Q. No. 1.1.

a) Solu^n:-

Given:-

$f(n) = 3n$ and $g(n) = n^3$.

We have:- $O(g(n)) = \{f(n) \mid \exists$ positive constants c and n_0, such that $0 \le f(n) \le c\,g(n), \forall n \ge n_0\}$.

$\Omega(g(n)) = \{f(n) \mid \exists$ positive constants c and n_0, such that $0 \le c\,g(n) \le f(n), \forall n \ge n_0\}$.

$f(n) = o(g(n))$ implies $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$. (Non-tight upper

$f(n) = \omega(g(n))$ implies $\lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$ and lower bound).

For given, functions we can state that.

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty} \frac{3n}{n^3} = \lim_{n\to\infty} \frac{3}{n^2} = 0.$$

This implies, $f \in O(g(n))$ and $f \in o(g(n))$.

Also, $\lim_{n\to\infty} \frac{g(n)}{f(n)} = \lim_{n\to\infty} \frac{n^3}{3n} = \infty$

This implies, $g \in \Omega(f(n))$ and $g \in \omega(f(n))$.

So, for these pairs of function:-

$f \in O(g)$, $f \in o(g)$, $g \in \Omega(f)$ and $g \in \omega(f)$ satisfies.

b) Solu^n:-

Given =

$f(n) = 7n^{0.7} + 2n^{0.2} + 13\log n$.

$g(n) = \sqrt{n}$.

We have:-

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \frac{7n^{0.7} + 2n^{0.2} + 13\log n}{n^{0.5}}$$

$$= \lim_{n\to\infty} 7n^{0.2} + \frac{2}{n^{0.3}} + \frac{13\log n}{n^{0.5}}$$

$$= \lim_{n\to\infty} 7n^{0.2} + \lim_{n\to\infty} \frac{2}{n^{0.3}} + \lim_{n\to\infty} \frac{13 \times \frac{1}{n}}{0.5 \times n^{-0.5}}$$

$$= \infty + 0 + \lim_{n\to\infty} \frac{13}{\sqrt{n}}$$

$$= \infty + 0 + 0 = \infty.$$

Also;

$$\lim_{n\to\infty} \frac{g(n)}{f(n)} = \lim_{n\to\infty} \frac{n^{0.5}}{7n^{0.7} + 2n^{0.2} + \log(n)}$$

Using L'hopital's rule, we get:-

$$\lim_{n\to\infty} \frac{1}{2\sqrt{n}\left(\frac{2}{5n^{4/5}} + \frac{49}{10n^{3/10}}\right)}$$

$$= \lim_{n\to\infty} \frac{5n^{3/5}}{4n^{3/10} + 49n^{4/5}}$$

Multiplying by $n^{4/5}$ and also dividing by $n^{4/5}$.

$$5 \lim_{n\to\infty} : \frac{\frac{n^{4/5} \cdot n^{3/5}}{n^{4/5}}}{n^{4/5} \times \frac{1}{n^{4/5}}\left(4n^{3/10} + 49n^{4/5}\right)}$$

$$= 5 \lim_{n\to\infty} \frac{1}{n^{0.2}\left(49 + \frac{4}{\sqrt{n}}\right)}$$

Clearly, a constant is being divided by large number.

So, $\lim_{n\to\infty} \frac{g(n)}{f(n)} = 0$.

This implies, $f \in \Omega(g)$ and $f \in \omega(g)$ also, $g \in O(f)$ and $g \in o(f)$.

c) Solu^n:-

Given ~~equations are~~ func. are:-

$$f(n) = \frac{n^2}{\log n} \quad \& \quad g(n) = n \log n.$$

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \frac{n^{\cancel{2}}}{\log n \times \cancel{n} \log n} = \frac{n}{(\log n)^2}$$

Using L'hopitals rule, we get.

$$\lim_{n\to\infty} \frac{1}{\frac{2\log n}{n}}$$

Again, using L'hopitals rule, we get:-

$$\lim_{n\to\infty} \frac{n1}{\frac{2 \cdot 1}{n}} = \infty.$$

Again, for;

$$\lim_{n\to\infty} \frac{g(n)}{f(n)} = \frac{(\log n)^2}{n^2}$$

~~the~~ Using L' Hopital ~~rule~~ rule.

$$\lim_{n\to\infty} \frac{2 \log n \times \frac{1}{n}}{1.}$$

$$\Rightarrow \lim_{n\to\infty} \frac{2\frac{1}{n}}{1}$$

$$= \lim_{n\to\infty} \frac{2}{n} = 0.$$

So, as, $\lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$ and $\lim_{n\to\infty} \frac{g(n)}{f(n)} = 0$. so;

~~$f \in O(g(n))$ and $f \in o(g)$ and $g \in \Omega(f)$ and $g \in \omega(f)$~~

$f \in \Omega(g)$, $f \in \omega(g)$, $g \in o(f)$ and $g \in O(f)$.

d) Solution:-

$f(n) = (\log(3n))^3$

$g(n) = 9 \log n$

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \frac{(\log(3n))^3}{9\log(n)}$$

Using L'hopital's rule, we get:-

$$\frac{1}{9} \lim_{n\to\infty} 3 \log(3n)^2.$$

$$= \frac{1}{3} \lim_{n\to\infty} \log(3n)^2 = \infty.$$

Again:-

$$\lim_{n\to\infty} \frac{g(n)}{f(n)} = \lim_{n\to\infty} \frac{9 (\log(n))}{(\log(3n))^3} = 9 \lim_{n\to\infty} \frac{\frac{1}{\cancel{n}}}{\cancel{3}\log(3n)^2 \times \frac{1}{\cancel{3n}}}.$$

$$= 9 \frac{1}{\infty} = 0.$$

Since, $\lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$ and $\lim_{n\to\infty} \frac{g(n)}{f(n)} = 0$ so;

~~$f \in O(g)$ and $f \in o(g)$ and $g \in \Omega(f)$ and $g \in \omega(f)$.~~

$f \in \Omega(g)$, $f \in \omega(g)$, $g \in O(f)$ and $g \in o(f)$.

1.2 a). Solu^n:-

```
Selection_sort ( A, size) {
    for i = 0 to (size -1)
        iMin = i
        for j = i+1 to size
            if ( A[j] < A[iMin] ){
                iMin = j;
            }
        temp_var = A[i]
        A[i] = A[iMin]
        A[iMin] = temp_var.
```

2b) Solu^n:-

Loop invariant needs three conditions to be satisfied.

1) Initilization
2) Maintainance
3) Termination,

① Initilization :-

We start by showing that the loop invariant holds before the first iteration, when i = 0. The sub-array ~~A[0...n-1]~~ contains only ~~one~~ zero element ~~arr[0]~~, which is infact the ~~original element~~ satisfies the condition. Moreover, this sub-array is sorted, which shows that loop invariant holds prior to the first iteration.

② Maintaencence :-

~~Assume, Loop invariant holds, for some input 'k'. Then, Let's check, it holds for 'k+1'~~

To see that each iteration mantaine the loop invariants, lets see the condition inside the loop i.e. if (arr [j] < arr [iMin]). This condition checks if ~~arr[2] < arr[1]~~ arr [1] < arr [0] in the first iteration, if holds true, then assigns, 'iMin' i.e. the index position of the min value in Array to the 'jth' index of the array which contain's the minimum value, and then swaps the respective index's value. So, in the loop, if we take an example, then following things are happening.

min-value

5	7	3	2	5	4	8	9	10	11

← At, the start of the iteration.

iMin = i = 0

First, it iterates through the Loop and only, when the condition is satisfied, then the value of iMin is changed to the ~~arr[j]~~ j th position, which points to 'min' value and then their value is swapped.

~~Now for~~

2	7	3	5	6	4	8	9	10	11

← At the start of 2nd iteration.

iMin

So, in second iteration, again iMin is setted to ith position and min in (n-1) array is searched and value is replaced.

In, third iterteration, again iMin is setted to 2rd index and min in (n-2) array is searched and value are swapped

So, we showed that it is true for next iteration if it is true to prior iteration.

Termination :-

As it reaches the end of loop i.e. $i < (n-1)$, then every element prior to it will be sorted, which satisfies the condition. As the last element will automatically be placed in it's right position, so, the result of this Selection-sort algorithm will be correct.

1.2 c) For generating the random sequence, I used C progamming language's 'stdlib.h' headear file rand() function. I passed large enough input ~~array~~ number and passed in each function that generates best case, worst case and average (random-case).

Depending on the input, the function returns a best case- in which elements of array is arranged in increasing order. In ~~best~~ worst case, elements of array is arranged in decreasing ~~wo~~ order.

In average random case, elements of array is arranged in random order.

12) e) → Solun:-

From the graph, we can state that the difference between the cases are minimal. ~~As for the number of times the loop~~ As the loop has a loop inside it for determining worst-case and base case too, so the we can conclude that the ~~bes~~ time complexity is $O(n^2)$.