Khairul Islam

1)

$$\frac{2}{N} < 37 < \sqrt{N} < N < N\log(\log(N)) < N\log N < N\log(N^2)$$
$$< N\log^2(N) < N^{1.5} < N^2 < N^4 < 2^{N/2} < 2^N$$

2)

a) $O(N) \Rightarrow \frac{2000}{500} = \frac{N}{10}$

$N = 40$ seconds

b) $O(N\log(N)) \Rightarrow \frac{2000\log(2000)}{500\log(500)} = \frac{N}{10}$

$$N = 40 \cdot \frac{\log(2000)}{\log(500)}$$

$N = 48.9228$ seconds

c) $O(N^2) \Rightarrow \frac{(2000)^2}{(500)^2} = \frac{N}{10}$

$$N = 64 \times 10$$

$N = 640$ seconds

3)

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

... and so on

This recurrence relation goes until $T(0) = 0$.

and so, This recurrence relation execute in the runtime "O(n)" because each iteration takes "n" elements.

Therefore, the pseudocode for sorting takes about "O(n)".

4)

a)

Run time of

f(): O(n): it is recursive program that runs n time for n = 0 to n.

g(): O(n): it has a loop that runs n times i = 0 to n-1.

b) Space complexity:

f(): O(n): since it is recursive calling n times, it takes n system stack.

g(): O(1); constant.

c)

```
int h(int n) {
    return n*(n-1)/2;
}
```

it takes O(1) time and O(1) Space.

5)

Recurrence relation of f(n)

$T(n) = T(n/2) + O(1)$

So, it's time complexity is $O(\log n)$

Complexity of g(n)

loop iterates log(n) times

complexity of f(n) is also log(n)

hence time complexity is $\log(n) * \log(n) = \log^2(n)$

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

6)

1) Read the value of n
2) Initialize a variable K = 0
3) Create a boolean array of size 10 and initialize the boolean array with false values
4) Repeat the loop untill all the digits 0-9 are found; In other word, repeat the loop until the boolean array's values are all true.
5) As the loop-repeat, increment K by 1.
6) Multiply n with K and store the result in another varinble x.
7) Now take each digit in x by mod(%) 10 and update values in boolean array corresponding to digit as true.
8) Exit loop
9) Return K.

7)

① In the regular battle ship board it's shap is square and we have to use two loops inordr to scan the complete board.

thus · efficiency $O(n*n)$

② Same goes with the normal square boards too. For a regualer square boards of N×N dimensions efficiency of the algorithm to scan the battle field will be $O(n*n)$.

③ N×M the efficiency will be $O(n*m)$.

8)

(A) O(1) is enough because it happens once per unit.

(B) O(n) because you have to loop through the list n times and compare each number in the list with the given number - weather the two number are equals.

(C) case 1: if the list is shorted, then it is O(1) because the first element is the smallest number.

case 2: if the list is not shorted; then it is O(n) because you need to go thru the list n times.

(D) $O(n^2)$ because you have to check each elements
↳ loop (loop).

(E) O(n) because you only need to go thru the list once.