**Student ID: 1512002**

**Name: Le Duong Tuan Anh**

**Chapter 1 – Homework – Week 01**

**Problem 1:**

Let f(n) = . **f(n) = O(n5).**

Select that c = 5; n0 = 2; g(n) = n5

=> f(n) ≤ c.g(n) n ≥ n0 (\*)

***Prove (\*):***

***We have: c.g(n) – f(n) =***

So, f(n) = O(n5)

**Problem 2:**

**The statement “The running time of algorithm A is at least O(n2)” is meaningless, because of phrase “at least”.**

The algorithm A with time complexity O(n­5) is more complex than algorithm B with O(n2), but both will have running time at least O(n2). For example, f(n) = in problem 1 above, we proved that f(n) = O(n5). However, we say that “algorithm running time with f(n) is at least O(n2)” is not wrong. But it’s meaningless because O(n2) is very different with O(n5).

**Problem 3:**

1. f(x) = 2x2 – x + 30 is O(x2)

Select: c = 10; x0 = 3; g(x) = x2

=> f(x) ≤ c.g(x) x ≥ x0 (\*)

***Prove (\*):***

***We have: c.g(x) – f(x) = 8x2 + x – 30*** ≥ ***45***

So, f(x) = O(x2)

1. f(x) = (3x + 2)log2(x2 + 5) is O(xlog2x)

Select: c = 13; x0 = 1; g(x) = xlog2x

=> f(x) ≤ c.g(x) x ≥ x0 (\*)

***Prove (\*):***

***We have:***

***c.g(x) – f(x) = 13xlog­2x– 3xlog2(x2 + 5) – 2log2(x2 + 5)***

***= 13xlog2x – 3xlog2(x2 + 5) – 2log2(x2 + 5)***

***= 3x(3log2x – log2(x2 + 5)) + 2(2xlog2x – log2(x2 + 5))***

***=***

So, f(x) = O(xlog­2x)

1. f(x) = (x2 + 4log2x)/(x+1) is O(x)

Select: c = 4; x0 = 1; g(x) = x

=> f(x) ≤ c.g(x) x ≥ x0 (\*)

***Prove (\*)***

***We have:***

***c.g(x) – f(x) = =***

So, f(x) = O(x)

**Problem 4:**

1. **f(x) = 10 is not O(x)**

Select: c = 10; x0 = 0; g(x) = 1

=> f(x) ≤ c.g(x) x ≥ x0

(10 ≤ 10\*1 x ≥ 0)

=> f(x) is O(1)

**O(1) is independent with x (O(1) is constant), so f(x) is not O(x)**

1. **f(x) = 3x + 7 is O(x)**

With c = 4; x0 = 10, g(x) = x

We have: f(x) ≤ c.g(x) x ≥ x0 (\*)

***Prove (\*):***

***f(x) – c.g(x) = x – 7 ≥ 0 x ≥ 10***

So, f(x) is O(x)

1. **f(x) = 2x2 + 2 is not O(x)**

Assume that f(x) is O(x), there will exist c and x0 that, with g(x) = x, we have:

f(x) ≤ c.g(x) x ≥ x0.

<=> 2x2 + 2 ≤ c.x x ≥ x0.

<=> 2x + 2/x ≤ c x ≥ x0.

Because c is constant, x is variable so

=> f(x) is not O(x)

**Problem 5:**

1. 3n+3 = O(3n) is TRUE.

We have:

With c = 10, n0 = 0; g(n) =3n

3n+3 ≤ c.g(n) n ≥ n0.

<=> 3n+3 = O(3n)

1. 33n = O(3n) is FALSE.

Let f(n) = 33n. Assume that f(n) = O(3n), there will exist c and x0 that,

with g(n) = 3n we have: f(n) ≤ c.g(n) n ≥ n0.

<=> 33n ≤ c.3n

<=> (33)n ≤ c.3n

<=> 9n ≤ c

<=> n.log9 ≤ log(c)/log(9)

<=> n ≤ log(c) / log2(9)

So for every n ≥ log(c)/log2(9), 33n ≥ c.3n.

Therefore, 33n is not O(3n).

**Problem 6:**

1. Function count the number of an element x in an array of n integers.

(assume array **a[]** with **n** integers).

int count\_num(int a[], int n, int x){

if (n == -1) return 0;

if (a[n] == x) return count\_num(a, n-1, x) + 1;

return count\_num(a, n-1, x);

}

1. Function find the nth Fibonacci number.

(start from 1)

int fibo(int n){

if (n == 1 || n == 2) return 1;

return fibo(n-1) + fibo(n-2);

}

**Problem 7:**

In real life applications, correctness is more important than performance, because:

* The achivement of solving problem is assessed based on result. If the answer is wrong, it will be meaningless.
* We can wait for a long time to solve problem correctly, but we are not allowed to be fast and then make a wrong result. It will affect the entire process.
* For example, for a Banking Sector, if the transaction fails because of wrong program, this will lead to many problems then.

**Problem 8:**

We need to study algorithms and perfomance for those things:

* We will know many different algorithm, about searching, sorting, storing data, etc. in order to apply it in coding and fasten our applications.
* We will able to analyze and compare many algorithms to find out which algo. is the best for use.
* We can not only improve the current algorithms but also make other algorithms that can work better.

**Problem 9:**

**Problem: Print the reversed version of number X.**

***Example:***

***Input: 123 Output: 321***

Code (C++):

void write\_reverse(int x){

if ( x < 10) cout << x;

else{

write\_reverse( x % 10 );

cout << x/10;

}

}

**Problem 10:**

Code (C++):  
void write\_reverse\_iteration(int x){  
 int ans = 0;

while (x > 0){

ans = ans \* 10 + x % 10;

x / = 10;

}

cout << ans; //Print answer

}