2.d. Finding Complexity using Counter Method

Aim: Convert the following algorithm into a program and find its timecomplexity using counter method.

```
void function(int n)
{
    int c= 0;
    for(int i=n/2; i<n; i++)
        for(int j=1; j<n; j = 2 * j)
            for(int k=1; k<n; k = k * 2)
                c++;
}
```

Note: No need of counter increment for declarations and scanf() and count variable printf() statements.

Input:
A positive Integer n
Output:
Print the value of the counter variable

Algorithm:

function(n) {

initialize count to 0

initialize c to 0

increment count by 1

// outer loop: i goes from n/2 to n-1

for each i from n/2 to n-1 {

increment count by 1

// middle loop: j starts at 1 and doubles each iteration until j < n

for each j starting from 1 and doubling each time (j = 2 * j) until j < n {

increment count by 1

```
        // inner loop: k starts at 1 and doubles each iteration until k < n
        for each k starting from 1 and doubling each time (k = k * 2) until k < n {
            increment count by 1
            increment c by 1
            increment count by 1
        }

        increment count by 1 // after inner loop ends
    }

    increment count by 1 // after middle loop ends
}

    increment count by 1 // after outer loop ends

    print count
}
```

Program:

```
#include<stdio.h>
void function(int n)
{
    int count=0;
    int c= 0;
    count++;
    for(int i=n/2; i<n; i++){
        count++;
```

```
        for(int j=1; j<n; j = 2 * j){
            count++;
            for(int k=1; k<n; k = k * 2){
                count++;
                c++;
                count++;
            }
            count++;
        }
        count++;
    }
    count++;
    printf("%d",count);
}
int main(){
    int n;
    scanf("%d",&n);
    function(n);
}
```

Output:

	Input	Expected	Got	
✔	4	30	30	✔
✔	10	212	212	✔