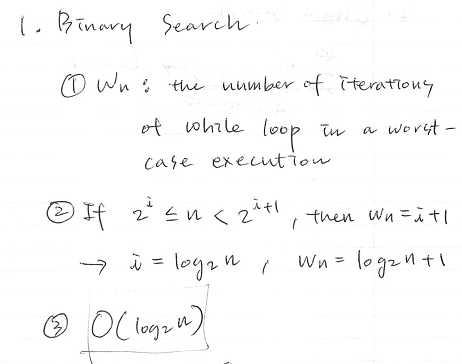
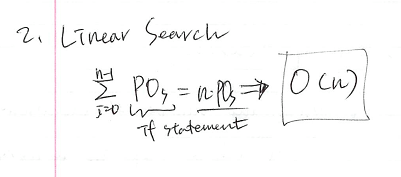
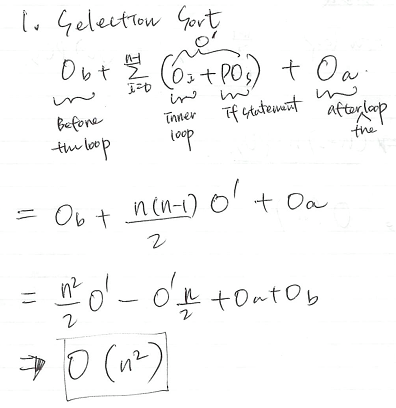
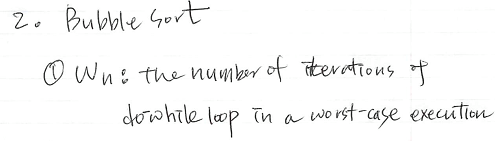
**Problem 1)**

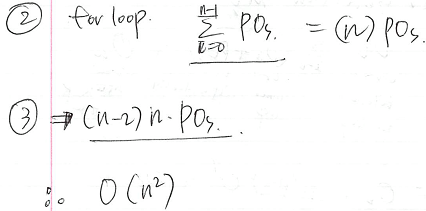




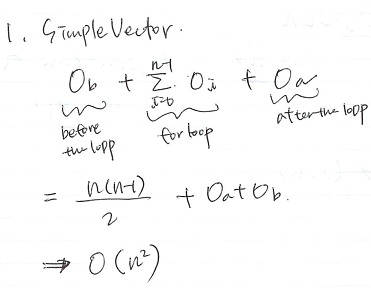
**Problem2)**

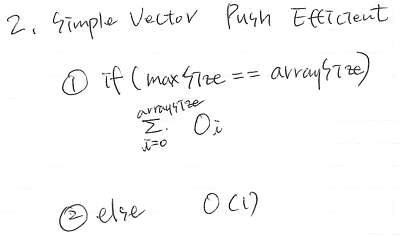




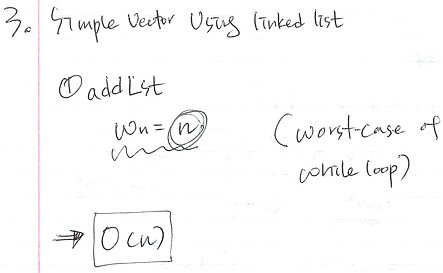


**Problem3)**

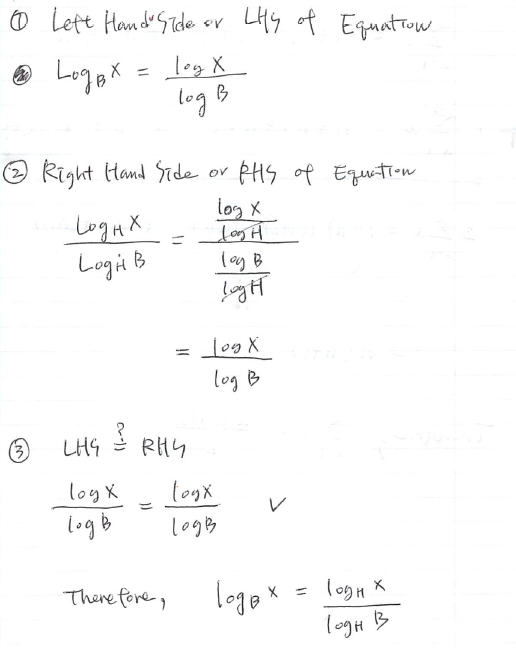




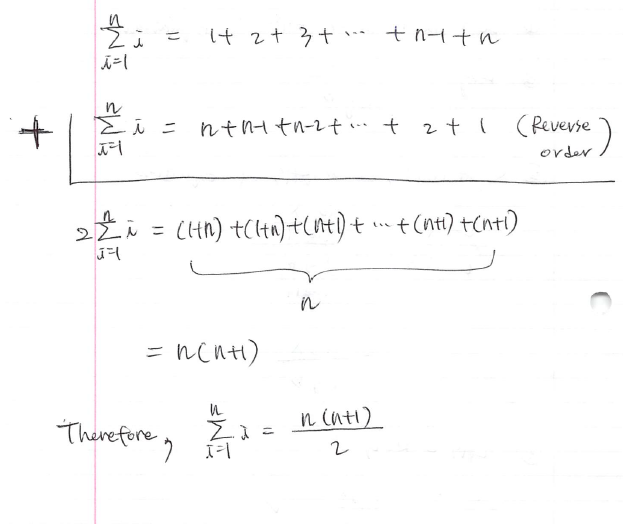
In the beginning, since the array size is relatively small it doesn’t affect to the efficiency and as the array size becomes larger, the case to create additional memory space, which uses for-loop, rarely happens. In this reason, its graph is y=c where c is constant in the simulation.



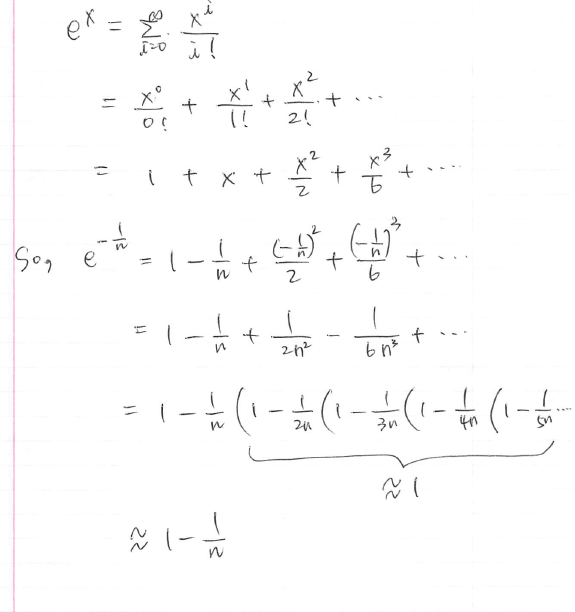
**Problem4)**



**Problem5)**



Problem6)



**Problem7)**

In the simulation,

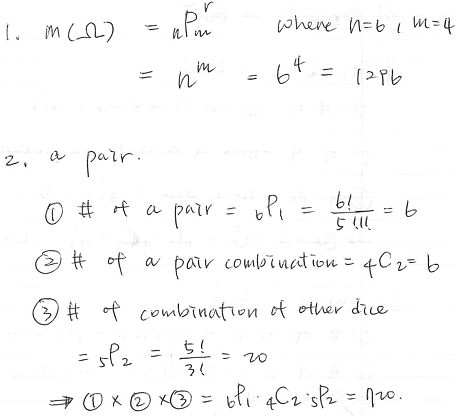
The number of calling fiboRec(0) + The number of calling fiboRec(1)

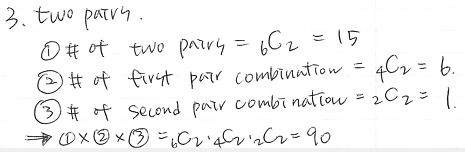
= fiboRec(n+1)

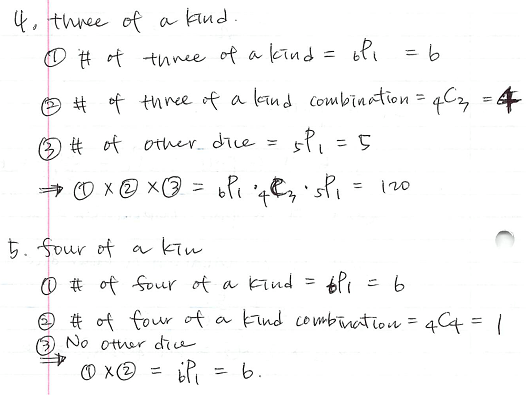
So, it takes Fibonacci time when we execute the recursive Fibonacci.

**Problem8)**

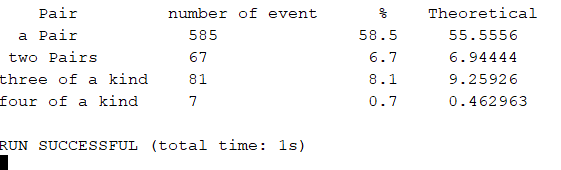
1. Calculation:







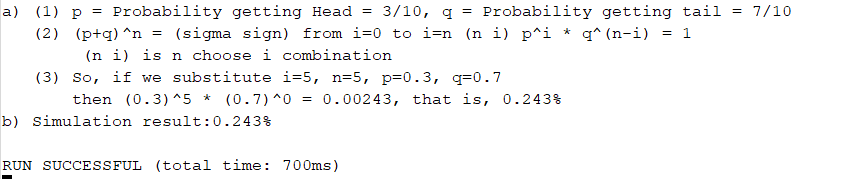
1. Simulation:



1. Sumarry

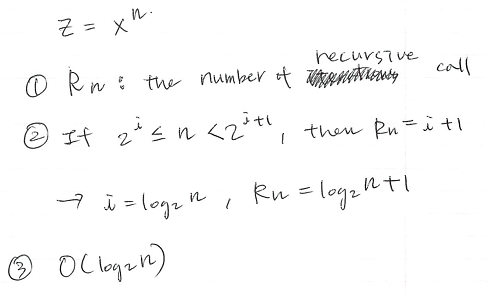
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Theoretical | | Simulation | | Diff |
| # | Probability | # | Probability |
| a pair | 720 | 56% | 585 | 59% | -3% |
| two pair | 90 | 7% | 67 | 7% | 0% |
| three of a kind | 120 | 9% | 81 | 8% | 1% |
| four of a kind | 6 | 0% | 7 | 1% | 0% |

**Problem9)**

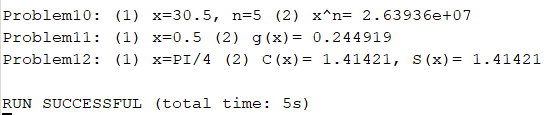


**Problem10, 11, 12)**

1. Calculation (Problem10):



1. Simulation Result(Problme10, 11, 12)



**Problem13)**

When arySize=50 and modNum=12, simulation result:

