2018-2019 FALL SEMESTER

CS 201

HOMEWORK 2 – 12.12.2018

SECTION: 1

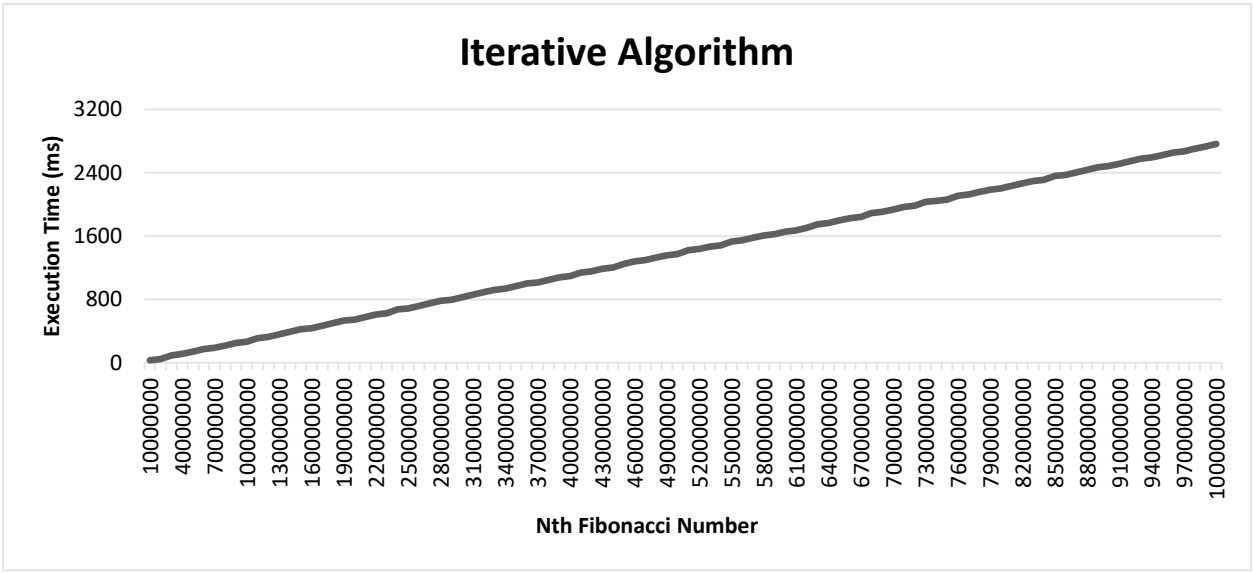
NAME: MUNIB EMRE

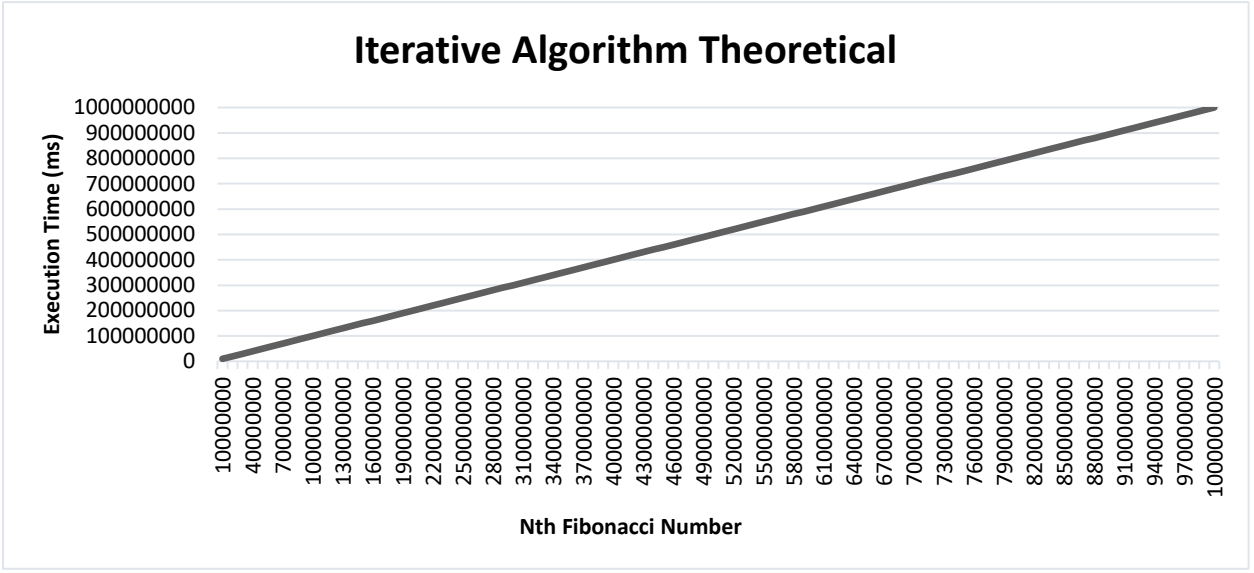
SURNAME: SEVILGEN

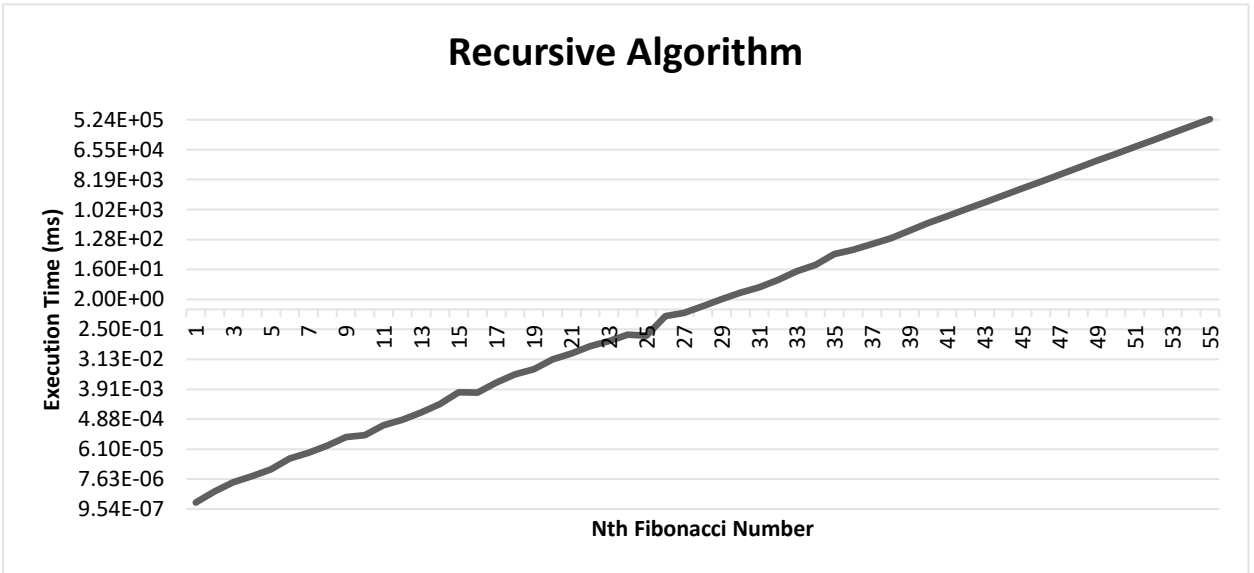
STUDENT ID: 21602416

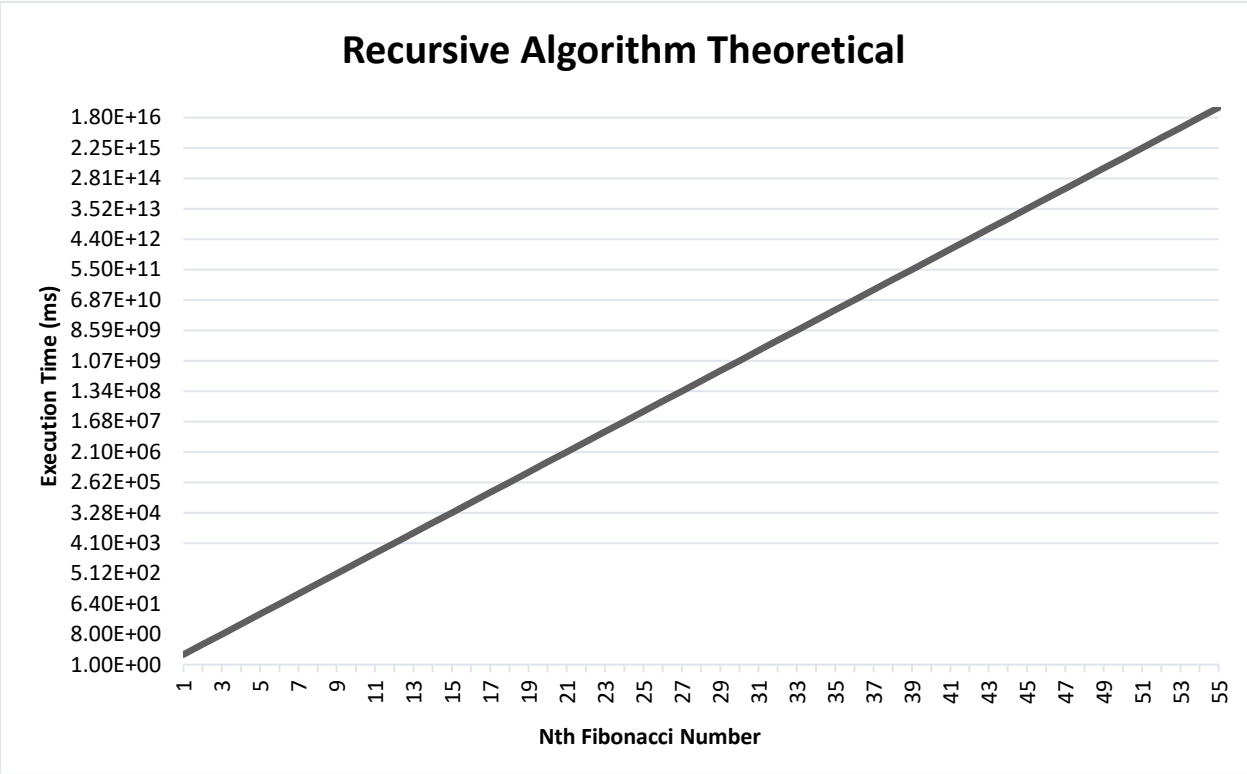
Data of the Iterative Algorithm							
Nth Fibonacci Number	Time (ms)	Nth Fibonacci Number	Time (ms)	Nth Fibonacci Number	Time (ms)	Nth Fibonacci Number	Time (ms)
10,000,000	31	260,000,000	719	510,000,000	1,422	760,000,000	2,109
20,000,000	47	270,000,000	749	520,000,000	1,437	770,000,000	2,124
30,000,000	93	280,000,000	782	530,000,000	1,468	780,000,000	2,156
40,000,000	110	290,000,000	796	540,000,000	1,484	790,000,000	2,187
50,000,000	140	300,000,000	828	550,000,000	1,531	800,000,000	2,202
60,000,000	172	310,000,000	859	560,000,000	1,547	810,000,000	2,234
70,000,000	188	320,000,000	891	570,000,000	1,578	820,000,000	2,265
80,000,000	218	330,000,000	921	580,000,000	1,609	830,000,000	2,296
90,000,000	250	340,000,000	938	590,000,000	1,624	840,000,000	2,312
100,000,000	266	350,000,000	968	600,000,000	1,656	850,000,000	2,359
110,000,000	312	360,000,000	1,000	610,000,000	1,671	860,000,000	2,374
120,000,000	328	370,000,000	1,015	620,000,000	1,703	870,000,000	2,406
130,000,000	359	380,000,000	1,047	630,000,000	1,749	880,000,000	2,437
140,000,000	391	390,000,000	1,078	640,000,000	1,765	890,000,000	2,468
150,000,000	422	400,000,000	1,093	650,000,000	1,797	900,000,000	2,484
160,000,000	437	410,000,000	1,140	660,000,000	1,827	910,000,000	2,515
170,000,000	469	420,000,000	1,156	670,000,000	1,844	920,000,000	2,546
180,000,000	499	430,000,000	1,187	680,000,000	1,890	930,000,000	2,578
190,000,000	532	440,000,000	1,203	690,000,000	1,906	940,000,000	2,593
200,000,000	546	450,000,000	1,249	700,000,000	1,937	950,000,000	2,624
210,000,000	578	460,000,000	1,281	710,000,000	1,968	960,000,000	2,656
220,000,000	609	470,000,000	1,297	720,000,000	1,984	970,000,000	2,671
230,000,000	625	480,000,000	1,328	730,000,000	2,031	980,000,000	2,702
240,000,000	672	490,000,000	1,359	740,000,000	2,046	990,000,000	2,734
250,000,000	687	500,000,000	1,374	750,000,000	2,062	1,000,000,000	2,765

Data of the Recursive Algorithm							
Nth Fibonacci Number	Time (ms)	Nth Fibonacci Number	Time (ms)	Nth Fibonacci Number	Time (ms)	Nth Fibonacci Number	Time (ms)
1	0.0000015	15	0.0032	29	2.03	43	1,687
2	0.0000032	16	0.0031	30	3.2	44	2,734
3	0.0000062	17	0.0062	31	4.6	45	4,421
4	0.0000094	18	0.011	32	7.8	46	7,155
5	0.000015	19	0.0156	33	14.1	47	11,606
6	0.000032	20	0.031	34	21.8	48	18,762
7	0.000047	21	0.047	35	47	49	30,383
8	0.000078	22	0.078	36	63	50	49,363
9	0.00014	23	0.11	37	94	51	79,466
10	0.00016	24	0.171	38	140	52	128,438
11	0.00032	25	0.16	39	234	53	207,879
12	0.00047	26	0.62	40	406	54	336,077
13	0.00078	27	0.78	41	641	55	543,962
14	0.0014	28	1.25	42	1,047		


Iterative Algorithm
Execution Time (ms)
3200
2400
1600
800
0
10000000
40000000
70000000
100000000
130000000
160000000
190000000
220000000
250000000
280000000
310000000
340000000
370000000
400000000
430000000
460000000
490000000
520000000
550000000
580000000
610000000
640000000
670000000
700000000
730000000
760000000
790000000
820000000
850000000
880000000
910000000
940000000
970000000
1000000000
Nth Fibonacci Number


Iterative Algorithm Theoretical
Execution Time (ms)
1000000000
900000000
800000000
700000000
600000000
500000000
400000000
300000000
200000000
100000000
0
10000000
40000000
70000000
100000000
130000000
160000000
190000000
220000000
250000000
280000000
310000000
340000000
370000000
400000000
430000000
460000000
490000000
520000000
550000000
580000000
610000000
640000000
670000000
700000000
730000000
760000000
790000000
820000000
850000000
880000000
910000000
940000000
970000000
1000000000
Nth Fibonacci Number


Recursive Algorithm
Execution Time (ms)
5.24E+05
6.55E+04
8.19E+03
1.02E+03
1.28E+02
1.60E+01
2.00E+00
2.50E-01
3.13E-02
3.91E-03
4.88E-04
6.10E-05
7.63E-06
9.54E-07
1
3
5
7
9
11
13
15
17
19
21
23
25
27
29
31
33
35
37
39
41
43
45
47
49
51
53
55
Nth Fibonacci Number



Specifications of the computer that is used to get the execution times:

- Processor: Intel® Core™ i7-7500U (2.70 - 2.90 GHz)
- RAM: 16 GB DDR3
- Operating System: Microsoft Windows 10 Home 64 Bit

The behavior of the theoretical graphs and the graphs formed by the data are almost the same. The iterative algorithm gives the linear graph as expected from the theoretical analysis of the algorithm that is given as $O(n)$. By looking the graph formed by the data, we can clearly say that the time complexity of the iterative algorithm is $O(n)$ because it is linear. Moreover, in the graphic of the recursive algorithm, y-axis is formed by logarithmic scale. Therefore, the recursive algorithm gives the exponential growth rate as expected. It is given that the theoretical analysis of the recursive algorithm gives the time complexity as $O(2^n)$. We can easily say that the time complexity of the recursive algorithm is $O(2^n)$ by considering both the graph formed by data and the theoretical graph. The growth rate of the graph of data is slightly less than the growth rate of the theoretical graph. Therefore, we can say that the time complexity of the recursion algorithm is $O(2^n)$, because the Big O notation says that the time complexity of the recursive algorithm is less than or equal to 2^n.