CS251 HW1

1.
1.1 C
For the first loop, it runs n times. Inside of every first loop, it will run n^2 times. Therefore, in total, there will be around n*n^2 = n^3 times. Based on the definition of O: f(n) is O(g(n)) if f(n) is asymptotically less than or equal to g(n). O(n^3) is the answer.

1.2 C
For the first loop, it runs for n times. Inside of every first loop, it will run for i times. Therefore, the total number is: 1 + 2 + 3 + … + n. There will be $\frac{1}{2}n(1+n) = \frac{1}{2}n^2 + \frac{1}{2}n$ times. Based on the definition of O: f(n) is O(g(n)) if f(n) is asymptotically less than or equal to g(n). O(n^2) is the answer.

1.3 B
For the first loop, it runs for around $\log_2 n$ times. Inside of every first loop, it will run for n times. Therefore, the total number is: $nlog_2n$. There will be $\frac{1}{2}n(1+n) = \frac{1}{2}n^2 + \frac{1}{2}n$ times. Based on the definition of O: f(n) is O(g(n)) if f(n) is asymptotically less than or equal to g(n). $\mathrm{O(nlogn)}$ is the answer.

1.4 A
For every call of this recursive function, it will call one more function (when n is not equal to 1). Therefore, the total number of repetition is n. Based on the definition of O: f(n) is O(g(n)) if f(n) is asymptotically less than or equal to g(n). O(n) is the answer.

1.5 B
For loop 1, the first loop runs for n times. Inside of the first loop, it will run for $\log_2 n$ times. For loop 2, it will run for n times. In total, there will be $nlog_2n + n$ times. Based on the definition of O: f(n) is O(g(n)) if f(n) is asymptotically less than or equal to g(n). $\mathrm{O(nlogn)}$ is the answer.

2.
2.1 B
Since n! > 3^n, based on the definition of O: f(n) is O(g(n)) if f(n) is asymptotically less than or equal to g(n). $\mathrm{O(n!)}$ is the answer.

2.2 A
$\sum_{i=0}^{2n} 5i + i^2 = \frac{2n(n+1)(2n+1)}{6} + P(n^2) \sim n^3$. Therefore, based on the definition of Ω: f(n) is Ω(g(n)) if f(n) is asymptotically greater than or equal to g(n). $\Omega(\mathrm{n}^3)$ is the answer.

3.
n should be the number of worst case. Since the cards are unsorted, if we search from one end, where the search target is in the other end, we need n times to find the object.

4.

$$2^N > N^3 > N^2logN > N^2 > Nlog^2N > 10NlogN^2 \geq NlogN > N > \log^2 N > 5logN \geq logN^2 \geq logN > 37 = 2$$

Since the function growth rate is not affected by constant factors or lower-order terms, we can eliminate the constant scalars in sorting this list. For $10NlogN^2 = 20NlogN \sim NlogN$, it is the same growth rate as $NlogN$. Also for $5logN \sim logN, logN^2 = 2logN \sim logN$, they have the same growth rate.

5.

a.

AboveAvg1 is better.

In AboveAvg1, the first loop will run for n times. The second loop will also run for n times. Therefore, the complexity is O(n).

In AboveAvg2, the outside loop will run for n times. For every time in the outside loop, a inside loop will run for n times. Therefore, the complexity is O(n^2).

In conclusion, AboveAvg1 is better since O(n) < O(n^2).

b.

EvensFirst2 is better.

For EvensFirst1, the first loop and the second loop will all run for n times. There will be 2n times together.

For EvensFirst2, the loop will run for n times.

Therefore, EvensFirst2 is better.