Review QUIZ 1 of 2

Algorithms: Efficiency and Analysis

(All questions are based on Lecture Notes. Go to the Lecture Notes and fully understand the topics!)

Q1. Two types of algorithms? How to represent Time complexity of an algorithm? Four time complexity cases? **Asymptotic time complexity**? Why not exact time complexity?

[GO TO Lecture Note Slide #15, #30, #32, #37, #39]

Q2. Notation for asymptotic time complexity **upper bound**? Define f(n) = **O** (g(n))

[GO TO Lecture Note Slide #42, #43]

Q3. Describe an **iterative Binary Search** algorithm? Calculate Big-O?
[GO TO Lecture Note Slide #78

Q4. Notation for asymptotic time complexity **lower bound**? Define f(n) = **Ω** (g(n))
[GO TO Lecture Note Slide #55, #56]

Review QUIZ 2 of 2

Algorithms: Efficiency and Analysis

(All questions are based on Lecture Notes. Go to the Lecture Notes and fully understand the topics!)

Q5. Calculate Big-O using **summation**? **[GO TO Lecture Note Slide #75, #76]**

```
sum1 = 0;
for (i=1; i<=n; i*=2)
   for (j=1; j<=n; j++)
      sum1 ++;


sum2 = 0;
for (i=1; i<=n; i*=2)
   for (j=1; j<=i; j++)
      sum2 ++;
```

Q6. Calculate Big-O using **substitution**? **[GO TO Lecture Note Slide #84, #86, #89]**

T(n) = T(n-1) + n for n>0

T(0) = 1

T(n) = T(n/2) + 1 for n>1

T(1) = 1

Q7. Describe a **recursive Binary Search** algorithm? Worst-case Recurrence Equation? Calculate Big-O?
[GO TO Lecture Note Slide #95]

Q8. Amortized time complexity? Why? **[GO TO Lecture Note Slide #104, #105]**