14/8/21

DATE

Tutorial Sheet - 1.

① int a = 0 , b = 0;

for (i = 0 ; i < N ; i++)   a += rand();

for (j = 0 ; j < M ; j++)   b += rand();

Time complexity = O(N + M);

space Comp = O(1)   Ans.

② int sum = 0 , i;

for (i = 0 ; i < n ; i = i + 2) { sum += i; }

↳ loop will sum ~~odd~~ even times.

∴ 0 + 2 + 4 ---- 2n.

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

Time Complexity

O(n/2) ⟹ O(n) Ans

③ int sum = 0, i;

for (i = 0 , i < n; i = i * 2) { sum += i }

Time Complexity = ~~O(n²)~~ . O(log n) Ans

④ int sum = 0, i

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

O(log n) Ans.

⑤ int j = 1, i = 0;

while (i <= n) {

i = i + j,

j++,

}

Time = O(n)

Space = O(1)

Ans

⑥

```
void recursion(int n){          → T(n)
    if(n==1) return;            → T(1)
    recursion(n-1);             → T(n-1)
    print(n);                   → T(1)
    recursion(n-1);             → T(n-1).
}
```

$$\boxed{T(n) = T(n-1) + 1}$$

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

$T(n) = T(n-2) + 2$ $\qquad \boxed{T(1) = 1}$

$\vdots$

$T(n) = T(n-k) + k$

$O(n)$ Ans.

⑦ The given Recursive funⁿ is of binary search

$\therefore$ Time complexity = $\cancel{O(n \log n)}$)

$$\boxed{T\left(\frac{n}{2}\right) + 1}$$

or

n is dividing by 2 every time

$\therefore$ $O(\log n)$ Ans

⑧ ① $T(n) = T(n-1) + 1$

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

$\therefore T(n) = T(n-2) + 2$

$T(n) = T(n-k) + k$

$= O(n)$ Ans

② $T(n) = T(n-1) + n$

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

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

$T(n-2) = T(n-4) + 2n - 5$

$T(n) = T(n-4) + 4n - 6$

$\vdots$

$T(n) = T(n-k) + kn - (k-1)(k)$

③ $T(n) = T(n/2) + 1$

$a = 1,\ b = 2,\ k = 0,\ p = 0$

$a = b^k$

$O(n^{\log_2 1} \cdot \log^{p+1} n)$

$O(\log n)$ Ans

④ $T(n) = 2T(n/2) + 1$

$a = 2 \quad b = 2 \quad k = 0$

$2 > 1 \quad (a > b^k)$

$O(n^{\log_2 2}) = O(n)$

Ans

⑤ $T(n) = 3T(n-1)$

$T(0) = 1$

$T(n-1) = 3T(n-2)$

$T(n) = 3(3T(n-2))$

$T(n) = 3^k T(n-k)$

$n - k = 0$

$\boxed{n = k}$

$3^n \times 1 = T(n)$

$T(n) = 3^n$

$\therefore O(3^n)$ Ans.

14/8

DATE

(9) $O(n)$

(10) $O(N * N)$

(11) $O(N \log N)$

(12) X will be always a better choice for ~~small~~ ~~large~~ inputs large

(13) $O(\log N)$

(14)

(14) $T(n) = 7T(n/2) + 3n^2 + 2$

using <u>Master's Theorem</u>

$a = 7$, $b = 2$; $k = 2$

$a > b^k$

$7 > 2^2$ = True

$T(n) = \Theta(n^{\log_2 7})$

$= \Theta(n^{\wedge}2.8)$ Ans.

(15) $f1(n) = n^{\wedge}\sqrt{n}$

$f2(n) = 2^{\wedge}n$

~~f3~~ $f3(n) = (1.00001)^{\wedge}n$

$f4(n) = n^{\wedge}(1,0 * 2^{\wedge}(n/2))$

$f4 > f2 > f3 > f1$

(16) $f(n) = 2^{\wedge}(2n)$

$\Omega(2^{\wedge}n)$

Vardhman

PAGE

(17) $T(n) = 2T(n/2) + n^2$.

$a = 2, \quad b = 2, \quad k = 2$.

$2 < 2^2$ $\quad \boxed{p = 0}$

$T(n) = \Theta(n^k \log^p n)$

$T(n) = \Theta(n^2)$ Ans

(18)
```
int gcd (int n, int m) {
    if (n % m == 0) return m;
    if (n < m) swap (n, m);
    while (m > 0) { n = n % m;
                    swap (n, m); }
    return m; }
```

↳ here n is gradually / exponentially decreasing

$\therefore$ $O(\log n)$ Ans.

(19)
```
int a = 0, b = 0;
for (i = 0; i < N ; i++) {          → O(n)
        for (j = 0; j < N ; j++)    → O(n)
              a = a + j;
}                                     O(n^2)

for (k = 0 ; k < N ; k++
      b = b + k                     ← O(n)
}
```

$\therefore$ $\cancel{O(n^2)}$

$O(n * n + n)$

$O(n^2 + n)$

$= O(n^2)$ Ans