①

Somefunction(rows, cols)	Steps/exec	Freq	Total
for(i=1 ; i≤rows ; i++)	2	rows+1	2*(rows+1)
for(j=1 ; j≤cols ; j++)	2	rows*(cols+1)	2*rows*(cols+1)
Print (*)	1	rows*cols	rows*cols
Print(newline)	1	rows	rows
			3*rows*cols + 5rows + 2

$T(rows, cols) = 3 . rows . cols + 5 . rows + 2$

$T(rows, cols) = O(rows . cols) = \Omega(rows . cols) = \Theta(rows . cols)$

$T(rows, cols) = \Theta(rows . cols)$

$c_1 . (rows . cols) \leq 3\, rows . cols + 5 . rows + 2 \leq c_2 . (rows . cols)$

$c_1 = 5$ $\quad$ $c_2 = 6$

$rows_0 = 2$ $\quad$ $rows_0 = 2$

$cols_0 = 3$ $\quad$ $cols_0 = 3$

$T_{Best}(rows, cols) = T_{worst}(rows, cols) = \Theta(rows . cols)$

②

Somefunction(a , b)	Steps/exec	Freq	Total
if (b == 0)	1	1	1
return 1	1	1	1
answer = a	1	1	1
increment = a	1	1	1
for (i=1 ; i<b ; i++)	2	b	2b
for (j=1 ; j<a ; j++)	2	(b-1).a	2ab - 2a
answer += increment	1	(b-1).(a-1)	ab - b - a + 1
increment = answer	1	(b-1)	b - 1
return answer	1	1	1
			3ab + 2b + 5

$T(a,b) = 3 . a . b + 5b + 7$

$c_1 . (a . b) \leq 3ab + 2b + 5 \leq c_2 . (a . b)$

$c_1 = 3$ $\quad$ $c_2 = 5$

$a_0 = 3$ $\quad$ $a_0 = 5$

$b_0 = 3$ $\quad$ $b_0 = 5$

$T(a,b) = \Omega(a . b) = O(a . b) = \Theta(a . b)$

③

Some function(arr[], arr_len)	Steps/exec	freq	Total
Val = 0	1	1	1
for (i=0 ; i<arr_len/2 ; i++)	2	ceil(arr_len/2)+1	2*(ceil(arr_len/2)+1)
Val = Val + arr[i]	1	ceil(arr_len/2)	ceil(arr_len/2)
for (i=arr_len/2 ; i<arr_len ; i++)	2	floor(arr_len/2)+1	2*(floor(arr_len/2)+1)
Val = Val - arr[i]	1	floor(arr_len/2)	floor(arr_len/2)
if (Val >= 0)	1	1	1
return 1	1	1	1
else	1	1	1
return -1	1	1	1
			3*ceil(arr_len/2) + 3*floor(arr_len/2) + 7

$T(arr\_len) = 3.\ ceil(arr\_len/2) + 3.\ floor(arr\_len/2) + 7$

↓ linear ↓ linear ↓ constant

$T(N) = O(N) = \Omega(N) = \Theta(N)$ $\quad T(N) = \Theta(N)$

$T_{best}(N) = T_{worst}(N) = T_{av}(N) = \Theta(N)$

④

Some function(n)	Steps/exec	Freq	Total
c = 0	1	1	1
for (i=1 to n*n)	2	(n*n)+1	((n*n)+1).2
for (j=1 to n)	2	(n*n)*(n+1)	[(n*n)*(n+1)].2
for (k=1 to 2*j)	2	(n*n)*n*(n+1)	(n*n)*n*(n+1)*2
c = c+1	2	(n*n)*n*(n+1)	(n*n)*n*(n+1)*2
return c	1	1	1
			$4n^4+8n^3+4n^2+4$

$\sum_{j=1}^{n} 2j+1 = 3+5+\cdots+2n+1 = n.(n+2)$ $\quad \sum_{j=1}^{n} 2j = 2+4+\cdots+2n = n.(n+1)$

$T(N) = 4n^4 + 8n^3 + 4n^2 + 4 = O(n^4) = \Omega(n^4) = \Theta(n^4)$

$T(N) = \Theta(n^4)$

$T_{best}(N) = T_{worst}(N) = T_{av}(N) = \Theta(N^4)$

⑤

```
otherfunction(xp, yp)
    temp = xp
    xp = yp
    yp = temp
```
$T_1(N) = 3 = O(1) = \Omega(1) = \Theta(1)$

```
Somefunction(arr[], arr_len)
    for(i=0; i<arr_len-1; i++)
        min_idx = i
        for(j=i+1; j<arr_len; j++)
            if(arr[j] < arr[min_idx])
                min_idx = j
        otherfunction(arr[min_idx], arr[i])
```
T(N) = arr_len - i
T(N) = O(arr_len - i)
T(N) = Ω(arr_len - 1)
T(N) = Θ(arr_len - i)
otherfunction(arr[min_idx], arr[i]) → Θ(1)

$T(N) = \sum_{i=0}^{arr\_len-1} (arr\_len - i) = \frac{arr\_len \cdot (arr\_len+1)}{2}$

$= \Theta(arr\_len^2)$

$T(N) = \Theta(N^2)$

$T_{best}(N) = T_{worst}(N) = T_{av}(N) = \Theta(N^2)$

⑥

```
otherfunction(a, b)
    if b == 0 :
        return 1
    answer = a
    increment = a
    for i=1 to b :
        for j=1 to a :
            answer += increment
        increment = answer
    return answer
```
$T_1(N) = \Theta(1)$

$T_2(N) = \Theta(a.b)$

When we consider the whole algorithm this can't be true b always equals to 2

$T_{av} = P(T).\Theta(1) + P(F).\Theta(a.b)$

$T = \Theta(a.b)$

```
Somefunction(arr[], arr_len)
    for i=1 to arr_len :                    → arr_len
        for j=1 to arr_len :                → sum_{i=1}^{arr_len} (arr_len - i) = arr_len.(arr_len+1)/2 = n.(n+1)/2
            if otherfunction(n%i, n) == arr[i]
                Print(arr[i])
```
$T_{worst}(n) = T_{best} = T(n) = \sum_{i=1}^{\frac{n.(n+1)}{2}} 2.(n\%i)$

⑦

```
otherfunction(X, i)
    s = 0
    for(j=1; j<=i; j=j*2)
        s = s + X[j]
    return s
```
$T(N) = \lg(i)$

```
Somefunction(arr[], arr_len)
    for(i=0; i<=arr_len-1; i++)
        A[i] = otherfunction(arr, i) // (i+1)
    return A
```
$\sum_{i=1}^{arr\_len-1} \lg(i) = \lg(1) + \lg(2) + \cdots + \lg(arr\_len-1)$

$T(N) = \log(N!)$

$T_{best}(N) = T_{worst}(N) = T(N) = \Theta(\log(N!))$

⑧

```
Somefunction(n)
    res = 0
    j = 1
    if(n < 10)
        return n+10
    for(i=9 ; i > 1 ; i--)
        while(n%i == 0)
            n = n/i
            res = res + j * i
            j *= 10
    if(n > 10)
        return -1
    return res
```

Part 2.1

```
Part2_1(points[], size, aPoint)
    distance = findDistance(points[0], aPoint)
    minDistance = distance
    index = 0
    for(i=0 ; i<size ; i++)
        distance = findDistance(points[i], aPoint)
        if(minDistance > distance)
            minDistance = distance
            index = i
    return points[index]
```

$\Theta(1)$ for the first block; $\Theta(1)$ for the loop body; $\sum_{i=0}^{size} 1 = size \cdot 1$

$T_B(N) = T_W(N) \Rightarrow T(N) = \Theta(N)$

```
findDistance(P1, P2)
    Dx = P1.x - P2.x
    Dy = P1.y - P2.y
    distance = sqrt(pow(Dx,2) + pow(Dy,2))
    return distance
```

There is 9 operation for all P1, and P2

$\Theta(1)$

Part 2.2a

```
Part2_2a(arr[], size)
    if(size > 2)
        for(i=1 ; i<size-1 ; i++)
            if(arr[i-1] >= arr[i] && arr[i] <= arr[i+1])
                return i
    return -1
```

$\Theta(1)$ for the loop body; $\Theta(N)$

$T_W(N) = O(N) \quad T(N) = \Theta(N) \quad T_{Best}(N) = \Theta(1)$

Part 2.2b

```
Part2_2b(arr[], size)
    if(size > 2)
        for(i=1 ; i<size-1 ; i++)
            if(arr[i-1] >= arr[i] && arr[i] <= arr[i+1])
                print(arr[i])
    print(newline)
```

$\Theta(1)$ for the loop body; $\Theta(N)$

$T_B(N) = T_W(N) = \Theta(N)$

Part 2.3

```
Part2_3(arr[], size, b)
    for(i=0; i<size; i++)
        for(j=i; j<size; j++)
Θ(1)  { if(arr[i]+arr[j]==b)
            return true
    return false
```

Θ(size − i)

$$\sum_{i=0}^{size} size - i = size + size-1 + \cdots + 0 = \frac{size \times (size+1)}{2}$$

$T(N) = O(N^2)$

$T_{Best}(N) = \Theta(1)$

Part 2.4

```
Part2_4(arr[], size)
    for(i=1; i<size; i++)
        if(!(part2_3(arr, i, arr[i])))   } Θ(N²)
            return false                 } Θ(1)
    return true
```

Θ(N²)

$$\sum_{i=1}^{size} i^2 = 1+4+9+\cdots+size^2 = \frac{size(size+1)(2\,size+1)}{2}$$

$T(N) = O(N^3)$

$T_{Best}(N) = \Theta(1)$