Assignment

Name : Desireddy Sai Saukeerthana
Student ID : 50465523

problem 1:

1(a) :

(i) $f(n) = \log_2(n^3)$ $\quad g(n) = 10\log_2 \sqrt{n}$

O	Ω	Θ
Yes	Yes	Yes

$3\log_2 n$ $\quad 10\log_2 (n)^{1/2}$

$3\log_2 n$ $\quad \cancel{\frac{10}{2}}^{5} \log_2 n$

$3\log_2 n \le 5\log_2 n \Rightarrow f(n) \le g(n) \Rightarrow O \Rightarrow f(n) = O(g(n))$

ignore constants $\log_2 n = \log_2 n \Rightarrow f(n) = g(n) = \Theta \Rightarrow f(n) = \Theta g(n)$

(ii) $f(n) = 5n^2+n$ $\quad g(n) = n\log n$

O	Ω	Θ
No	Yes	No

$n^2 \ge n$

$f(n) \ge g(n) \Rightarrow f(n) = \Omega g(n)$

(iii) $f(n) = 10n^2+n+10$ $\quad g(n) = n^3$

taking highest coefficient in f(n)

O	Ω	Θ
Yes	No	No

$n^2 \le n^3$

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

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

(iv) $f(n) = e^n$ $\quad g(n) = 2^{2n}$

O	Ω	Θ
Yes	No	No,

$n = 1$

$e^1 \le 2^2$

$\therefore$ e^1 value is 2.71

$2.71 \le 4$ $\quad f(n) \le g(n)$

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

1(b) : $[10n\sqrt{n}] = O(n\sqrt{n})$

$f(n) = \lceil 10n\sqrt{n} \rceil$

if we want to prove using the definition of O-notation ~~Should~~ to satisfy $O(n\sqrt{n})$ it should ~~$f(n) \le c \cdot g(n)$~~ Satisfy $f(n) \le c \cdot g(n)$ where $g(n) = n$

So for every $c >= 11$ and $n \ge 1$ the value of $f(n)$ will always be less than $c * g(n)$

hence we can say $f(n) <= c * g(n) \Rightarrow O(n)$

problem 2 :

```
undirected_cycle_graph (v, visited, parent)
    mark vertex(v) as "visited" [visited[v] = True]
    add v to visited_vertices_list [visited_vertices_list.append(v)]
    for all neighbours x of v do
        if x is equal to parent then
            continue
        if x is "not visited" then
            undirected_cycle_graph (x, visited, v)
        else if x is not equal to parent then
            return true
    return false.
```

problem 3 :

visited [v] ← true , top ← top+1 , stack [top] ← v, pos[v] ← top

for every outgoing edges (v, u) of v do

if pos [u] ≠ 0 then

exit the whole algorithm, by returning the cycle stack [pos[u] . . . top]

elseif visited [u] = false then

DFS (u)

pos [v] ← 0 top ← top − 1

the DFS for a vertex v have running time $O(1+d^{out}(v))$, where $d^{out}(v)$ is the out - degree of v. Every v will be handled in atmost one iteration of DFS so the running time will be $O(n+m)$ // $n = O(m)$