Noah Hayek (27080409)

OB1 - Algorithm Measurements

Algorithms	Computer	Execution time for the following values of n								Big-O
		1	10	100	1000	10000	100000	1000000	10000000	
a)	ThinkPad	0 ms	0 ms	0 ms	0 ms	1 ms	7 ms	1 ms	8 ms	**O(n)**
	MacBook	0 ms	0 ms	0 ms	0 ms	0 ms	2 ms	5 ms	11 ms	
b)	ThinkPad	0 ms	0 ms	1 ms	9 ms	73 ms	5187 ms	745714 ms	60418163 ms	**O(n^2)**
	MacBook	0 ms	0 ms	0 ms	5 ms	50 ms	5115 ms	481511 ms	47654454 ms	
c)	ThinkPad	0 ms	0 ms	0 ms	0 ms	0 ms	0 ms	0 ms	0 ms	**O(1)**
	MacBook	0 ms	0 ms	0 ms	0 ms	0 ms	0 ms	0 ms	0 ms	

Results and comments:

The above result demonstrates the time it takes for executing each algorithm in milliseconds. Two computers were used, a ThinkPad Yoga with an intel Core i7 processor and a MacBook late 2009 model with an intel Core 2 duo. While the results show that the MacBook finished earlier then the ThinkPad, they do not represent the efficiency of the code. Furthermore, this measurement is restricted by the hardware and software/operating system where the algorithm is executed. For example, algorithm b took longer to finish on the ThinkPad despite being equipped with a faster processor compared to the MacBook. This delay is due to the ThinkPad running multiple programs and applications during that execution. The Big-O notation is a better way of measuring efficiency as it does not rely on hardware or software. The results of Big-O in the table bellow show that the most efficient algorithm is c with O(1), followed by algorithm a with a O(n), and in the last place, algorithm b with O(n^2).

Calculating the Big-O for each algorithm:

Algorithms/Statements	Worst case number of executions.
a) sum ← 0	1
for m ← 1 to n	n-1
sum ← sum + m	1
Big-O	1+(n-1)+1 = n+1 ⇔ O(n+1) ⇔ **O(n)**
b) sum ←0	1
for m ← 1 to n	n-1
for k ← 1 to m	* [n-1
sum ← sum + 1	2]
Big-O	n(n+1)/2=(1/2)(n^2+n) ⇔ **O(n^2)**
c) sum ← n * (n + 1)/2	4
Big-O	O(4) ⇔ **O(1)**

```
Computer running: Windows 10

      Testing execution time when n = 1.

            The first algorithm took 0 ms to execute.
            The sum is equal to: 1

            The second algorithm took 0 ms to execute.
            The sum is equal to: 1

            The third algorithm took 0 ms to execute.
            The sum is equal to: 1

      Testing execution time when n = 10.

            The first algorithm took 0 ms to execute.
            The sum is equal to: 55

            The second algorithm took 0 ms to execute.
            The sum is equal to: 55

            The third algorithm took 0 ms to execute.
            The sum is equal to: 55

      Testing execution time when n = 100.

            The first algorithm took 0 ms to execute.
            The sum is equal to: 5050

            The second algorithm took 1 ms to execute.
            The sum is equal to: 5050

            The third algorithm took 0 ms to execute.
            The sum is equal to: 5050

      Testing execution time when n = 1000.

            The first algorithm took 0 ms to execute.
            The sum is equal to: 500500

            The second algorithm took 9 ms to execute.
            The sum is equal to: 500500

            The third algorithm took 0 ms to execute.
            The sum is equal to: 500500
```

```
Testing execution time when n = 10000.


      The first algorithm took 1 ms to execute.
      The sum is equal to: 50005000

      The second algorithm took 73 ms to execute.
      The sum is equal to: 50005000

      The third algorithm took 0 ms to execute.
      The sum is equal to: 50005000


Testing execution time when n = 100000.


      The first algorithm took 7 ms to execute.
      The sum is equal to: 5000050000

      The second algorithm took 5187 ms to execute.
      The sum is equal to: 5000050000

      The third algorithm took 0 ms to execute.
      The sum is equal to: 5000050000


Testing execution time when n = 1000000.


      The first algorithm took 1 ms to execute.
      The sum is equal to: 500000500000

      The second algorithm took 745714 ms to execute.
      The sum is equal to: 500000500000

      The third algorithm took 0 ms to execute.
      The sum is equal to: 500000500000


Testing execution time when n = 10000000.


      The first algorithm took 8 ms to execute.
      The sum is equal to: 50000005000000

      The second algorithm took 60418163 ms to execute.
      The sum is equal to: 50000005000000

      The third algorithm took 0 ms to execute.
      The sum is equal to: 50000005000000
```

```
Computer running: Mac OS X

      Testing execution time when n = 1.

            The first algorithm took 0 ms to execute.
            The sum is equal to: 1

            The second algorithm took 0 ms to execute.
            The sum is equal to: 1

            The third algorithm took 0 ms to execute.
            The sum is equal to: 1

      Testing execution time when n = 10.

            The first algorithm took 0 ms to execute.
            The sum is equal to: 55

            The second algorithm took 0 ms to execute.
            The sum is equal to: 55

            The third algorithm took 0 ms to execute.
            The sum is equal to: 55

      Testing execution time when n = 100.

            The first algorithm took 0 ms to execute.
            The sum is equal to: 5050

            The second algorithm took 0 ms to execute.
            The sum is equal to: 5050

            The third algorithm took 0 ms to execute.
            The sum is equal to: 5050

      Testing execution time when n = 1000.

            The first algorithm took 0 ms to execute.
            The sum is equal to: 500500

            The second algorithm took 5 ms to execute.
            The sum is equal to: 500500

            The third algorithm took 0 ms to execute.
            The sum is equal to: 500500
```

```
Testing execution time when n = 10000.


      The first algorithm took 0 ms to execute.
      The sum is equal to: 50005000

      The second algorithm took 50 ms to execute.
      The sum is equal to: 50005000

      The third algorithm took 0 ms to execute.
      The sum is equal to: 50005000


Testing execution time when n = 100000.


      The first algorithm took 2 ms to execute.
      The sum is equal to: 5000050000

      The second algorithm took 5115 ms to execute.
      The sum is equal to: 5000050000

      The third algorithm took 0 ms to execute.
      The sum is equal to: 5000050000


Testing execution time when n = 1000000.


      The first algorithm took 5 ms to execute.
      The sum is equal to: 500000500000

      The second algorithm took 481511 ms to execute.
      The sum is equal to: 500000500000

      The third algorithm took 0 ms to execute.
      The sum is equal to: 500000500000


Testing execution time when n = 10000000.


      The first algorithm took 11 ms to execute.
      The sum is equal to: 50000005000000

      The second algorithm took 47624454 ms to execute.
      The sum is equal to: 50000005000000

      The third algorithm took 0 ms to execute.
      The sum is equal to: 50000005000000
```