SHREEKARA SS (001545668)

**Program Structures & Algorithms**

**Fall 2021**

**Assignment No. 2**

* **Task**

**Timer.java:**

1. Worked on repeat function to return the meanlaptime in milliseconds with pre and post functions validations
2. Worked on getClock function which return the system time in nanoseconds.
3. Worked on toMillisecs function to convert ticks into milliseconds.

InsertionSort.java:

1. Worked on sort function which sorts the array of type X using swapConditional method present in Helper class.

InsertionSort\_Timer.java:

1. Created a class called InsertionSort\_BenchMark.java under elementary folder present in sort file.
2. Imported Timer class and InsertionSort class into this class.
3. Wrote a main function which is used to measure the running times of this sort, using four different initial array ordering situations: random, ordered, partially-ordered and reverse-ordered.

* **Conclusion: (For ex : z = a \* b)**

Considering the values of N and mean lap time from insertion sort tests, it is very clear that the relationship N and mean lap time . When considered with concurrent and increasing values of N, it is evident that this relationship holds good.

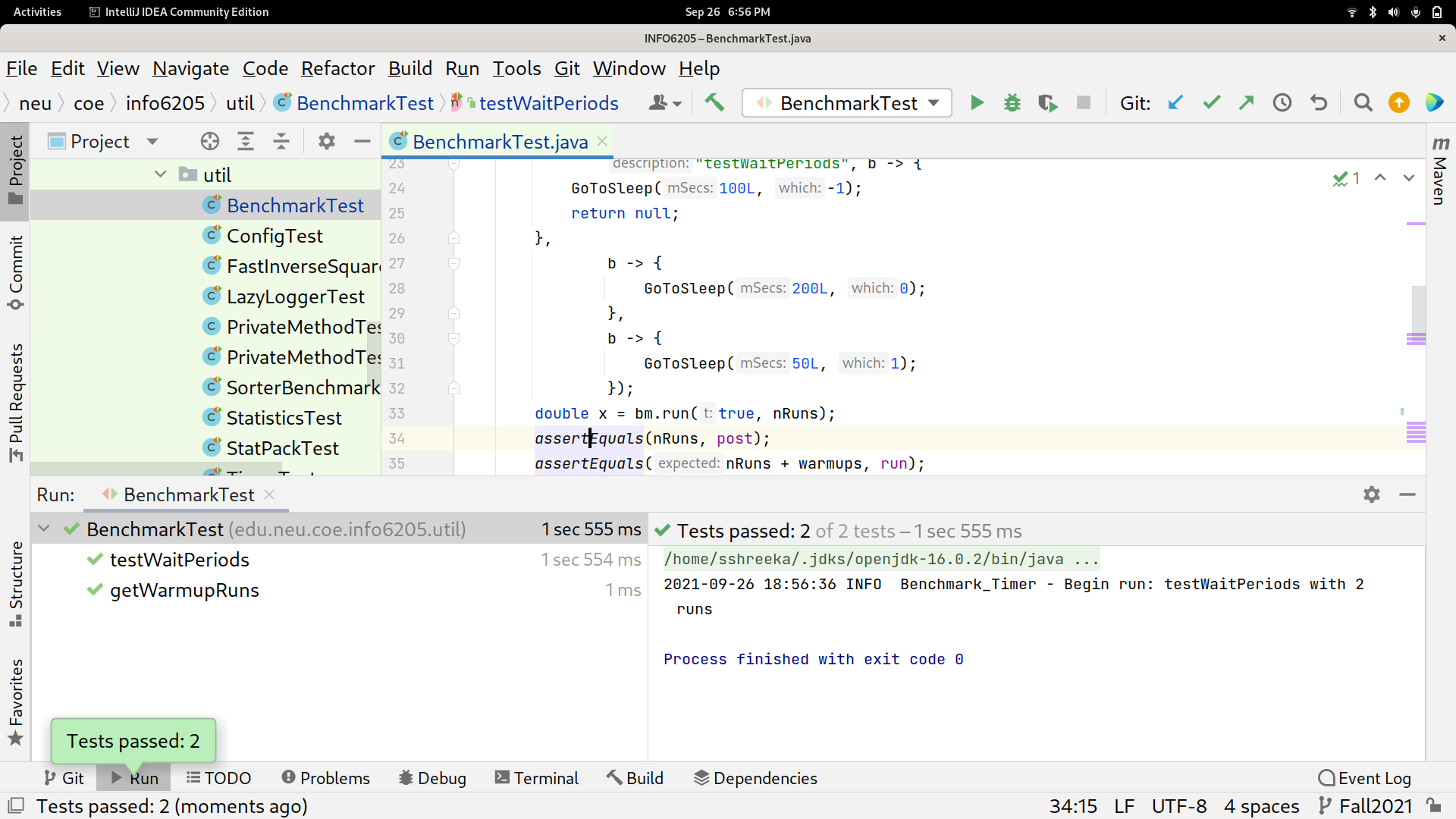
* + Since the reciprocal of N is not exactly equal to mean lap time obtained, it can be justified as an approximated value rather than exact value. In terms of mathematical terms

meanlaptime = 1/N

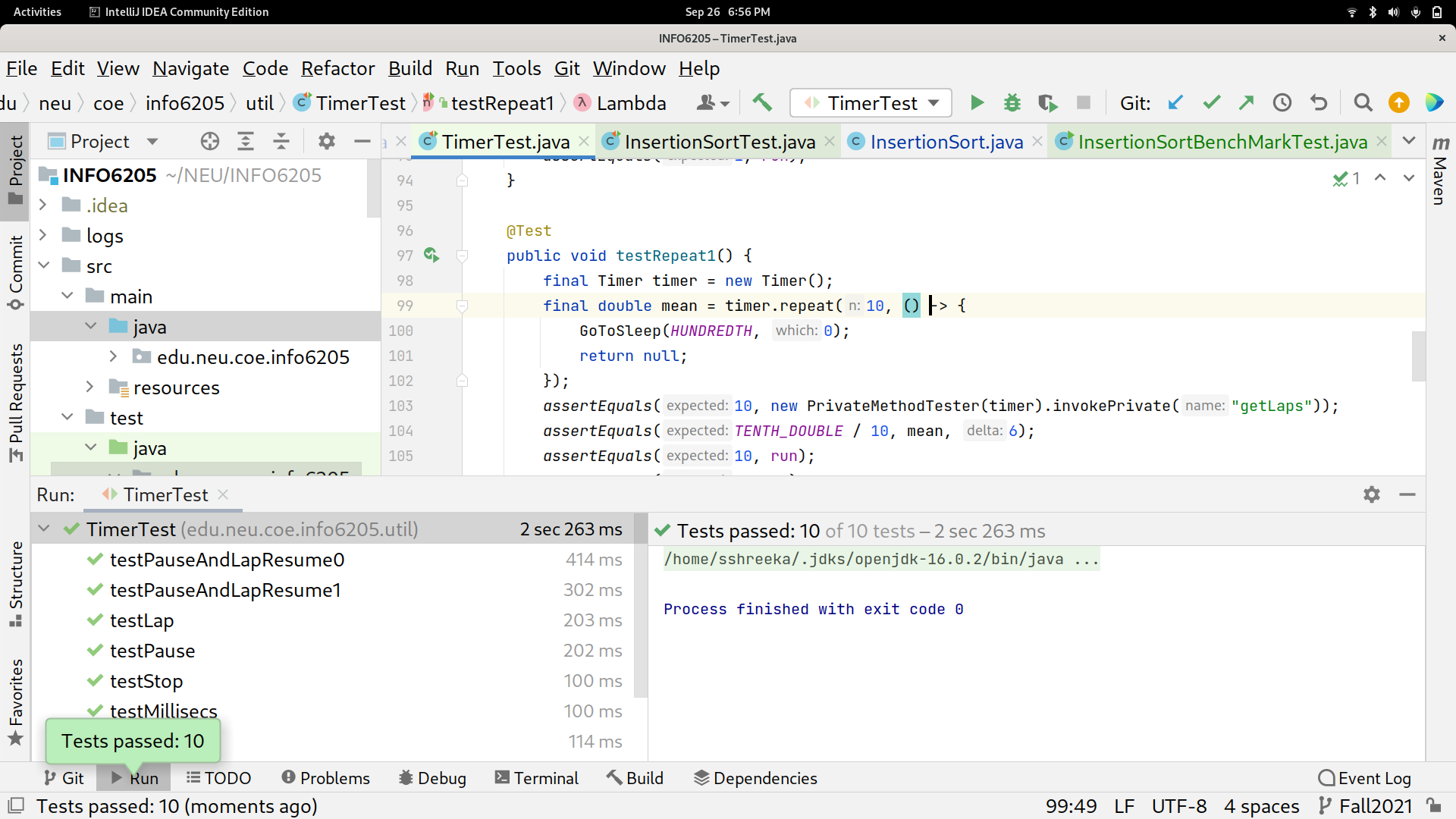
* + These values are calculated for the series of experiments and for each step value. Hence it can be concluded that this mathematical expression holds mostly good with all the experiments provided the given constraints remaining the same.
* **Evidence to support the conclusion:**

1. **Output:**

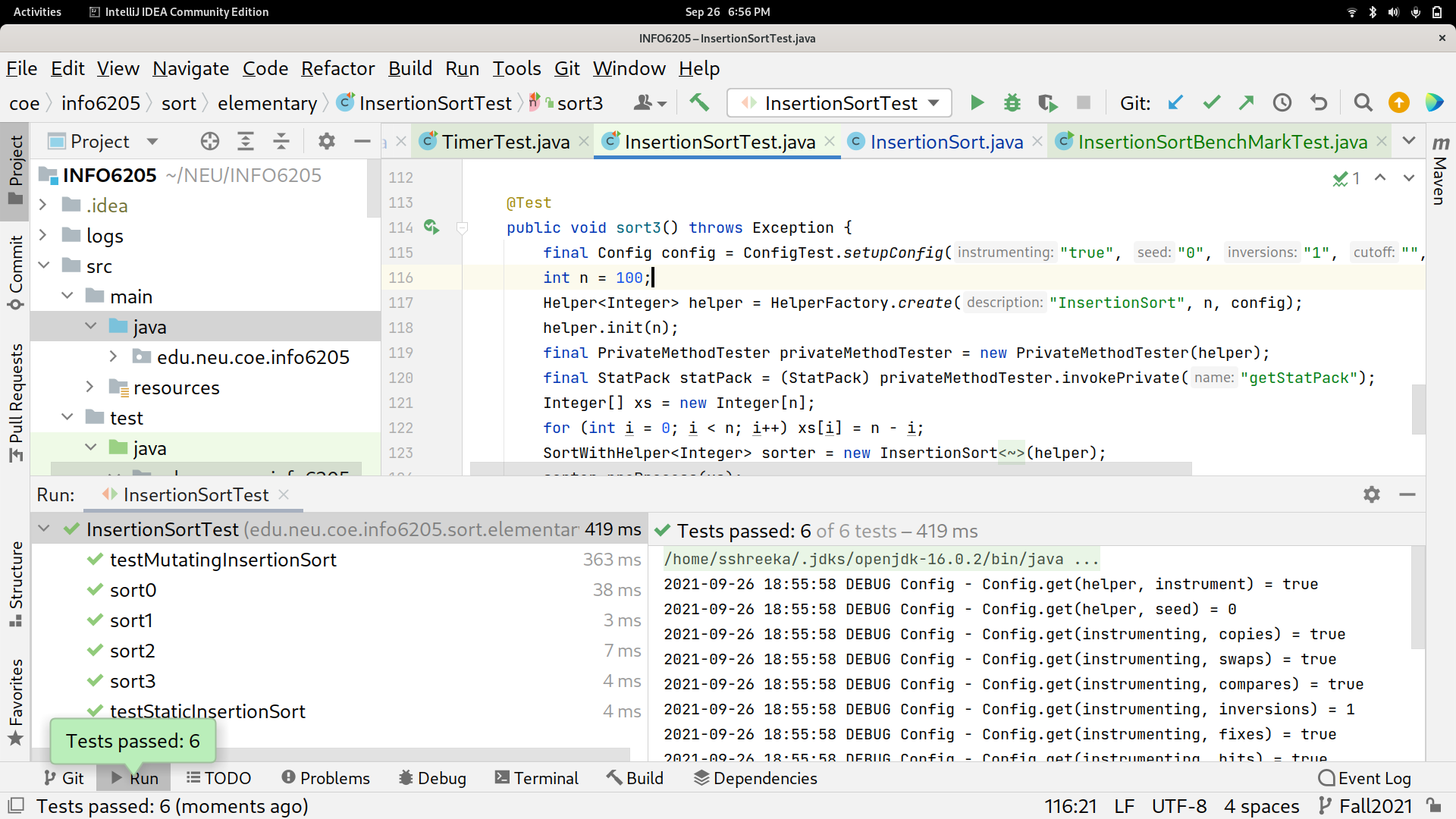
BenchmarkTest.java:

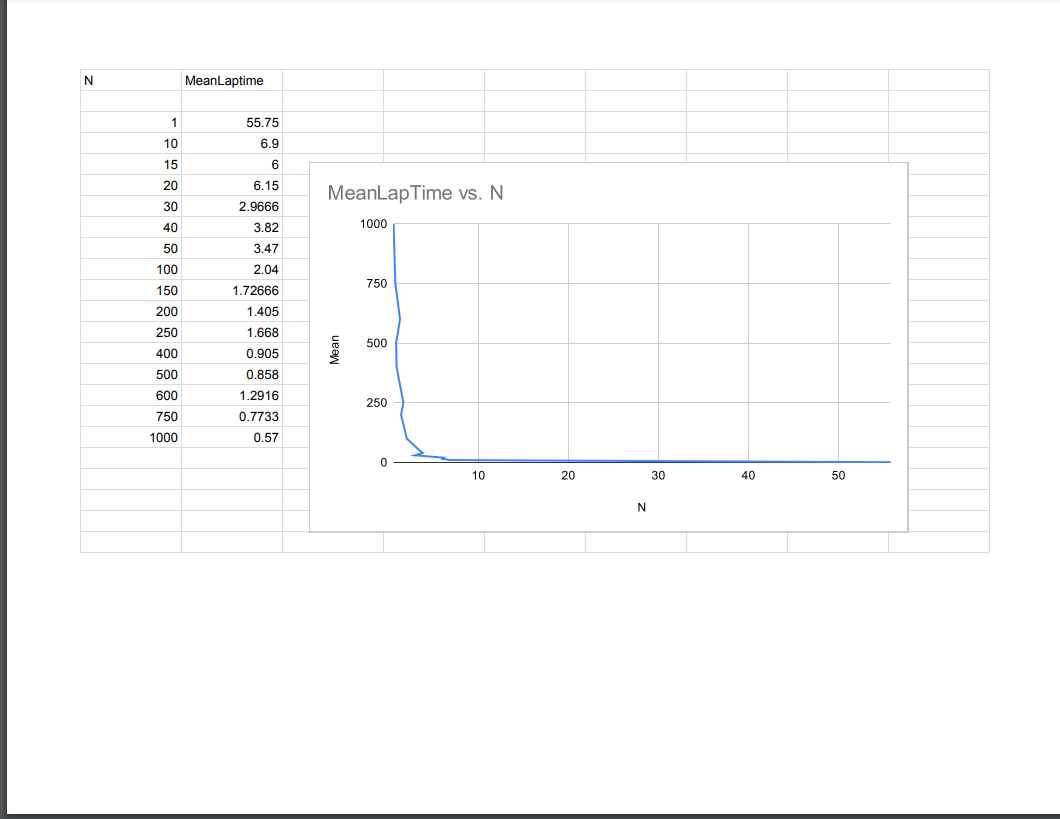
****

TimerTest.java:



InsertionSortTest.java:

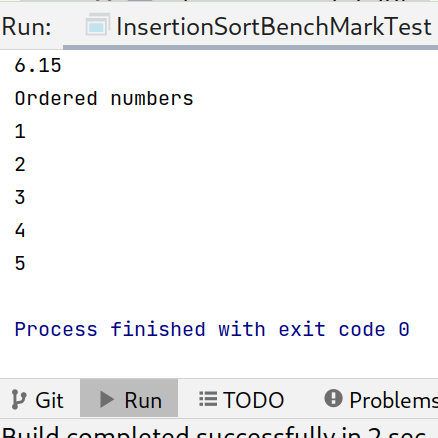
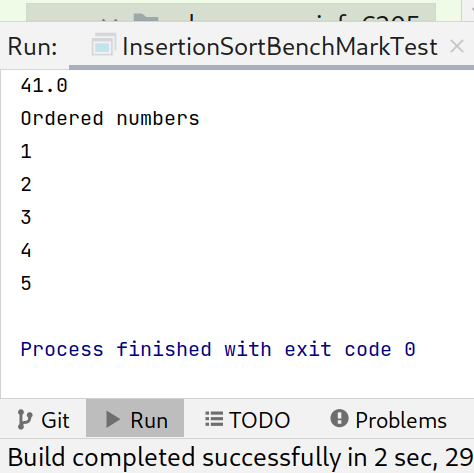


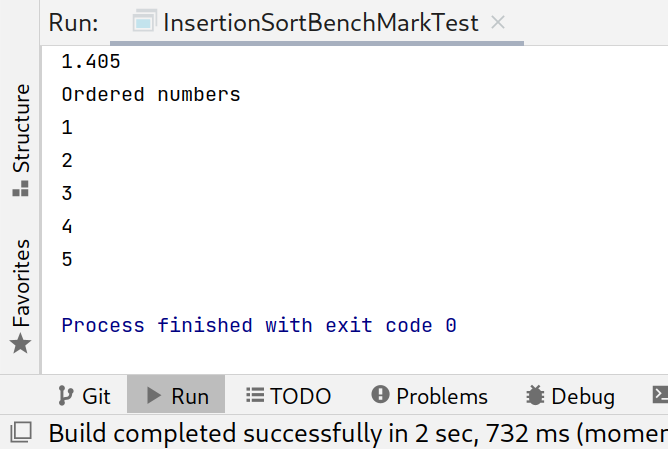
1. **Graphical Representation(Observations from experiments should be tabulated and analyzed by plotting graphs(usually in excel) to arrive on the relationship conclusion)**
2. 

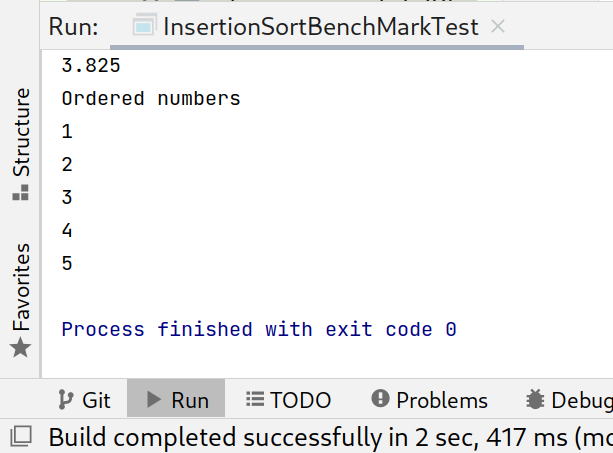
* **Unit tests result:**

**ORDERED NUMBERS**

When n = 1: n= 20

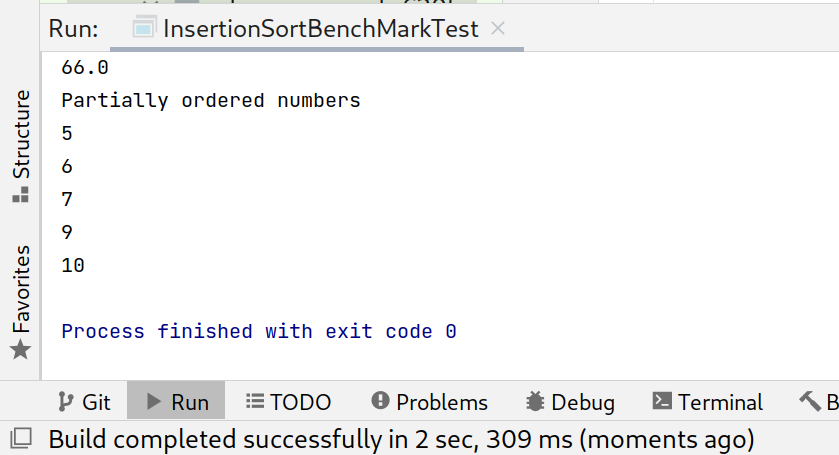
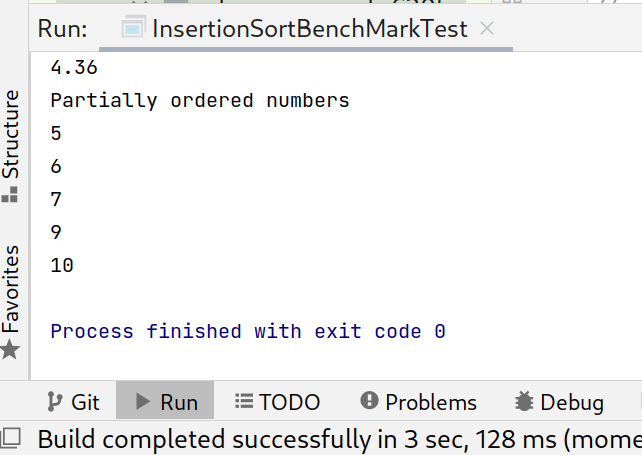


n = 40 n = 200

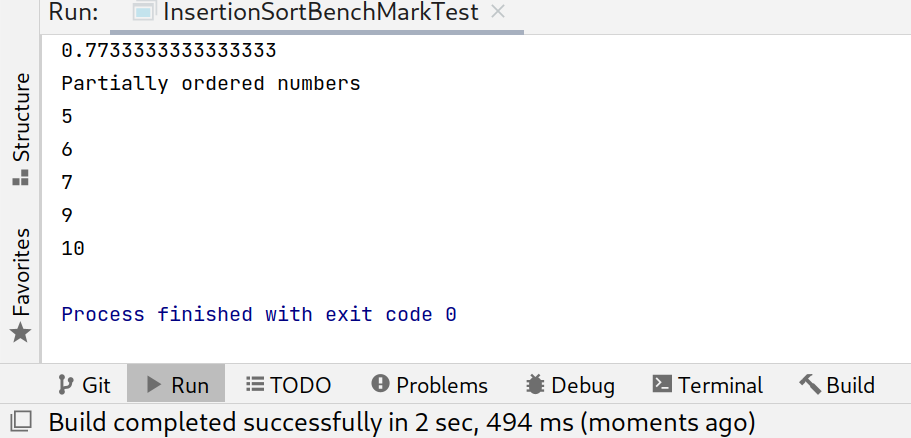
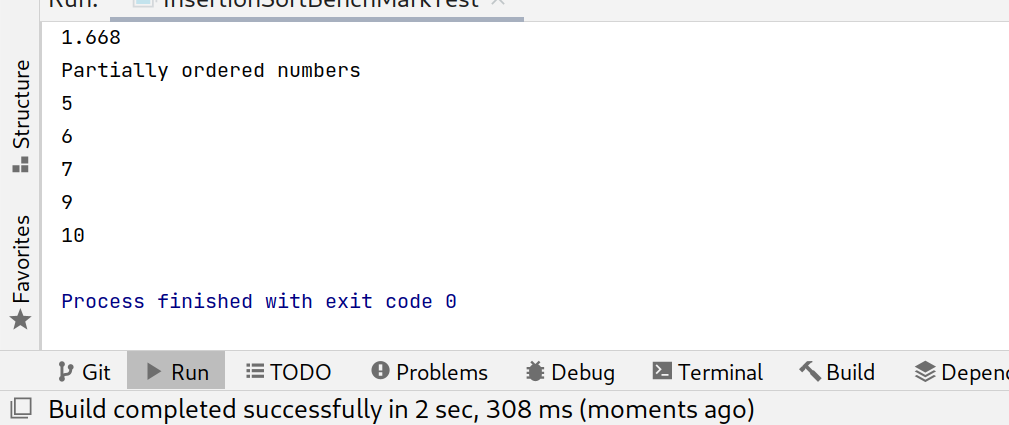


**Partial ordered NUMBERS**

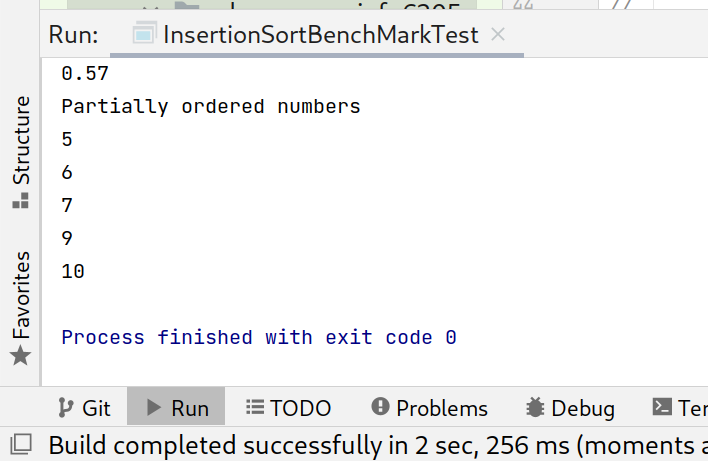
n =1 n = 50



n = 250 n = 750

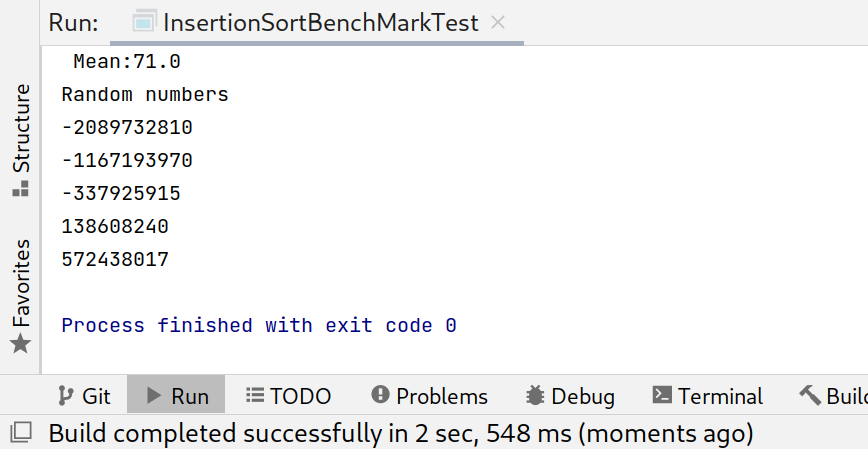
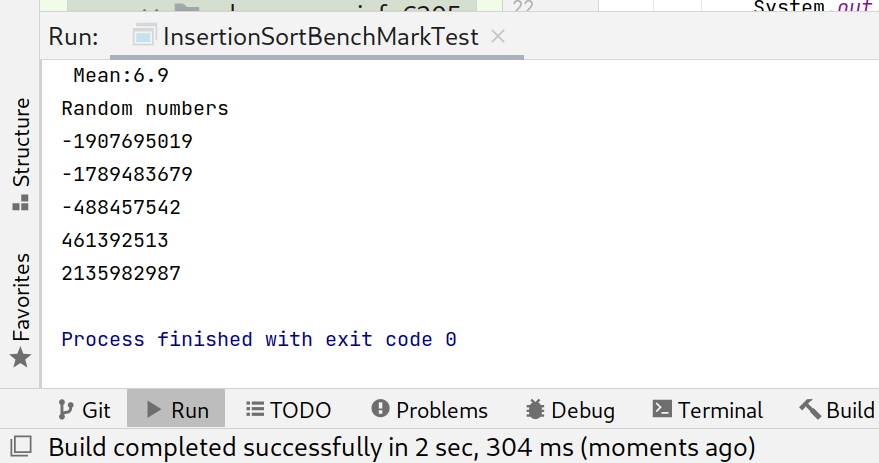


n= 1000

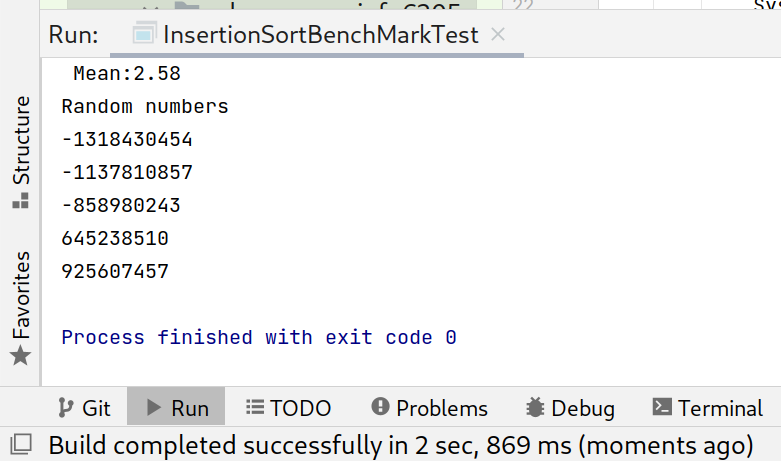
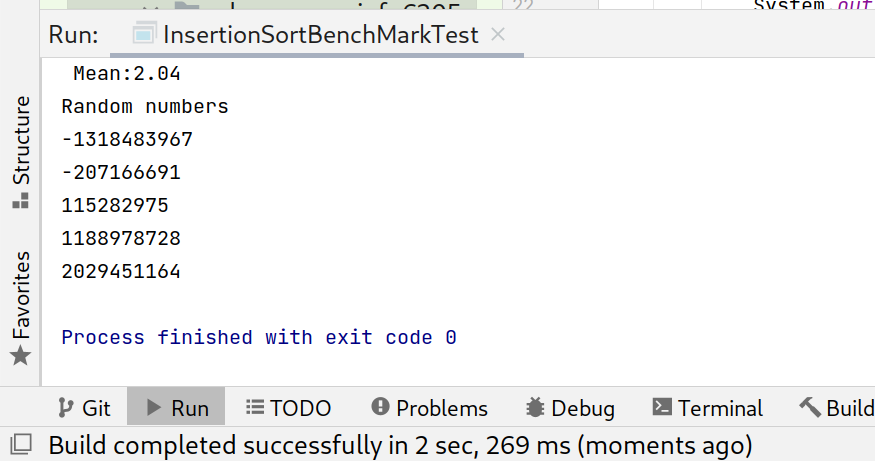


**Random Numbers**

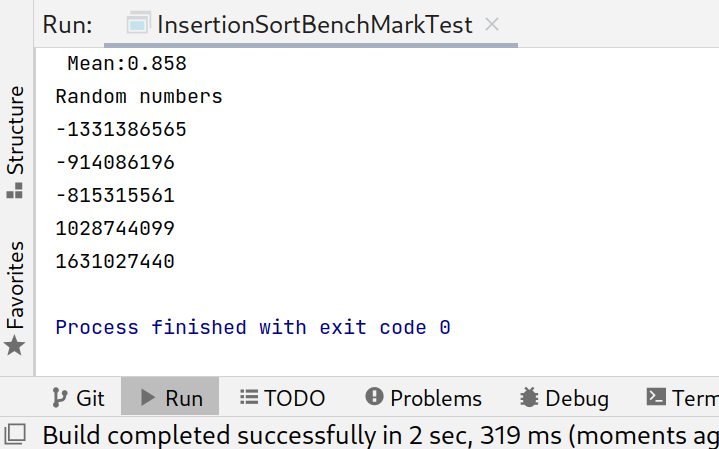
n = 1 n= 10



n= 50 n= 100

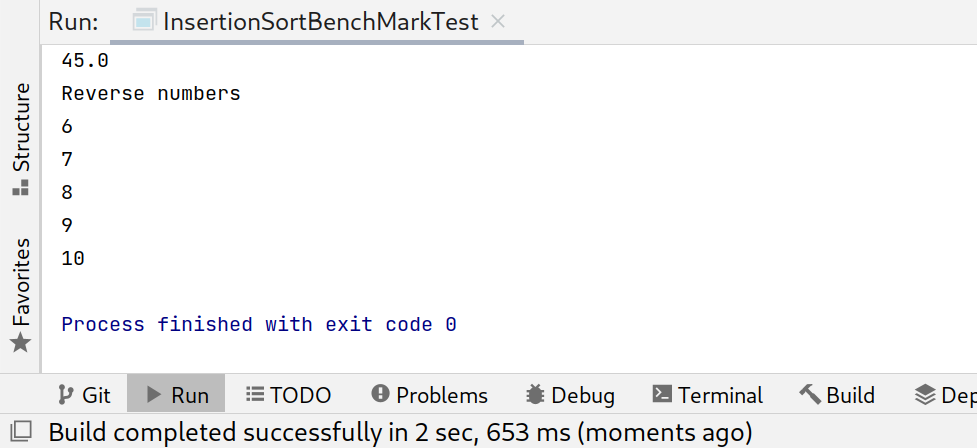
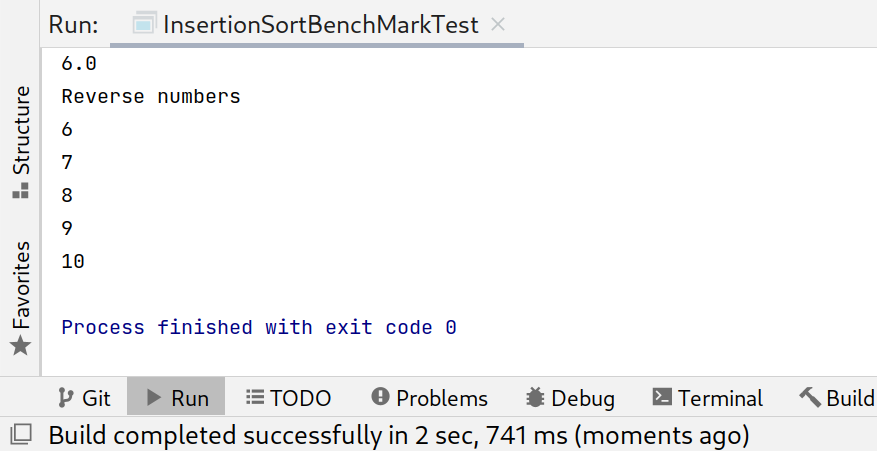


n = 500



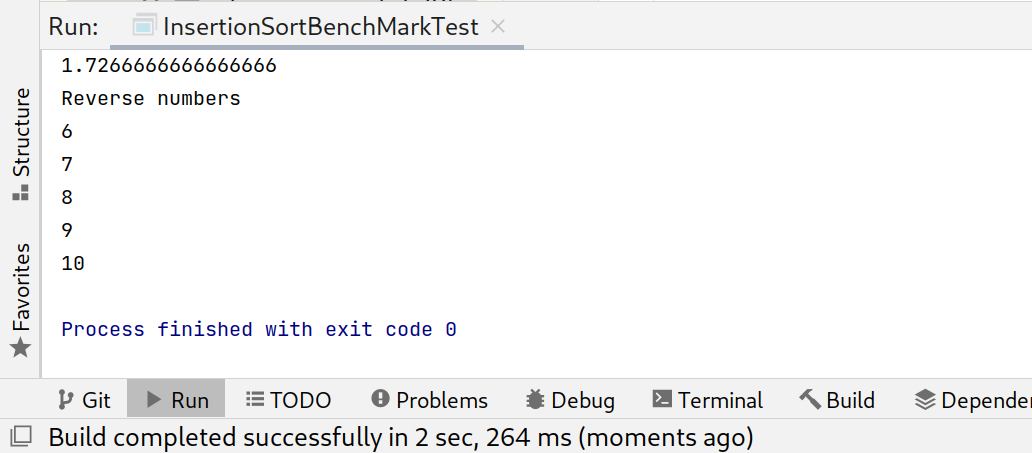
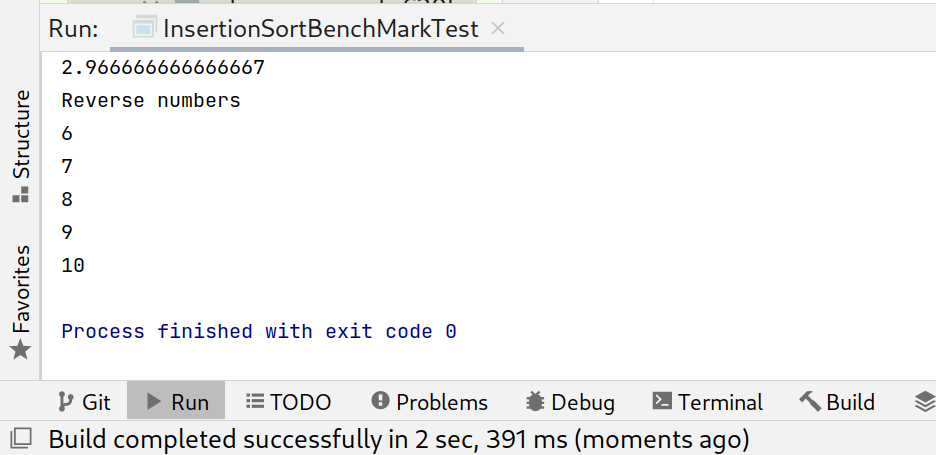
Reverse Ordered numbers

n =1 n= 15



n= 30z

n =150



n= 600

