Sribalaji Prabhakar (001517537)

Program Structures & Algorithms

Spring 2021

Assignment No. 2

Task:

Our task is to implement the repeat function, complete the insertion sort code, and then compare the insertion sorts benchmark for the different input arrays (random, ordered, partially ordered, and reverse-ordered).

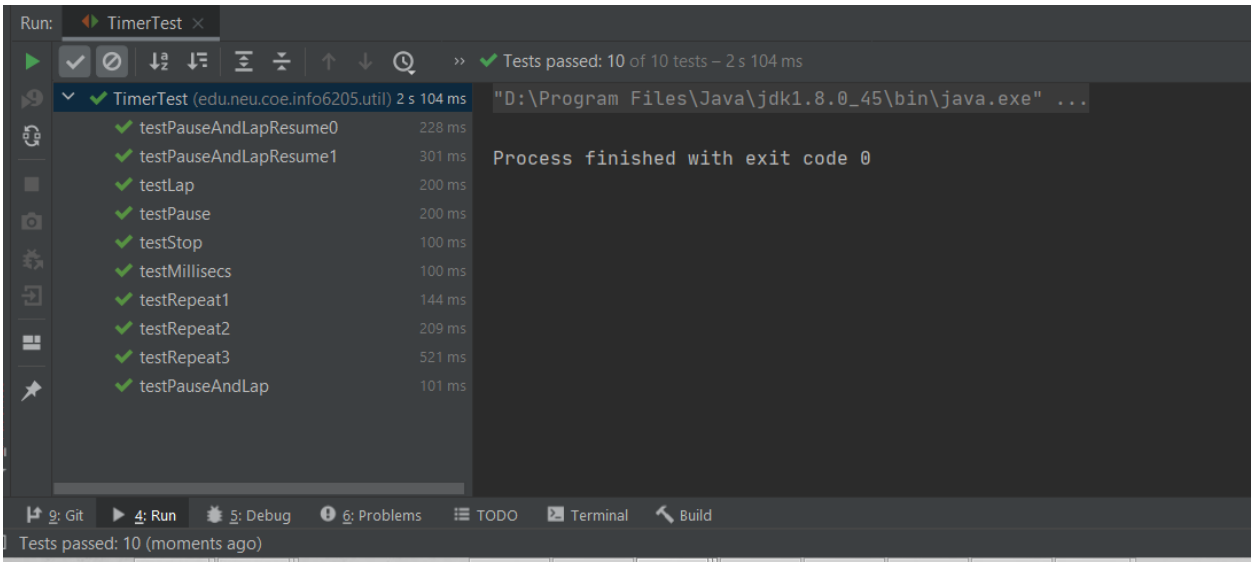
Part 1:

Added the following logic, and the test cases passed.

```
public <T, U> double repeat(int n, Supplier<T> supplier, Function<T, U> function, UnaryOperator<T> preFunction,
                            Consumer<U> postFunction) {
    logger.trace("repeat: with " + n + " runs");
    // TO BE IMPLEMENTED: note that the timer is running when this method is called and
    // should still be running when it returns.
    pause();
    T t = supplier.get();
    for(int i=1 ; i <= n ; i++) {
        if(preFunction!=null) {
            t = preFunction.apply(t);
        }
        resume();
        U fun = function.apply(t);
        lap();
        pause();
        if (postFunction != null)
            postFunction.accept(fun);
    }
    return meanLapTime();
}
```

Benchmark Test cases: Passed

TimerTest case: Passed



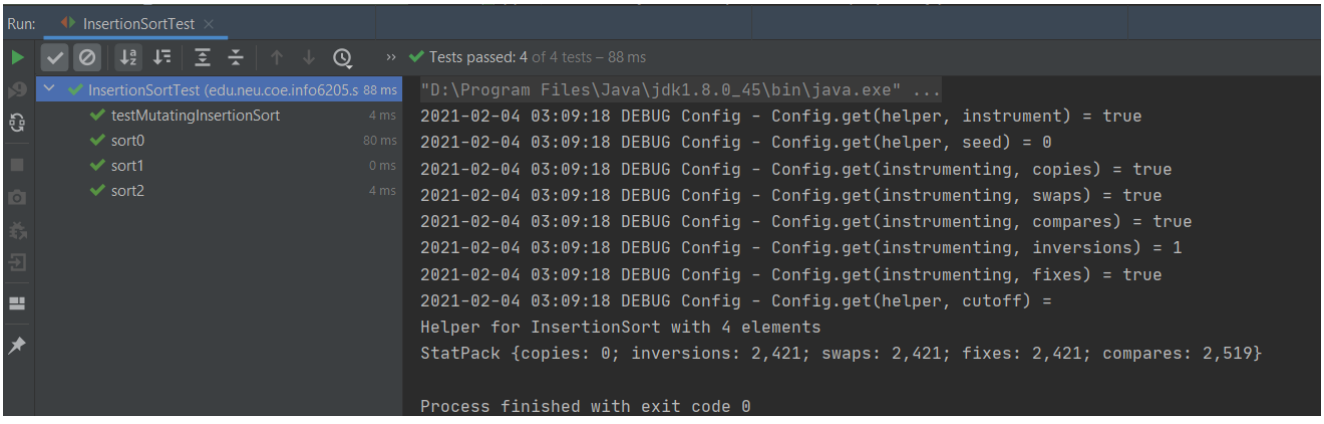
Part 2:

Used the helper method to implement insertion sort.

```
public void sort(X[] xs, int from, int to) {
    final Helper<X> helper = getHelper();

    // TO BE IMPLEMENTED
    for(int i = from; i < to ; i++) {
        //int key = xs[i].;
        int j = i;
        while (j >0 && helper.less(xs[j],xs[j-1])) {
            helper.swap(xs,  i: j-1, j);
            j--;
        }
    }
```

The following insertionSort test cases were passed:



Part 3:

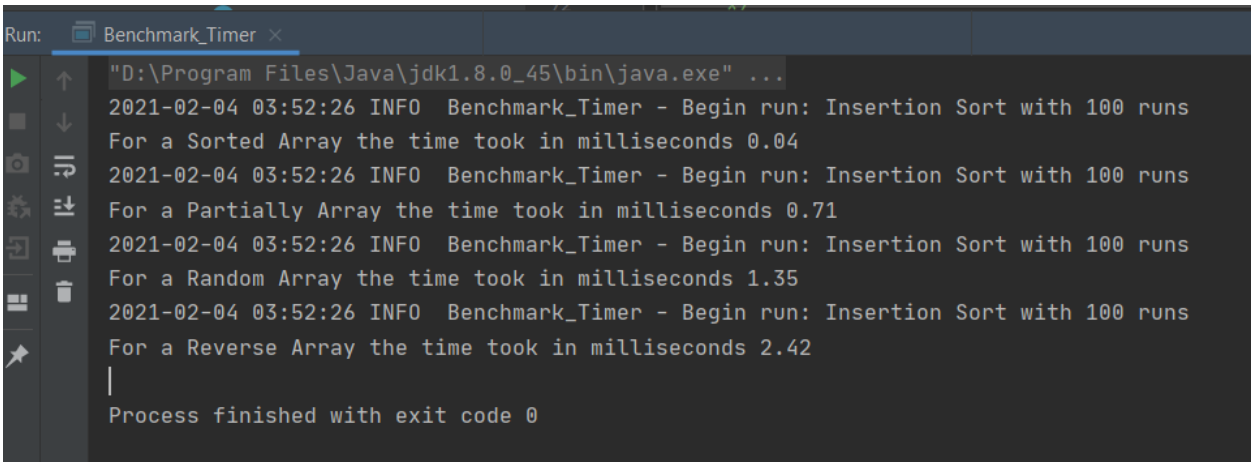
Implemented a main method in Benchmark.java and utilized the run method.

Four types of Input Arrays:

1. Sorted Array
2. Reverse Sorted
3. Partial Sorted
4. Random Integers

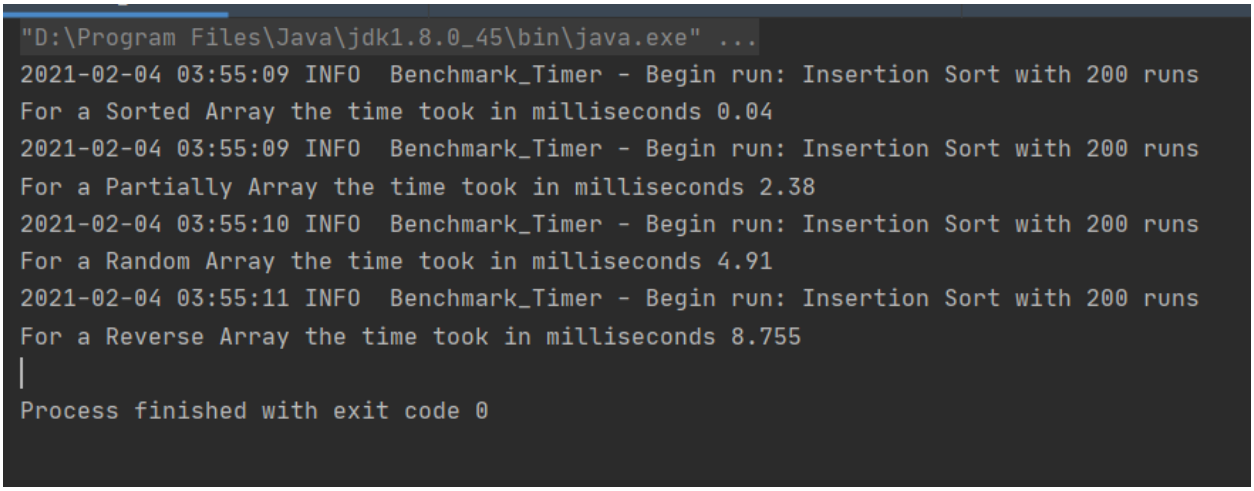
1st run with 100 times(m)

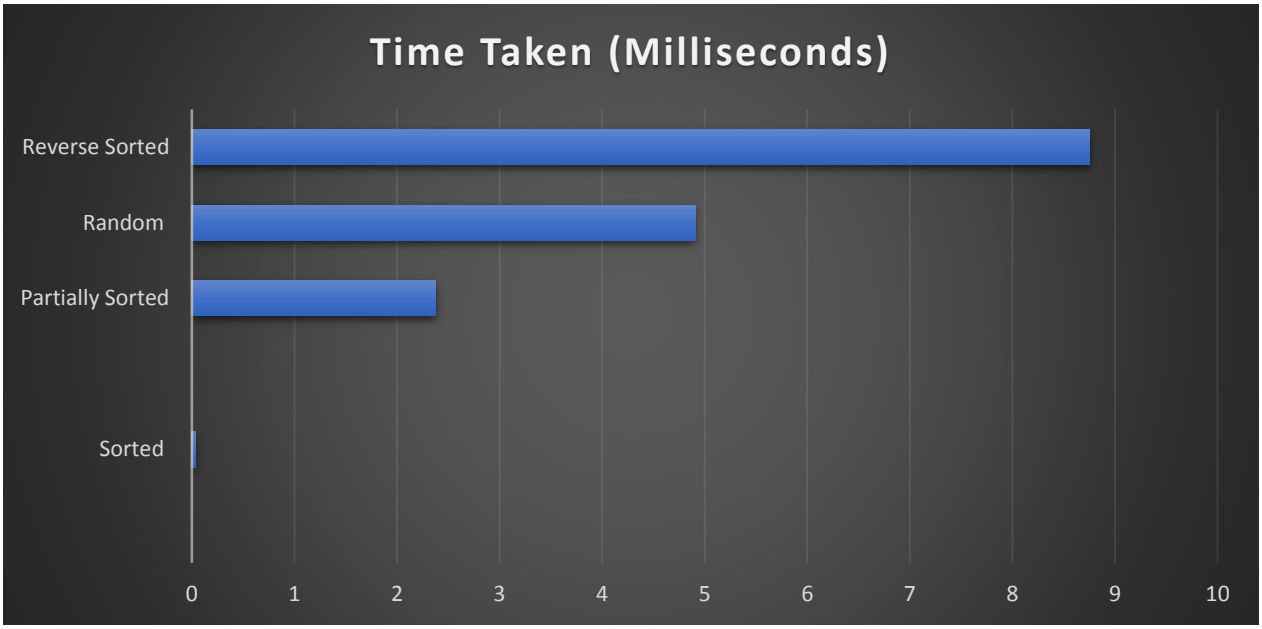
Type of Array	No of elements	Time Taken (Milliseconds)
Sorted	1000	0.04
Partially Sorted	1000	0.71
Random	1000	1.35
Reverse Sorted	1000	1.35



2nd Run with 200 times(m)

Type of Array	No of elements	Time Taken (Milliseconds)
Sorted	2000	0.04
Partially Sorted	2000	2.38
Random	2000	4.91
Reverse Sorted	2000	8.755





Conclusion:

Based on the above benchmark results and graph, we can say that the Insertion sort algorithm takes more time when all the elements in the arrays are in reverse sorted, and it takes linear time when the elements are already in a sorted fashion.

Time taken:
Reverse sorted > Random > Partially Sorted > Sorted array

Time complexity of insertion sort:

Best case: O(n)

Worst case: O(n^2)

Average case: O(n^2)