

**Ex. No: 2**

**Date: 20.08.24**

**Register No.: 230701386**

**Name: R YASHVINTHINI**

## **Finding Time Complexity of Algorithms**

### **2.a. Finding Complexity using Counter Method**

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

Void function (int n)

```
{  
    int i= 1;  
    int s =1;  
    while(s <= n)  
    {  
        i++;  
        s += i;  
    }  
}
```

**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:**

1. Initialize count to 0.
2. Initialize i to 1 and increment count by 1.
3. Initialize s to 1 and increment count by 1.
4. While s is less than or equal to n:
5. Increment count by 1.

6. Increment i by 1 and increment count by 1.
7. Add i to s and increment count by 1.
8. Increment count by 1 after the loop ends.
9. Print the value of count

**Program:**

```
#include <stdio.h>

void function(int n)
{
    int count=0;
    int i=1;
    count++;
    int s=1;
    count++;
    while(s<=n)
    {
        count++;
        i++;
        count++;
        s+=i;
        count++;
    }
    count++;
    printf("%d",count);
}

int main()
{
    int n;
```

```
scanf("%d",&n);  
function(n);  
}
```

**Output:**

	Input	Expected	Got	
✓	9	12	12	✓
✓	4	9	9	✓

Passed all tests! ✓

## 2.b. Finding Complexity using Counter Method

Aim:

Convert the following algorithm into a program and find its time complexity using the counter method.

```
void func(int n)
{
    if(n==1)
    {
        printf("*");
    }
    else
    {
        for(int i=1; i<=n; i++)
        {
            for(int j=1; j<=n; j++)
            {
                printf("*");
                printf("*");
                break;
            }
        }
    }
}
```

**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:**

1. Initialize count to 0.
2. If n is equal to 1:
3. Increment count by 1.
4. Print \*.
5. Else:
6. Increment count by 1.
7. For each i from 1 to n:
  - a. Increment count by 1.
  - b. For each j from 1 to n:
    - i. Increment count by 1.
    - ii. Increment count by 1 (for printf("\*")).
    - iii. Increment count by 1 (for printf("\*")).
    - iv. Increment count by 1 (for break statement).
  - c. Increment count by 1 (after inner loop ends).
8. Increment count by 1 (after outer loop ends).

**Program:**

```
#include void func(int n)

{
    int count=0;
    if(n==1)
    {
        count++;
        printf("*");
    }
    else
```

```

    {
        count++;
        for(int i=1; i<=n; i++)
        {
            count++;
            for(int j=1; j<=n; j++)
            {
                count++;
                //printf("*");
                count++;
                //printf("*");
                count++;
                break;
            }
            count++;
        }
        count++;
    }
    printf("%d",count);
}

int main()
{
    int n;
    scanf("%d",&n);
    func(n);
}

```

**Output:**

	Input	Expected	Got	
✓	2	12	12	✓
✓	1000	5002	5002	✓
✓	143	717	717	✓