**Assignment - Week 1 Day 3**

**Question 1**. Write an algorithm

beautiful(A, n)

Input : An integer array with n elements

such that the best case running time is equal to the worst case running time. Write the algorithm and give your analysis to justify your claim.

**Solution:** The following findingMax algorithm has the same best and worst case, that is θ(n).

By using pseudocode;

Algorithm findMax(A, start, end)

Input arrayA with n integers

Output maximum element of A

if start=end then

return A[start]

if start+1=end then

return max(A[start], A[end])

mid←(start+end)/2

a←findMax(A, start, mid)

b←findMax(A, mid+1, end)

return max(a,b)

Using master theorem,

T(1) = 1

T(n) = 2T(n/2) + c

so a=2, b= 2 , k=0

b^k=2^0=1

As a>b^k, using the master theorem,

T(n) = θ(n^log2) =θ(n^1) =θ(n)

—--------------------------------—--------------------------------—--------------------------------

**Question 2.** Order them based on their complexity.

2^n , 2^(2n), 2^(n + 1), 2^( 2^n ) (Note: ^ stands for exponent operation. Example: 2^n = 2^n )

**Solution:**

1. O(2^n)
2. O(2^(n + 1)),
3. O(2^(2n))
4. O(2^( 2^n ))

—--------------------------------—--------------------------------—--------------------------------

**Question 3.** Mention one algorithm you know for each of the time complexities listed.

O(1), O(log n), O(n), O(n log n), O(n^2 ), O(n^3 ), O(2^n )

Example. O(n log n) : Quicksort

**Solution:**

* O(1): Indexing an array e.g A[0].
* O(log n):Binary Search
* O(n): GCD with Euclidean Algorithm
* O(n log n):Merge Sort, Quicksort
* O(n^2 ): Bubble Sort
* O(n^3 ): T(*n*) = 3*n^3* + 2*n* + 7
* O(2^n ): recursive calculation of Fibonacci numbers

—--------------------------------—--------------------------------—--------------------------------

**Question 4.** Apply Master Theorem and determine the time complexity of

fib(n) shown in slide 48.

**Solution:**

T(n) = T(n-1) + T(n-2) + c

= 2T(n-1) + c //from the approximation T(n-1) ~ T(n-2)

= 2\*(2T(n-2) + c) + c

= 4T(n-2) + 3c

The Master Theorem cannot be applied to the algorithm that finds the nth Fibonacci number because b is 1. The recursion decreases the problem statement by subtracting rather than by dividing into smaller problems.

So, we can prove using the induction hypothesis.

Assume F(n) > (4/3)^n for n>4

To prove that assumption,

F(1)=1

F(2)=1

F(3)=2

F(4)=3

Step 1. Base case(s).

Let n = 5.

Then LHS = F(5)=F(4)+F(3)=3+2=5 > (4/3)^5=4.21=RHS

Let n = 6.

Then LHS = F(6)=F(5)+F(4)=5+4=9 > (4/3)^6=5.62=RHS

Hence the base cases are proved.

Step 2. Induction hypothesis.

Assume the result is true for all values of n in the interval [5, m]

Step 3. Induction step.

F(m+1) = F(m)+F(m-1)

> (4/3)^m + (4/3)^(m-1)

= (4/3)^(m-1)[(4/3)+1]

= (4/3)^(m-1)[7/3]

= (4/3)^(m-1) [21/9]

> (4/3)^(m-1)[16/9]

= (4/3)^(m-1) (4/3)^2

= (4/3)^(m+1)

So, F(m+1) > (4/3)^(m+1)

Therefore, the time complexity for fib algorithm Ω(r^n) for some r>1. In other words, fib is an exponentially slow algorithm.

—--------------------------------—--------------------------------—--------------------------------

**Question 5**

Solve the recurrence

T(1) = 1

T(n) = 2T(n/2) + c

without using the Master Theorem.

Hint: Assume n = 2^k.

Follow the steps I did just before lunch break. Only difference is I did for 32. You are doing it for any perfect power of 2.

**Solution:**

T(1) = 1

T(n) = 2T(n/2) + c

Assume n=2^k

Let k=7 then n=2^6 =64

T(2^6) =2T(2^5)+c (i)

2T(2^5) =2[2T(2^4)+c]=4T(2^4)+2c (ii)

4T(2^4) =4[2T(2^3)+c]=8T(2^3)+4c (iii)

8T(2^3) =8[2T(2^2)+c]=16T(2^3)+8c (iv)

16T(2^2) =16[2T(2)+c]=32T(2)+16c (v)

16T(2) =32[2T(1)+c]=64T(1)+32c (vi)

Adding all equation s from (i) to (vi)

T(2^6) =c+2c+4c+8c+16c+32c+64T(1)

=64T(1)+63c , T1=1

=64+(64-1)c

=n+(n-1)c

= θ(n)

—--------------------------------—--------------------------------—--------------------------------

**Question 6.** Practice Master theorem. It is a very important result in Analysis of algorithms. There are many resources on the internet. Show three different examples covering three possible cases. Show your detailed work.



……… 1st case



………. 2nd case

…… 3rd case