CS 315 : Design and Analysis of Algorithm.

Tutorial 01. - S/17/350

(01). $37 < N\log_2(\log n) < N^2 \log N < N^3 < 2^{N/2}$.

(02). Theoritical Approach.

(i) Get a general description of algorithm [redacted] for an implementation of an algorithm.

(ii) Observe the pseudo Code and assign a Cost for each statement.

(iii) By Considering the total Cost by finding the total number of times each statement is executed.

(iv) we can determine the maximum number of primitive operations executed by an algorithm as a function of the input size (n).

(v) This is defined as T(n).

(03). Algorithm intersection (Array1, Array2)
Input Sequence of 2 Array elements (Array01, Array02)
Output Array representing the intersection of the 2 arrays.

Create an empty array - intersection.
for $i \leftarrow 0$ to length[Array1] do

~~array1[i]~~ ~~array~~

for $j \leftarrow 0$ to length[Array2] do

If array1[i] in in array2

Append to empty array intersection

return intersection

No: ________ Date: __/__/__

Python Code.

```
def intersection(array1, array2):
    intersection = []
    for i in array1:
        if i in array2:
            intersection.append(i)
    return intersection
```

(4) (a)

```
def function(n)
    sum = 0                      → 1
    i = 1                        → 1
    for k in range(0, n, 2*i):   → n/2
        sum = sum + 1            → n/2
        i *= 2                   → n/2
    return sum                   → 1
```

$T(n) = \frac{3n}{2} + 2$

$T(n) = O(n)$

(b) def function(n):

```
sum = 0                          → 1
for i in range(0,n):             → n
    for j in range(0, i^2):      → [1+4+9+...+n^2]
        for k in range(0,j):     → [1+2+...+n^2]
            sum = sum + 1        → [1+2+...+n^2]

return sum                       → 1
```

$T(n) = 2 + n + [1+4+\dots+n^2] + 2[1+2+\dots+n^2]$

$= 2 + 2 + \frac{n(n+1)(2n+1)}{6} + \cancel{2} \times \frac{n^2(n^2+1)}{\cancel{2}}$

$T(n) = n^4.$

(c) def function(r,n):

```
sum = 0                          → 1
for i in range(0,n):             → n
    for j in range(0,r):         → n*r
        sum = sum + 1            → n*r
return sum                       → 1
```

$T(n) = 2 + n + 2nr$

$= (2r+1)n + 2 = 2nr + n + 2$

$T(n) = O(n).$

No: Date: / /

05 (a) $O(n^2)$

(b) $O(n\log n)$

(c) $O(n^{3/2})$

(d) $O(2^n)$

(e) $O(n(\log n)^2)$

06 (b) $T_n = 5n\log n + 8n - 200.$

by definition, we must find that C, n. positive Constant s.t $T(n) \leq C\,g(n)$. So here $g(n) = n\log n$.

$5n\log n + 8n - 200 \leq Cn\log n$

$5 + \frac{8}{\log n} - \frac{200}{n\log n} \leq C$

let $n = 100$

$5 + 4 - \frac{2}{2} \leq C$

$8 \leq C$

$\therefore n = 100, \; C = 8$

CamScanner

No : ...... Date: / /

(c) $T_n = 500n + 100n^{3/2} + 50n\log_{10}n$

by definition, we have to find c, n_0 positive constant such that $T(n) \leq c\,g(n)$

There $g(n) = n^{3/2}$

$$T(n) \leq c\,g(n)$$

$$500n + 100n^{3/2} + 500n\log_{10}n = cn^{3/2}$$

$$\frac{500}{\sqrt{n}} + 100 + \frac{50\log_{10}n}{\sqrt{n}} \leq c$$

assume $n = 1$

$$500 + 100 + 0 \leq c$$

$$600 \leq c$$

$\therefore n_0 = 1,\ c = 600$

07

31, 41, 59, 26, 41, 58.

31	41	59	26	41	58

Iteration 1
Step 0.

31	41	59	26	41	58

Iteration 2
Step 0.

31	41	26	59	41	58

Iteration 3
Step 0.

31	26	41	59	41	58

Iteration 3
Step 1.

26	31	41	59	41	58

Iteration 3
Step 2.

26	31	41	41	59	58

Iteration 4
Step 0.

26	31	41	41	58	59

Iteration 5
Step 0.