=

Name: M. Shafeen Roll No: 22P-9278

Section: BAI-5A

Assignment #02

Q1)

1.1

a) $f(n) = 3n^3+5n^2+7$

Sol:- Asymptotic Complexity: $O(n^3)$.

In this case we can say that the time-complexity is $O(n^3)$ because of the highest degree polynomial which is $3n^3$ here but other lower order terms and some constants can be ignored which leaves n^3 in the end

hence Answer $= O(n^3)$.

b) $(g(n) = 2^{\sqrt{n}}$

Sol:

Asymptotic Complexity: $O(2^{\sqrt{n}})$

The Constant 2 raise to the power $\sqrt{n}$ doesn't change because 2 is a constant value and also because $2^{\sqrt{n}}$ is an exponential function with a growth rate higher than polynomials.

(c) $h(n) = n\log^2(n)$

Solution:

Asymptotic Complexity : $O(n\log^2(n))$

While $\log^2(n)$ grows slower, than n,

we multiply it by n results in a function

that still grows faster, than $\log(n)$ but slower

than n^2.

(d) $K(n) = n!$

Asymptotic Complexity $(O)(n^n)$

Sol.

The property of $n!$ is : $1\times2\times3\times4\cdots$

for some constant lets say 4.

but we also know

$n! = n\times(n-1)\times(n-2)\cdots2\times1$: The growth

of every term is linear with n and

as there are n terms. The the higher

bound would be n^n which grows

faster. ~~than~~

$O(n^n)$.

1.2) $f(n) = O(g(n))$

let.

$f(n) = 10\log n + 1000$ and $g(n) = \log n$

$f(n)$ is $O(\log n)$

$f(n) \le C_1 g(n)$

$10 \log n + 1000 \le C_1 g(n)$

$10 \log n + 1000 \le C_1 \log n.$

As n grows large, the term $3 \log n$ dominates the constant term 100.

There exists a constant $c > 0$ (e.g $c = 501$) and on $n_0 \ge 0$ (e.g $n_0 = 1$) such that

$3\log n\ 100 \le 301 \log n$ for all $n \ge 1$)

for $f(n)$ is Not $O(g(n))$

let

$f(n) = 3n^2 \quad g(n) = n$

$3n^2 \le C_1 n$

As n grows large. $3n^2$ grows much faster than n.

There does not exist any constant $C > 0$

Such that $3n^2 \le Cn$. for all sufficiently large n.
This violate the definition of O-notation
proving that $f(n)$ is not $O(n)$.

1.3) $p(n) = 5n^2 + 3n + 1$

Sol.

function = polynomial

→ highest power of $n = 2$; finite power.

→ Consists of terms ~~are~~ raised to integer power.

$q(n) = 4^n$

Sol:

function = exponential

→ Consists of a term which is exponential.

→ base 4 = Constant and exponent (n) raises.

$r(n) = \log(n) \cdot n$

Sol.

function: Not polynomial Nor Purely exponential

→ And $\log(n)$ is a logarithmic function,
not a polynomial or exponential function.

$3(n) = \sqrt{n} \cdot 2^n$
Sol:
$3(n) = \sqrt{n} \cdot 2^n$

function = Exponential.

Expontential term 2^n ~~always~~ will always dominate the term $\sqrt{n}$.

Q 2)
2.1) Determine the time Complexity.
Sol:
Code : In assignment
→ Outer loop will iterate n times with an added +1 for ~~it~~ ~~the~~ the false condition
↳ Inner loop = n-1 times (ea)
Total iteration

$$1+2+3 \dots n - \frac{n(n+1)}{2}$$

$\boxed{\text{Time Complexity } O(n^2)}$

2.2) Code. In assignment
↳ Outer loop: Runs $(n+1)$ Times.
Inner loop: $(n+1)$ times.
Inner loop: $(n+1)$ times
Total iteration

$$\sum_{i=1}^{n} \sum_{k=0}^{n-1} (n-k) =$$

Time Complexity

$\boxed{O(n^3)}$.

(2.3) Code: In assignment

↳ Outer loop: $(n+1)$ times

↳ Middle loop: $(n+1)$ times.

↳ Inner loop: $\sum_{i=1}^{i} (i+1)$ times.

Total

$$\sum_{i=1}^{n} \sum_{k=1}^{i} (i-k+1)$$

Time Complexity = $\boxed{O(n^3)}$.

(2.4) Code: In assignment.

↳ Outer loop: $(n+1)$ times

↳ Inner loop: $n/2$ times

So time Complexity = $\boxed{O(n^2)}$

(2.5) Code: In Assignment.

$$\sum_{i=1}^{n-1} \sum_{j=1}^{i-1} \sum_{k=1}^{j-1} 1$$

$$\sum_{i=1}^{n-1} \frac{i(i+1)}{2}$$

$$\sum_{i=1}^{n-1} i^2$$

$$= \frac{n(n-1)(2n-1)}{6}$$

$$= n^3$$

Time Complexity of this algorithm will be $O(n^3)$

3.1)

Prove or disprove that the function

$$T(n) = 5 \cdot 2^{n/2} + 3n^2 \text{ is } O(2^n).$$

Sol:

To check if $T(n)$ can be upper-bounded by 2^n

$$T(n) \leq c \cdot 2^n$$

$$5 \cdot 2^{n/2} + 3n^2 \leq c_1 \cdot 2^n$$

$$\frac{5 \cdot 2^{n/2}}{2^n} + \frac{3n^2}{2^n} \leq c$$

$$5 \cdot 2^{n/2 - n} + \frac{3n^2}{2^n} \leq c_1$$

$$5 \cdot 2^{-\frac{n}{2}} + 3\frac{n^2}{2^n} \leq c_1$$

$$5 \cdot 2^{\frac{-(1)}{2}} + 3\frac{(1)^2}{2^1} \leq c_1$$

$$5.03 \leq c_1$$

$$5 \leq c_1 \quad \text{positive value.}$$

Thus, we can prove that $T(n) = 5 \cdot 2^{n/2} + 3n^2$ is ~~exact~~

$$\boxed{O(2^n)}$$