***BÀI TẬP CHƯƠNG 1***

***Bài 1: Hãy đánh giá độ phức tạp của đoạn chương trình sau:***

*for (i = 0; i < n; i++)*

*for (j = 0; j < m; j++)*

*if (a[ i ][ j ] = = x) return 1;*

*return -1;*

- Số phép gán: n \* m => O(n \* m)

- Số phép so sánh: n \* m + 1 => O(n \* m + 1)

- Số phép tính cơ bản: 2 => O(2)

- Số vòng lặp: 2, vì hai vòng lặp long nhau nên => O(n \* m)

- Độ phức tạp của hàm if: O(1)

Đô phức tạp của đoạn code : O(n \* m + n \* m + 1 + 2 + n \* m + 1) = O( 3(n \* m) + 4) = **O(n \* m).**

***Bài 2: Hãy đánh giá độ phức tạp của đoạn chương trình sau:***

*sum = 0;  
for( i = 0; i < n ; i++)*

*for( j = 0; j < i ; j++)*

*sum++;*

- Số phép gán: n => O(n)

- Số phép so sánh: n => O(n)

- Số phép tính cơ bản: 3 => O(3)

- Số vòng lặp: 2, vì hai vòng lặp long nhau nên => O(n2)

Đô phức tạp của đoạn code : O(n + n + 3 + n2) = O(n2 + 2n + 3) = **O(n2).**

***Bài 3: Đánh giá độ phức tạp của đoạn code sau:***

*for (i = 0; i < n; i++)*

*sum1+=i;*

*for (i = 0; i < n\*n; i++)*

*sum2+=i;*

Số phép gán: 64

-N= 0 thì có 2 phép gán

-N=1 thì có 6 phép gán

-N=2 thì có 12 phép gán

-N = k thì có (k+2) . ( k+1) phép gán

T(n) = (n+2).(n+1)

O(n2)

Độ phức tạp của thuật toán là O(n2)

***Bài 4: Đánh giá độ phức tạp của hàm tính giai thừa sau:***

*int GT(int n)*

*{*

*if (n == 1)*

*return 1;*

*return n\*GT(n-1);*

*}*

Tức là giải PT: T(n) = C1 khi n=0 và T(n) = T(n-1)+C2 khi n>0

Ta có:  
T(n) = T(n-1) + C2  
= (T(n-2) + C2) + C2 = T(n-2) + 2C2  
= T(n-i) + iC2

PT kết thúc khi n-i = 0 hay n = i

=> T(n) = T(0) + nC2 = C1 + nC2

Theo quy tắc Big-O thì độ phức tạp của thuật toán là O(C1+nC2)=O(n)

***Bài 5: Đánh giá độ phức tạp của hàm tính dãy FIBONACCI sau****:*

*int Fibo(int n)*

*{*

*if (n <=1)*

*return n;*

*return Fibo(n-1) + Fibo(n-2);*

T(n) = 0 với n <= 1

T(n) = T(n-1) + T(n-2) + 1 với n > 1

Nhận xét: T(n) là hàm tăng theo số mũ. T(n) = O()

Suy ra: O() = O() + O() + O(1)

Bỏ qua O(1): = + ⇒ = a + 1 ⇒ a = 1.618

Vậy T(n) =