Q1)
```
i = 0
while (i < 2n):
    for j in range(i):
        k = 0
        while (k < √n):
            k = k+1
    i = i+1
```

$2+\frac{1}{2}$

$i=0$	$i=1$	$i=2$	$i=3$	....	$i=2n$
j ×	$j=0$ (1 time)	$j=0,1$ (2 times)	$j=0,1,2$ (3 times)		$j \Rightarrow 2n$ times
k ×	$k \Rightarrow \sqrt{n}$ times	$k \Rightarrow \sqrt{n}$ times	$k \Rightarrow \sqrt{n}$ times		$k \Rightarrow \sqrt{n}$ times

$$= (1 * 1 * \sqrt{n}) + (1 * 2 * \sqrt{n}) + (1 * 3 * \sqrt{n}) + \cdots + (1 * 2n * \sqrt{n})$$

$$= \sqrt{n}\,(\underbrace{1+2+3+\cdots+2n}_{\text{summation series of } 2n \text{ terms}})$$

$$\Downarrow$$

$$\frac{2n(2n+1)}{2} \qquad \left\{ \because \text{Summation series formula for } n \text{ terms} = \frac{n(n+1)}{2} \right\}$$

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

$$= \sqrt{n}\,(2n^2+n)$$

$$= 2n^2\sqrt{n} + n\sqrt{n}$$

Dominant term ⇓ so $O(n^2\sqrt{n})$

or you can also write $O(n^{5/2})$

Q2)
```
i = 1
while (i <= n*n):
    j = 0
    while (j < n):
        j = j*2
        for k in range(j):
            print("Hello")
    i = i+1
```

$i = 1$
$j \Rightarrow \log n$ times
$k \Rightarrow \log n$ times

$i = 2$
$j \Rightarrow \log n$ times
$k \Rightarrow \log n$ times

$i = 3$
$j \Rightarrow \log n$ times
$k \Rightarrow \log n$ times

. . . .

$i = n^2$
$j \Rightarrow \log n$ times
$k \Rightarrow \log n$ times

$= 1 * \log n * \log n + 1 * \log n * \log n + 1 * \log n * \log n + \ldots + 1 * \log n * \log n$

This pattern is being added n^2 times, hence

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

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

or you can also write : $O(n^2 \log^2 n)$

or " " " " : $O(n^2 (\log n)^2)$

Q3)
```
i = 1
while (i <= n!):
    j = 1
    while (j <= n!):
        j = j+1
        m = 1
        while (m <= i):
            m = m+1
    i = i*2
```

Iteration 1	Iteration 2	Iteration 3	Iteration 4	....	Iteration k
$i=1$	$i=2$	$i=4$	$i=8$		$i=2^{k-1}$
$j \Rightarrow n!$ times	$j \Rightarrow n!$ times	$j \Rightarrow n!$ times	$j \Rightarrow n!$ times		$j \Rightarrow n!$ times
$k \Rightarrow 1$ time	$k \Rightarrow 2$ times	$k \Rightarrow 4$ times	$k \Rightarrow 8$ times		$k \Rightarrow 2^{k-1}$ times

$$= (1*n!*1) + (1*n!*2) + (1*n!*4) + (1*n!*8) + \cdots + (1*n!*2^{k-1})$$

$$= n!\,(\underbrace{1+2+4+8+\cdots+2^{k-1}}_{\text{Sum of power of 2 of } (k-1) \text{ terms}})$$

$$\Downarrow$$

$$2^k - 1 \quad \{\because 2^0+2^1+2^2+\cdots+2^k = 2^{k+1}-1\}$$

$$\therefore \; = n!\,(2^k - 1) \quad \text{—— (1)}$$

But what is k?

We know $i = 2^{k-1}$

Substitute in condition
$$i <= n!$$
$$2^{k-1} <= n!$$
$$\frac{2^k}{\cancel{2}} <= n!$$
$$2^k <= n!$$
$$k <= \log(n!)$$

Now that we have 'k', substitute in equation (1) to get

$$n!\,(2^k - 1) = n!\,(2^{\log(n!)} - 1)$$
$$= n!\,(n! - 1)$$
$$= \underbrace{n!\,n!}_{\text{Dominant term}} - n!$$

so $O(n!\,n!)$ or you can also write $O((n!)^2)$

Q4)
```
i = 1
while (i < log n):
    j = 0
    while (j < i):
        j = j + 1

    k = 0
    while (k < 2^n)
        k = k + 1

    i = i + 1
```

$i=1$	$i=2$	$i=3$	. . . . .	$i=\log n$
$j=0$ (1 time)	$j=0,1$ (2 times)	$j=0,1,2$ (3 times)		$j \Rightarrow (\log n)$ times
$k \Rightarrow 2^n$ times	$k \Rightarrow 2^n$ times	$k \Rightarrow 2^n$ times		$k \Rightarrow 2^n$ times

* j & k loops are not nested so their complexities need to be added before deducing the final value.

$$\therefore \; 1*(1+2^n) + 1*(2+2^n) + 1*(3+2^n) + \dots + 1*(\log n + 2^n)$$

$$= \underbrace{1+2+3+\dots+\log n}_{\text{Summation series of } \log n \text{ terms}} + \underbrace{2^n + 2^n + \dots + 2^n}_{\text{This value is repeated for } \log n \text{ number of terms so we can write it as:}}$$

$$\Downarrow$$

$$\frac{\log n(\log n+1)}{2} \quad \left\{ \because \text{for } n \text{ terms, it is } \frac{n(n+1)}{2} \right\} \qquad (2^n)(\log n)$$

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

$$= \underbrace{2^n \log n}_{\text{Dominant term}} + \frac{\log^2 n}{2} + \frac{\log n}{2}$$

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

Q5)
```
for i in range(n*n):
    print(i)
for j in range(4n):
    print(j)
for k in range(n-5):
    print(k)
```

Loops follow each other in which none of the following loop variables are dependent on the previous loop, so we see how many times each loop runs & add them together to get overall complexity

⇒ i loop runs n^2 times
⇒ j loop runs $4n$ times
⇒ k loop runs $(n-5)$ times

$\therefore n^2 + 4n + (n-5)$

$= n^2 + 5n - 5$

Dominant term

$\therefore O(n^2)$

Q6)

```
for i in range (n²):
    for j in range (4n):
        print (j)
    for k in range (n-5):
        print (k)
```

j & k loops follow each other in which the loop variables are independent, so we can add them together. However, both these happen as a nested loop(s) of i loop so i loop needs to be incorporated but not added, of course. In nested loops, if outer & inner loop variables are independent, we can simply multiply them

$\therefore$ Inner loops (j & k) : $4n+(n-5) = 4n+n-5 = 5n-5$ times

Outer loop (i) : n^2 times

$\therefore n^2 * (5n-5) = 5n^3 - 5n^2$

Dominant term

$O(n^3)$

Q7)

```
for i in range (n²):
    for j in range (i):
        print (j)
    for k in range (i+20):
        print (k)
```

$i = 0$
$j \times$
$K = 0, 1, 2, \cdots, 19$
(20 times)
$\approx 0 + 20$

$i = 1$
$j = 0$ (1 time)
$K = 0, 1, 2, \cdots, 19, 20$
(21 times)
$\approx 1 + 20$

$i = 2$
$j = 0, 1$ (2 times)
$K = 0, 1, \cdots, 20, 21$
(22 times)
$\approx 2 + 20$

. . . . . .

$i = n^2$
$j \Rightarrow n^2$ times
$K \Rightarrow (n^2 + 20)$ times

Inner loops (j & K) follow each other so can be added together

$$= 1 * \{\underset{j}{0} + \underbrace{(0+20)}_{K}\} + 1 * \{\underset{j}{1} + \underbrace{(1+20)}_{K}\} + 1 * \{\underset{j}{2} + \underbrace{(2+20)}_{K}\} + \cdots + 1 * \{\underset{j}{n^2} + \underbrace{(n^2+20)}_{K}\}$$

$$= (0 + 0 + 20) + (1 + 1 + 20) + (2 + 2 + 20) + \cdots + (n^2 + n^2 + 20)$$

$$= \underbrace{(0 + 1 + 2 + \cdots + n^2) + (0 + 1 + 2 + \cdots + n^2)} + \underbrace{(20 + 20 + 20 + \cdots + 20)}$$

this is being added n^2 times
∴ it can be written as
$20n^2$

$$= 2\underbrace{(0 + 1 + 2 + \cdots + n^2)}_{\text{Summation series of } n^2 \text{ terms}} + 20n^2$$

Summation series of n^2 terms

⇓

$\frac{n^2(n^2+1)}{2}$ $\{\because$ For n terms, it is : $\frac{n(n+1)}{2}\}$

$$= \cancel{2}\left(\frac{n^2(n^2+1)}{\cancel{2}}\right) + 20n^2$$

$$= n^4 + n^2 + 20n^2$$

$$= n^4 + 21n^2$$

Dominant term

$\therefore O(n^4)$

Q8)
```
for i in range(n):
    if i%2 == 1:
        print("Odd")
    else:
        print("Even")
```

For if-elif-else statements, we pick the Best & Worst case. However, when they are inside a loop, both will execute a certain number of times, so we derive the overall complexity.

In the given example, for n values, roughly half will be even & half will be odd, so if will run half of n times i.e. $\frac{n}{2}$ times & else will also run half of n times i.e. $\frac{n}{2}$ times. ~~For~~ For complete iterations of the loop, we get:

$$\frac{n}{2} + \frac{n}{2} = n \text{ times}$$

$$\therefore O(n)$$

Q9) for i in range (n//2):

```
for i in range(n//2):
    if i % 2 == 0:
        print("Even")
    else:
        for j in range(n):
            print(j)
```

Same logic for if-else, as explained in previous question. According to the given logic, approximately half of the values will be even, i.e. half of n//2 = $\frac{n}{4}$ times; and other half will be odd i.e. $\frac{n}{4}$ times.

$i = 0$ — if runs 1 time, j ✗

$i = 1$ — if ✗, else j runs n times

$i = 2$ — if ⇒ 1 time, j ✗

$i = 3$ — if ✗, else j ⇒ n times

. . . . . $i = n/2$

Or you can say if runs $\frac{n}{4}$ times i.e. $1 * \frac{n}{4} = \frac{n}{4}$

& else runs $\frac{n}{4}$ times i.e. $n * \frac{n}{4} = \frac{n^2}{4}$

∴ Total time $= \frac{n}{4} + \frac{n^2}{4}$ → Dominant term

∴ $O(n^2)$

Q10)
```
for i in range(n):
        if (i <= n//3):
            for j in range(i):
                print(j)

        else:
            for j in range(n):
                print(j)
```

$i=0$ if: ✓ j ✗ else: ✗

$i=1$ if: ✓ $j=0$ (1 time) else: ✗

$i=2$ if ✓ $j=0,1$ (2 times) else ✗

$i=3$ if ✓ $j=0,1,2$ (3 times) else ✗

.... $i=n/3$ if ✓ $j \Rightarrow \frac{n}{3}$ times else ✗

... $i=\frac{n}{3}+1$ if ✗ else ✓ $j \Rightarrow n$ times

$i=\frac{n}{3}+2$ if ✗ else ✓ $j \Rightarrow n$ times

$i=\frac{n}{3}+3$ if ✗ else ✓ $j \Rightarrow n$ times

$i=n$ if ✗ else ✓ $j \Rightarrow n$ times.

$$\therefore \underbrace{(1*1)+(1*2)+(1*3)+\cdots+(1*\tfrac{n}{3})}_{\text{Pattern when if is executed only}} + \underbrace{(1*n)+(1*n)+\cdots+(1*n)}_{\text{Pattern when else is executed}}$$

$$= \underbrace{\left(1+2+3+\cdots+\tfrac{n}{3}\right)}_{\text{Summation series for } \frac{n}{3} \text{ terms}} + \left(n+n+\cdots+n\right)$$

This starts after $i=\frac{n}{3}$ & goes to n, so $\left(n-\frac{n}{3}\right)$ times $=\frac{2n}{3}$ times $\therefore$ we can write

$\Downarrow$

$$\frac{\frac{n}{3}\left(\frac{n}{3}+1\right)}{2} \quad \left\{\because \text{For } n \text{ terms, it is } \frac{n(n+1)}{2}\right\}$$

$\Downarrow$

$$n * \frac{2n}{3}$$

$$\therefore \frac{\frac{n}{3}\left(\frac{n}{3}+1\right)}{2} + n\left(\frac{2n}{3}\right)$$

$$= \frac{n^2}{18} + \frac{n}{6} + \frac{2n^2}{3}$$

$$= \frac{13\,n^2}{18} + \frac{n}{6}$$

n^2 Dominant term

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