1. **Time Complexity:** For each question, write down the best asymptotic (big-O) characterization of the following functions:
    - **a.** $f(n) = 10^7 n^{0.6} + \frac{n^3}{log(n^5)} + nlog(n^4) + 0.1n^3$
    - **b.** $Let\ f(n) = 10n^3$ and $g(n) = 5n^3 + 21n$
        - i. $f(n) + g(n)$
        - ii. $f(n) - g(n)$
        - iii. $f(n) .\ g(n)$
        - iv. $\frac{g(n)}{f(n)}$
        - v. $f^2(n)$

SOLUTION:

1a) $O(n^3)$

1b)

- I. $f(n) + g(n) = 15n^3 + 21n => O(n^3)$
- II. $f(n) - g(n) = 5n^3 - 21n => O(n^3)$
- III. $f(n) .\ g(n) = 50n^6 + 210n^4 => O(n^6)$
- IV. $\frac{g(n)}{f(n)} = \frac{(5n^3+21n)}{10n^3} => \frac{1}{2} + \frac{2.1}{n^2} => O(1)$
- V. $f^2(n) = 100n^6 => O(n^6)$

2. **Search Algorithms: Sequential (Linear) Search and Binary Search**
    - **a.** What is the main difference between these two search algorithms in terms of input list type?
    - **b.** What are the best and worst case time complexities of each algorithm? And, when does the best case occur in binary search algorithm?
    - **c.** Imagine that you run Sequential Search algorithm on the input list “13, 5, 148, -230, 645, 135, 1099, 1980, 127, 16”. In which step you find the number 1980?
    - **d.** Imagine that you run Binary Search algorithm on the input list “-123, 17, 66, 97, 123, 125, 1071, 1453”. In which step you find the number 1071? Please show the each step.

SOLUTION:

a. Sorted List => Binary Search,

Input list does not have to be sorted for sequential search

b. Sequential Search

- i. BEST CASE: Searched item is the first element (O(1))
- ii. WORST CASE: Searched item is the last element (O(N))

Binary search:

- i. BEST CASE: Searched item is the medium element (O(1))
- ii. WORST CASE: Searched item is the first or last element (O(logn))

c. ???

d. ????

3. Programming Skills

What do you expect at the output when you execute the following C code?

```
void main() {
    int i, a[10] = { 10, 55, 9, 423,
2345, 20, 30, 40, -22, 34 };
    for(i=8;i>0;i--)
           printf("%d\n",a[i]);
    system ("pause");
    return 0;
}
```

In terms of n, what is the highest and lowest values of **count** that may be printed at the last line?

```
void myFunc(int n) {
    int i, count=0;
    for(i=1;i<=n;i++){
        r=1+rand()%n;
        if (i%r >4)
            count++
    }
    printf("count=%d",count);
}
```