Time complexity :-

Date : / /

①
```
int a=0, b=0;
for (i=0; i<N; i++) {
    a= a+rand();
}
for (j=0; j<m; j++) {
    b = b+rand();
}
```

→ $O(N+m)$

②
```
int a=0;
for (i=0; i<N; i++) {
    for (j=N; j>i; j--) {
        a = a+i+j;
    }
}
```

→ $O(n^2)$

③
```
int i, j, k=0;
for (i=n/2; i<=n; i++) {
    for (j=2; j<=n; j=j*2) {
        k= k+n/2;
    }
}
```

→ $O(n \log n)$

AKATRIKAS

④
```
int a=0, i=N;
while (i>0) {
    a+=i;
    i/=2;
}
```

→ O(log n)

⑤
```
for (var i=0; i<n; i++)
    i*=k
```

→ $O(\sqrt{N})$

⑥
```
def fun(n);
    if (n<5);
        print ("Geeks for Geeks", end="")
    else:
        for ii in range(n):
            print (i. end="")
```

→

* best case O(1)
* worst case O(n)

MATRIKAS

(7)
```
def fun(a, b):
    while (a != b):
        if (a > b):
            a = a - b
        else:
            b = b - a
```

→ $O(n)$

(8)
```
void fun(int n)
{
    for (int i = 0; i*i < n; i++)
        cout << "GeeksforGeeks";
}
```

→ $O(\sqrt{n})$

(9)
```
void fun(int n, int x)
{
    for (int i = 1; i < n; i = i * x)
        // or for (int i = n; i >= 1; i = i / x)
        cout << "GeeksforGeeks";
}
```

→ $O(\log n)$

MATRIKAS

(10)
```
void fun (int n)
{
  for (int i = 0; i < n/2; i++)
    for (int j = 1; j + n/2 <= n; j++)
      for (int k = 1; k <= n; k = k * 2)
        cout << "GeeksforGeeks";
}
```

→ $O(n^2 \log n)$

(11)
```
Void fun (int n)
{
  int i = 1;
  while (i < n) {
    int j = n;
    while (j > 0) {
      j = j / 2;
    }
    i = i * 2;
  }
}
```

→ $O(\log^2 n)$