Time Complexity Practice Questions

😵‍💫 ✖️
😁 ✅

1. Summing Elements in an Array

```
int sum = 0;
for (int i = 0; i < n; i++) {
    sum = sum + i;
}
```

Time Complexity: $O(n)$

2. Matrix Addition

```
for (int i = 0; i < n; i++) {
    for (int j = 0; j < n; j++) {
        c[i][j] = a[i][j] + b[i][j];
    }
}
```

Time Complexity: $O(n^2)$

3. Normal Loops

a. <u>Simple Loop</u>

```
for (int i = 0; i < n; i++) {
    stmt();
}
```

b. Decrementing Loop

```
for (int i = n; i > 0; i--) {
    stmt();
}
```

Time Complexity: O(n)

4. Increment by Two

```
for (int i = 0; i < n; i += 2) {
    stmt();
}
```

Time Complexity: O(n)

5. Nested For Loops

```
for (int i = 0; i < n; i++) {
    for (int j = 0; j < n; j++) {
        stmt();
    }
}
```

Time Complexity:O(n^2)

6. Dependent For Loops

What happens if the inner loop is dependent on the outer loop?

```
for (int i = 1; i < n; i*=2) {
    for (int j = 0; j < i; j++) {
         for(int k=0 ; k<j ; k++){
         }
        stmt();
    }
}
```

7. Non-Standard Outer Loop Execution

```
int p = 0;
for (int i = 1; p <= n; i++) {
    p = p + i;
    stmt();
}
```

Time Complexity: $O(n^{1/2})$

8. Multiply i Value

```
for (int i = 1; i < n; i = i * 2) {
    stmt();
}
```

Time Complexity: O(log n)

9. Divide i Value

```
for (int i = n; i > 0; i = i / 2) {
    stmt();
}
```

Time Complexity: O(log n)

Practice Problems

1)

```
bool List::Equalize_Occurrences(char d, int maxcount)
{
   Node* ptr = first;
   bool chk=false;
   Node* temp;
   while (ptr != NULL)
   {
      if (ptr->data == d)
      {
         chk = true;
         int count = 0;
         temp = ptr;
         while (ptr != NULL)
         {
            if (ptr->data == d)
            {
               count++;
               ptr = ptr->next;
            }
            else
               break;
         }
         if (count > maxcount)
         {
            while (count > maxcount)
            {
               del_after(temp);
               count--;
            }
         }
         else if (maxcount > count)
         {
            while (count < maxcount)
            {
               ins_after(temp,d);
               count++;
            }
         } // else
      } // outer if
      ptr = ptr->next;
   } // while
   return chk;
}
```

2)

```
for(int i=2 ; i<n ; i=i*i){
    ;
}
```

3)

```
int m = (int)((15 + Math.round(3.2 / 2)) * (Math.floor(10 / 5.5) / 2.5) *
Math.pow(2,5));
for (int i = 0; i < m; i++) {
    cout<<"hello";
}
```

4)

```
for (int i = 1; i <= N  * N; i *= 2)
    {
        for (int j = 0; j < i; j++) {
            cout<<"hello";
    }
}
```

5)

```
for(int i=1 ; i<n ; i*=2){
    for(int j=0 ; j<i ; j++){
        x = 0;
    }
}
```

6)

```
for(int i=1;i<=n;i++){
    for(int j=2;j<=n;j=j*j*j){
        cout<<i<<j<<endl;
    }
}
```

7)

```
for (i=n/2; i<=n; i++)
	for (j=1; j+n/2<=n; j++)
		for (k=1; k<=n; k = k * 2){
			cout<<”hello “;
			c++;
}
```

8)

```
s=1;
While(s<=n)
{
	for(int i=1; i<=s; i++)
		cout<<” hello”;
	s*=2;
}
```

9)

```
for (j=1; j<=n; j++)
	for (k=1; k<=j*3; k ++)
	{
		cout<<”hello “;
		}
```

10)

```
for(int i=n/2;i<=n;i++){
	for(int j=2;j<=n;j=j*j){
		cout<<i<<j<<endl;
	}
}
```

11)

```
for (j=1; j<=n; j*=2)
	for (k=n; k>=1; k --) {
		for (i=1; i<=n; i*=3)
			cout<<”hello “;
	}
```

12)

```
For ( i = 1 to n)
    For ( j = 1 to i * i)
        If (j mod i == 1)
            Cout << i
```

13)

Algorithm

Initialize count as 0
Sort all numbers in increasing order using quicksort
Remove duplicates from the array.
Let D be the new array
Do the following for each element A[i], where i varies from 1 to D
    -> Binary search for A[i] + K in subarray from i+1 to D
    -> if A[i] + K found, increment count
Return count

14)

```
Int key, j
For (int i = 1; i < size; i++)
    Key = array[i]
    J = i
    While (j > 0 && array[j-1] > key)
        Array[j] = array[j-1]
        J - -
    Array[j] = key
```

15)

```
For (int i = 0; i < n; i++)
    For (int j = 1; j <= n*n; j++)
        If (j % 2 == 0)
            For (int k = 0; k < n; k++)
                Cout << “*”;
```

16)

```
Void fun(int n, int k)
        For (int i = 1; i <=n; i++)
                Int p = pow(i, k)
                For (int j = 1; j <= p; j++)
                        Stmt
```

17)

```
For (int i = 1; i<= n; i++)
        For (int j = 1; j < i*i; j*=2)
                Stmt
```

18)

```
int p = 0;
for (int i = 1; p <= n; i++) {
    p = p + i;
    stmt();
}
```

Solution:

1) O(N)

2) O(log(log n))

3) O(1)

4)

Outer loop executes $\log_2 n^2 = 2 \log_2 n$ times.
Inner loop then executes $1+2+4+8+16+....+(2^{\log_2 n})^2 = n^2$ times.
Time complexity = O(n^2)
cout runtime = n^2

5)

Outer loop executes $\log_2 n$ times.
Inner loop executes $1+2+4+8+16+...+2^{\log_2 n} = n$ times.
Time complexity = O(n)
cout runtime = n times

6)

Outer loop executes n times.
Inner loop then executes $2+2^3+2^{3^3}+...+2^{3^k}$ times.
$2^{3^k} = n \rightarrow 3^k = \log_2 n \rightarrow k = \log_3 \log_2 n$
Time complexity = O(n log log n)
cout runtime = n log log n .

7)

First loop executes n/2 times.
Second loop executes $n^2/4$ times.
Third loop executes $n^2 \log_2 n / 4$ times.
Time Complexity = $O(n^2 \log n)$
cout runtime = $n^2 \log n$

8)

Outer loop executes $\log_2 n$ times.
Inner loop executes
$1+2+4+8+16+\ldots+2^{\log_2 n} = n$ times.
Time complexity = O(n)
cout runtime = n times

9)

Outer loop executes n times.
Inner loop executes 3+6+9+…+3n times = 3n(n+1)/2 times.
Time complexity = $O(n^2)$
cout runtime =3n(n+1)/2

10)

Outer loop executes n/2 times.
Inner loop then executes $2+2^2+2^4+2^8+\ldots.+2^{2^k}$ times.
$2^{2^k} = n \rightarrow 2^k = \log_2 n \rightarrow k = \log_2\log_2 n$
Time complexity = O(n log log n)
Cout runtime = n log log n .

11)

First loop executes $\log_2 n$ times.
Second loop executes n $\log_2 n$ times.
Third loop executes n $\log_2 n * \log_3 n$ times.
Time Complexity = $O(n (\log n)^2)$
Cout runtime =n $\log_2 n * \log_3 n$

12) $O(n^3)$
13) $O(n \log n + n + n \log n) = O(n \log n)$
14) $O(n^2)$
15) $O(n^4)$
16) $O(n^{k+1})$

17)
Outer = O(N)
Inner = O(lg N)
Total = O(N lg N)
18) $O(n^{1/2})$