Design and Analysis of Algorithms – Assignment 1

Q1. If $f(n) = 3n^2 + n^3 \lg n$, then f(n) is
a. O(n2)
b. $O(n^{3/2})$
c. $O(n^3 \lg n)$
d. $O(n^2/3)$
Answer: C

Reason : Highest power of n is n^3 lg n. Hence answer C.

Q2. What is the asymptotic relationship between the functions: x^p and k^x?
(Assuming that $p \geq 1$ and $k > 1$ are constants:)
a. x^p is $O(k^x)$
b. k^x is $O(x^p)$
c. x is O(k)
d. Both b and c
Answer: A

Reason : x^p is polynomial function and k^x is exponential function. Exponential functions grow faster than polynomial. Hence answer A.

Q3. For functions, nk and cn , what is the asymptotic relationship between these functions? Assume that $k \geq 1$, and $c \geq 1$ are constant.
a. n^k is O(cn)
b. n^k is Ω(cn)
c. nk is Θ(cn)
d. None of the above
Answer: This question was stated incorrectly. So everyone gets a bonus point for this one.

Q4. If $f(n) = 5 \lg n + 2 \lg n! + (n^2 + 1) \lg n$, what is the big-O notation for f(n)?
a. n
b. n^2
c. n lg n
d. n^2 lg n
Answer: D

Reason : The equation with highest power of n is $(n^2 + 1)\lg n$. Hence it will grow with n^2 lg n.

Q5. What is the time complexity for the following piece of code?

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

a. O(n2)
b. O(n)
c. O(lg n)
d. O(n lg n)
Answer: D

Reason : The outer loop will run with time complexity of n and inner loop will run with time complexity of lg n. Hence, O(n lg n).

Q6. If $f(x) = (x^3 - 1) / (5x + 1)$ then f(x) is
a. $O(x^2)$
b. $O(x)$
c. $O(x^3/5)$
d. $O(1)$
Answer: A

Reason : The highest power of x will be x^2.

Q7. The Big-O complexity of 1 + 2 + 3 + 4 ... + n is? (Assume we must add the numbers one at a time, rather than using Gauss's trick to get a closed form for the sum.)
a. $O(n)$
b. $O(n^2)$
c. $O(3n)$
d. $O(n^3)$
Answer: A

Reason:Here, the function will grow in linear manner. Hence O(n).

Q8. The Big-O complexity of 1 + 2 + 3 + 4 + ... + 100 is? (Assume we must add the numbers one at a time, rather than using Gauss's trick to get a closed form for the sum.)

a. O(1)
b. O(n)
c. O(n2)
d. O(3n)
Answer: A

Since we know the total numbers to be added, the function growth is going to remain constant. Hence, O(1).

Q9. What is big O for following code?

```
void complex(int n)
{
    int i, j;
    for(i = 1; i < n; i++) {
        for(j = 1; j < log(i); j++)
    }
    printf("Algorithms");
}
```

Answer: O(n lg n), O(lg(n!))
Reason : As we can see the inner loop will run max logn times and the outer loop runs for n times so n times logn operations would be executed so complexity is nlogn.

Q10. What is big O for following code?

```
void complex(int n)
{
    int i;
    for(i = n; i > 0; i = i/2){
        printf("Algorithms")
    }
}
```

Answer: O(logn)
Reason :As we can see after each iteration of the loop the value of i divides by 2. As we know that Time Complexity of a loop is considered as O(Logn) if the loop variables is divided / multiplied by a constant amount.