# Marks Allocation:

Best Case: T(n) = O(n) ---------------------------------------- 0.5 mark

Worst Case: Correct T(n) equation ---------------------------------------- 0.5 mark

T(n) = O(n2) ---------------------------------------- 0.5 mark

One line explanation ---------------------------------------- 0.5 mark

void Func(int n)

{

int\* A=new int[n+1]; -------------------------------------- 0.25 mark

for (int i=0; i < n; i++) ----------------------------------- 0.5 x 3 mark

A[i]=i; -------------------------------------- 0.25 mark

for (int i=1; i <= n; i=i\*2) ----------------------------------- 0.5 x 3 mark

A[i]=1; ---------------------------------------- 0.5 mark

Int j=n; ---------------------------------------- 0.5 mark

while(j>0) ---------------------------------------- 0.5 mark

{

if(A[j]==1) ---------------------------------------- 0.5 mark

{

for (int k=n; k >=0; k=k/2) ------------------------------------ 0.5 x 3 mark

A[k]++; ---------------------------------------- 0.5 mark

}

j--; ---------------------------------------- 0.5 mark

}

}

# Solution:

|  |  |
| --- | --- |
| **Statement** | **Number of times executed** |
| int\* A=new int[n+1] | 1 |
| i=0 | 1 |
| i<n | n + 1 |
| i++ | n |
| A[i]=i | n |
| i=1 | 1 |
| i <= n | Log2n + 1 |
| i=i\*2 | Log2n |
| A[i]=1 | Log2n |
| int j=n | 1 |
| while(j>=0) | n + 1 |
| if(A[j]==1) | n |
| int k=n | Log2n |
| k >=0 | Log2n\* Log2n + Log2n |
| K=k/2 | Log2n\* Log2n |
| A[k]++; | Log2n\* Log2n |
| j-- | n |
| **Total** | **1 + 1 + n + 1 + n + n + 1 + Log2n + 1 + Log2n + Log2n + 1 + n + 1 + n + Log2n + Log2n\* Log2n + Log2n + Log2n\* Log2n + Log2n\* Log2n +n**  **T(n) = 7 + 6n + 4Log2n + 3(Log2n)2**  **T(n) = O(n)** |