of the sort using four different initial array ordering situations i.e. Random, Ordered, Partially-Ordered and Reverse-ordered array.
- **Output:** Below is the output received for 5 different values of n with different initial array ordering situations ran 10 times each and doubling the value of n each time-

```

workspace-INFO6205-PSA - INFO6205/src/main/java/edu/neu/coe/info6205/utl/Benchmark_Timer.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help

Package Explorer:
  workspace-INFO6205-PSA
  edu.neu.coe.info6205
  edu.neu.coe.info6205.algorithm
  edu.neu.coe.info6205.bsp
  edu.neu.coe.info6205.coupling
  edu.neu.coe.info6205.equable
  edu.neu.coe.info6205.functions
  edu.neu.coe.info6205.graphs
  edu.neu.coe.info6205.graphs.gis
  edu.neu.coe.info6205.graphs.huntwells
  edu.neu.coe.info6205.graphs.undirected
  edu.neu.coe.info6205.greedy
  edu.neu.coe.info6205.hashable
  edu.neu.coe.info6205.lite
  edu.neu.coe.info6205.life.base
  edu.neu.coe.info6205.life.library
  edu.neu.coe.info6205.pq
  edu.neu.coe.info6205.randomwalk
  edu.neu.coe.info6205.reduction
  edu.neu.coe.info6205.run.lengthEncoding
  edu.neu.coe.info6205.sort
    Benchmark_Timer.java
    Benchmark.java
    Config.java
    Fast.java
    FileHandler.java
    FileHandlerImpl_CSV.java
    KeyCopy.java
    Range.java
    SortBenchmark.java
    SortBenchmarkHelper.java
    SortBenchmarkHelper.java

src/main/java
  edu.neu.coe.info6205
    utl
      Benchmark_Timer.java

src/main/java/edu/neu/coe/info6205/utl/Benchmark_Timer.java
231  int disorderMap = (int) (0.2 * N);
232
233  //Generate random indices to create disorder and replace it
234  for(int i = 0; i < disorderMap; i++) {
235      int index = random.nextInt(N);
236      arr[index] = random.nextInt("1000");
237  }
238  return arr;
239
240

Problems: 0 Javadoc: 0 Declaration: 0 Console: 0
<terminated> Benchmark_Timer [Java Application] C:\Program Files\Java\jdk-8.0.261\bin\java.exe (27-Sep-2020, 2:06:53 pm)
2020-09-27 14:06:54 INFO Benchmark_Timer - Begin run: Benchmark Test with 10 runs
Value of N: 500 Order Situation- Randomly Ordered Time Taken: 1.5
2020-09-27 14:06:54 INFO Benchmark_Timer - Begin run: Benchmark Test with 10 runs
Value of N: 1000 Order Situation- Randomly Ordered Time Taken: 1.8
2020-09-27 14:06:54 INFO Benchmark_Timer - Begin run: Benchmark Test with 10 runs
Value of N: 2000 Order Situation- Randomly Ordered Time Taken: 5.9
2020-09-27 14:06:54 INFO Benchmark_Timer - Begin run: Benchmark Test with 10 runs
Value of N: 4000 Order Situation- Randomly Ordered Time Taken: 28.8
2020-09-27 14:06:54 INFO Benchmark_Timer - Begin run: Benchmark Test with 10 runs
Value of N: 8000 Order Situation- Randomly Ordered Time Taken: 111.7
-----
2020-09-27 14:06:56 INFO Benchmark_Timer - Begin run: Benchmark Test with 10 runs
Value of N: 500 Order Situation- Ordered Time Taken: 0.0
2020-09-27 14:06:56 INFO Benchmark_Timer - Begin run: Benchmark Test with 10 runs
Value of N: 1000 Order Situation- Ordered Time Taken: 0.0
2020-09-27 14:06:56 INFO Benchmark_Timer - Begin run: Benchmark Test with 10 runs
Value of N: 2000 Order Situation- Ordered Time Taken: 0.0
2020-09-27 14:06:56 INFO Benchmark_Timer - Begin run: Benchmark Test with 10 runs
Value of N: 4000 Order Situation- Ordered Time Taken: 0.1
2020-09-27 14:06:56 INFO Benchmark_Timer - Begin run: Benchmark Test with 10 runs
Value of N: 8000 Order Situation- Ordered Time Taken: 0.1
-----
2020-09-27 14:06:56 INFO Benchmark_Timer - Begin run: Benchmark Test with 10 runs
Value of N: 500 Order Situation- Partially Ordered Time Taken: 0.2
2020-09-27 14:06:56 INFO Benchmark_Timer - Begin run: Benchmark Test with 10 runs
Value of N: 1000 Order Situation- Partially Ordered Time Taken: 0.7
2020-09-27 14:06:56 INFO Benchmark_Timer - Begin run: Benchmark Test with 10 runs
Value of N: 2000 Order Situation- Partially Ordered Time Taken: 2.7
2020-09-27 14:06:56 INFO Benchmark_Timer - Begin run: Benchmark Test with 10 runs
Value of N: 4000 Order Situation- Partially Ordered Time Taken: 10.8
2020-09-27 14:06:56 INFO Benchmark_Timer - Begin run: Benchmark Test with 10 runs
Value of N: 8000 Order Situation- Partially Ordered Time Taken: 27.5
-----
2020-09-27 14:06:57 INFO Benchmark_Timer - Begin run: Benchmark Test with 10 runs
Value of N: 500 Order Situation- Reverse Ordered Time Taken: 0.0
2020-09-27 14:06:57 INFO Benchmark_Timer - Begin run: Benchmark Test with 10 runs
Value of N: 1000 Order Situation- Reverse Ordered Time Taken: 0.3
2020-09-27 14:06:57 INFO Benchmark_Timer - Begin run: Benchmark Test with 10 runs
Value of N: 2000 Order Situation- Reverse Ordered Time Taken: 10.4
2020-09-27 14:06:57 INFO Benchmark_Timer - Begin run: Benchmark Test with 10 runs
Value of N: 4000 Order Situation- Reverse Ordered Time Taken: 67.3
2020-09-27 14:06:58 INFO Benchmark_Timer - Begin run: Benchmark Test with 10 runs
Value of N: 8000 Order Situation- Reverse Ordered Time Taken: 233.2

```

Console Output-

```

2020-09-27 14:06:54 INFO Benchmark_Timer - Begin run: Benchmark Test with 10 runs
Value of N: 500 Order Situation- Randomly Ordered Time Taken: 1.5
2020-09-27 14:06:54 INFO Benchmark_Timer - Begin run: Benchmark Test with 10 runs
Value of N: 1000 Order Situation- Randomly Ordered Time Taken: 1.8
2020-09-27 14:06:54 INFO Benchmark_Timer - Begin run: Benchmark Test with 10 runs
Value of N: 2000 Order Situation- Randomly Ordered Time Taken: 5.9
2020-09-27 14:06:54 INFO Benchmark_Timer - Begin run: Benchmark Test with 10 runs
Value of N: 4000 Order Situation- Randomly Ordered Time Taken: 28.8
2020-09-27 14:06:54 INFO Benchmark_Timer - Begin run: Benchmark Test with 10 runs

```

Value of N: 8000 Order Situation- Randomly Ordered Time Taken: 111.7

 2020-09-27 14:06:56 INFO Benchmark_Timer - Begin run: Benchmark Test with 10 runs
 Value of N: 500 Order Situation- Ordered Time Taken: 0.0

2020-09-27 14:06:56 INFO Benchmark_Timer - Begin run: Benchmark Test with 10 runs
 Value of N: 1000 Order Situation- Ordered Time Taken: 0.0

2020-09-27 14:06:56 INFO Benchmark_Timer - Begin run: Benchmark Test with 10 runs
 Value of N: 2000 Order Situation- Ordered Time Taken: 0.0

2020-09-27 14:06:56 INFO Benchmark_Timer - Begin run: Benchmark Test with 10 runs
 Value of N: 4000 Order Situation- Ordered Time Taken: 0.1

2020-09-27 14:06:56 INFO Benchmark_Timer - Begin run: Benchmark Test with 10 runs
 Value of N: 8000 Order Situation- Ordered Time Taken: 0.1

 2020-09-27 14:06:56 INFO Benchmark_Timer - Begin run: Benchmark Test with 10 runs
 Value of N: 500 Order Situation- Partially Ordered Time Taken: 0.2

2020-09-27 14:06:56 INFO Benchmark_Timer - Begin run: Benchmark Test with 10 runs
 Value of N: 1000 Order Situation- Partially Ordered Time Taken: 0.7

2020-09-27 14:06:56 INFO Benchmark_Timer - Begin run: Benchmark Test with 10 runs
 Value of N: 2000 Order Situation- Partially Ordered Time Taken: 2.7

2020-09-27 14:06:56 INFO Benchmark_Timer - Begin run: Benchmark Test with 10 runs
 Value of N: 4000 Order Situation- Partially Ordered Time Taken: 10.8

2020-09-27 14:06:56 INFO Benchmark_Timer - Begin run: Benchmark Test with 10 runs
 Value of N: 8000 Order Situation- Partially Ordered Time Taken: 27.5

 2020-09-27 14:06:57 INFO Benchmark_Timer - Begin run: Benchmark Test with 10 runs
 Value of N: 500 Order Situation- Reverse Ordered Time Taken: 0.8

2020-09-27 14:06:57 INFO Benchmark_Timer - Begin run: Benchmark Test with 10 runs
 Value of N: 1000 Order Situation- Reverse Ordered Time Taken: 4.3

2020-09-27 14:06:57 INFO Benchmark_Timer - Begin run: Benchmark Test with 10 runs
 Value of N: 2000 Order Situation- Reverse Ordered Time Taken: 18.5

2020-09-27 14:06:57 INFO Benchmark_Timer - Begin run: Benchmark Test with 10 runs
 Value of N: 4000 Order Situation- Reverse Ordered Time Taken: 67.3

2020-09-27 14:06:58 INFO Benchmark_Timer - Begin run: Benchmark Test with 10 runs
 Value of N: 8000 Order Situation- Reverse Ordered Time Taken: 213.2

- **Relationship conclusion:** It can be concluded from the results of the benchmark test:
 - In best-case scenario which happens when the array is sorted, the insertion sort runs in $O(n)$ time as it simply compares the elements with no swapping required
 - In the average-case when the array is sorted in randomly-ordered or partially ordered manner the insertion sort runs in $O(n^2)$ time
 - Similarly, in the worst-case scenario, when the array is sorted in the reverse order manner the insertion sort takes $O(n^2)$ time because for inserting the last element, the algorithm will need $n-1$ comparisons and perform $n-1$ swaps and so on.

Therefore, we get - $2(1+2+3+.....+n-2+n-1)$

$= n(n-1)$ [Using Sum of n formula]

$= n^2 - n$

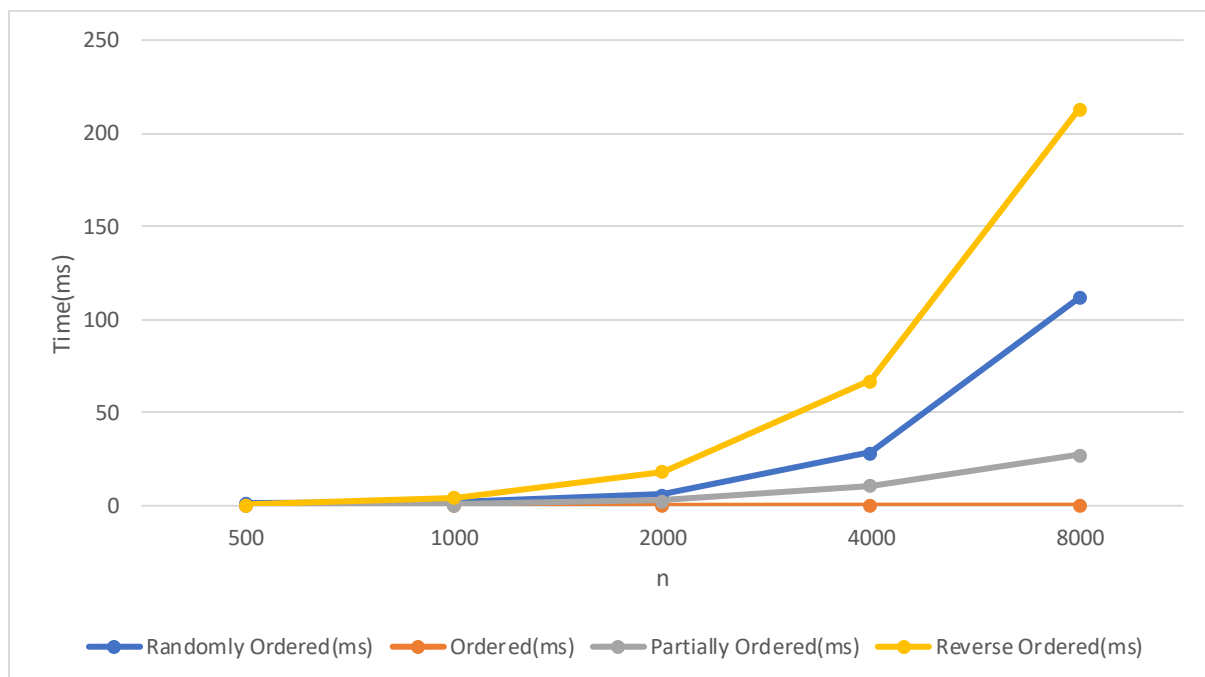
$\sim n^2$

- Thus, it can be derived that the following pattern is followed by the run time of insertion sort for different order situation in arrays-

Ordered < Partially Ordered < Randomly Ordered < Reverse Ordered

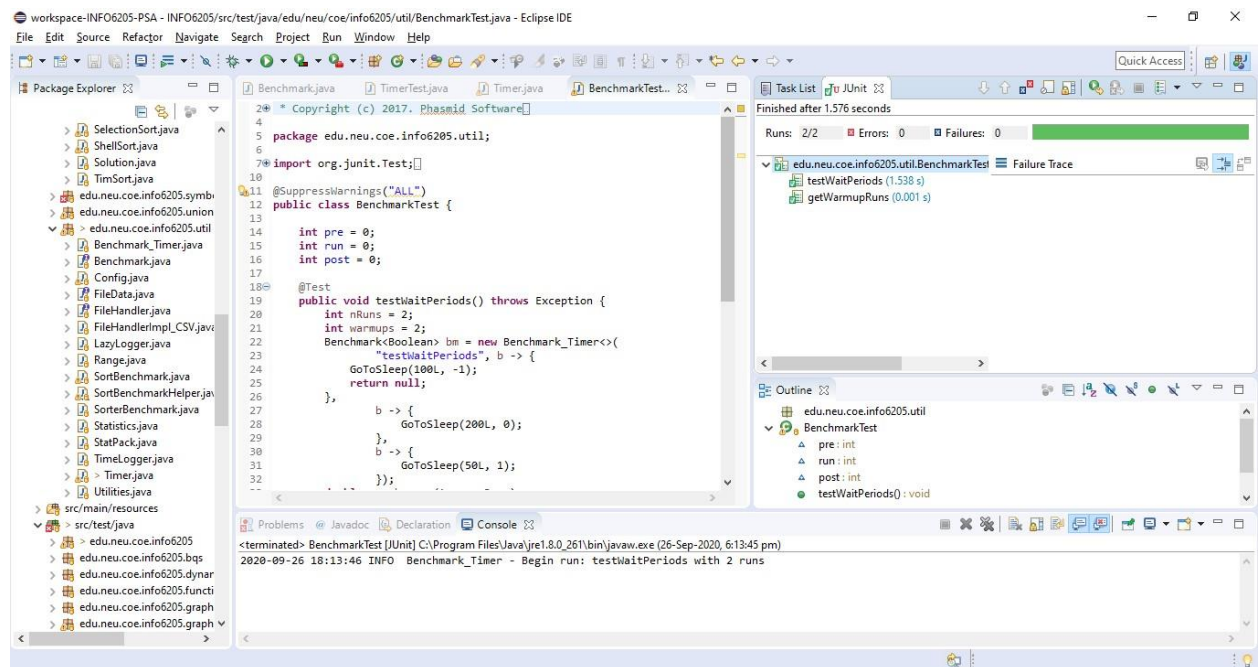
- **Evidence to support relationship:** I have attached a chart and a table stating the data of the different output observed for the different set of inputs of n. As a result, we can see the proportionate increase in the run time of insertion sort with different sorted situation of arrays-

Value of n	Randomly Ordered(ms)	Ordered(ms)	Partially Ordered(ms)	Reverse Ordered(ms)
500	1.5	0.0	0.2	0.8
1000	1.8	0.0	0.7	4.3
2000	5.9	0.0	2.7	18.5
4000	28.8	0.1	10.8	67.3
8000	111.7	0.1	27.5	213.2



- **Screenshot of Unit test passing:** Below is the screenshot of all the unit tests which ran successfully

BenchmarkTest



TimerTest

