

02 - Finding Time Complexity of Algorithms

Ex. No. : 2.1

Date: 20.08.24

Register No.: 230701336

Name: SRI AKASH U G

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

For example:

Input	RESULT
9	12

ALGORITHM:

Step 1: Start

Step 2: Read the value of n from the user.

Step 3: Call the function func(n), initializing count, i to 1, and s to 1. Increment count (1st increment).

Step 4: While $s \leq n$, increment count (2nd increment), then increment i and update s by adding i. Increment count again (3rd increment).

Step 5: After exiting the loop, increment count (4th increment).

Step 6: Print the value of count.

Step 7: End

PROGRAM:

```
#include<stdio.h> void
```

```
function(int n)
```

```
{
```

```
    int count=0;
```

```
    int i=1;
```

```
    count++;
```

```
    int s=1;
```

```
    count++;
```

```
    while(s<=n)
```

```
{
```

```
        i++; count++;  
        s+=i;  
        count++;  
        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! ✓

Correct

Marks for this submission: 1.00/1.00.

RESULT:

Hence the above program has been executed successfully.

Ex. No. : 2.2

Date: 20.08.24

Register No.: 230701336

Name: SRI AKASH U G

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:

Step 1: Start

Step 2: Read the value of n from the user.

Step 3: Call the function func(n).

Step 4: In func, if n == 1, increment count (1st increment).

Step 5: If n > 1, increment count (2nd increment) and loop i from 1 to n, increment count (3rd increment) for each iteration, and loop j from 1 to n, incrementing count (4th increment) three times, then break. Increment count (5th increment) after the inner loop, and once more after the outer loop (6th increment).

Step 6: Print the value of count.

Step 7: End

PROGRAM:

```
#include<stdio.h> void
```

```
func(int n)
```

```
{  
    int count=0; if(n==1)  
    {  
        count++;  
    }  
    else  
    {  
        count++; for(int i=1;i<=n;i++)  
        {  
            count++; for(int j=1;j<=n;j++)  
            {  
count++; count++; count++;  
                break;  
            }  
            count++;  
        }  
        count++;  
    }  
    printf("%d",count);  
}
```

```
}
```

```
int main()
```

```
{
```

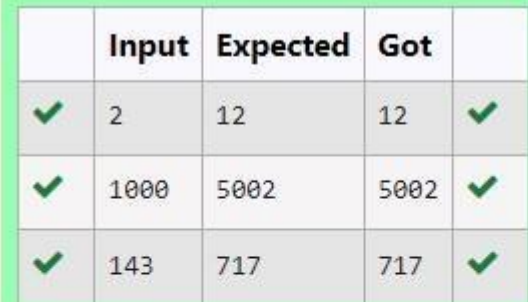
```
    int n;
```

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

```
    func(n);
```

```
}
```

OUTPUT:



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

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

RESULT:

Hence the above program has been executed successfully.

Ex. No. : 2.3

Date: 20.08.24

Register No.: 230701336

Name: SRI AKASH U G

AIM:

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

```
Factor(num) {  
    {  
        for (i = 1; i <= num;++i)  
        {  
            if (num % i== 0)  
            { printf("%d ", i);  
            }  
        }  
    }  
}
```

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

Input:

A positive Integer n Output:

Print the value of the counter variable ALGORITHM:

Step 1: Start

Step 2: Read the value of n from the user.

Step 3: Call the function Factor(n).

Step 4: In Factor, loop i from 1 to num, increment count (1st increment).

Step 5: For each i, check if $\text{num} \% i == 0$. If true, increment count (2nd increment). Increment count again (3rd increment) for the end of the loop.

Step 6: After the loop, increment count (4th increment).

Step 7: Print the value of count.

Step 8: End

PROGRAM:

```
#include<stdio.h> void
Factor(int num)
{
    int i,count=0; for(i=1;i<=num;i++)
    {
        count++; if(num%i==0)
        {
            count++;
        }
        count++;
    }
    count++; printf("%d",count);
}
int main()
{
    int num; scanf("%d",&num);
    Factor(num);
}
```

OUTPUT:

	Input	Expected	Got	
✓	12	31	31	✓
✓	25	54	54	✓
✓	4	12	12	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

RESULT:

Hence the above program has been executed successfully.

Ex. No. : 2.4

Date: 20.08.24

Register No.: 230701336

Name: SRI AKASH U G

AIM:

Convert the following algorithm into a program and find its time complexity 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:

Step 1: Start

Step 2: Read the value of n from the user.

Step 3: Call the function function(n).

Step 4: In function, initialize c to 0 and increment count (1st increment).

Step 5: Loop i from n/2 to n, incrementing count (2nd increment), and for each i, loop j from 1 to n, doubling j each time, incrementing count (3rd increment).

Step 6: Inside the j loop, loop k from 1 to n, doubling k each time, incrementing count (4th increment), increment c, and increment

count (5th increment). Increment count again after the k loop (6th increment) and after the j loop (7th increment).

Step 7: Increment count after the i loop (8th increment).

Step 8: Print the value of count.

Step 9: End

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)  
  
            {
```

```
        c++; count++;  
        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	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

RESULT:

Hence the above program has been executed successfully.

Ex. No. : 2.5

Date: 20.08.24

Register No.: 230701336

Name: SRI AKASH U G

AIM:

Convert the following algorithm into a program and find its time complexity using counter method. void reverse(int n)

```
{ int rev = 0, remainder; while
    (n != 0)
    {
        remainder = n % 10; rev =
        rev * 10 + remainder; n/=
        10;
    } print(rev);
```



```
}
```

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:

Step 1: Start

Step 2: Read the value of n from the user.

Step 3: Call the function reverse(n).

Step 4: In reverse, initialize rev to 0 and increment count (1st increment).

Step 5: While n is not 0, increment count (2nd increment), calculate remainder as $n \% 10$, and increment count (3rd increment). Update rev by multiplying it by 10 and adding remainder, then increment count (4th increment). Divide n by 10 and increment count (5th increment).

Step 6: After exiting the loop, increment count (6th increment) and again for the commented print statement (7th increment).

Step 7: Print the value of count.

Step 8: End

PROGRAM:

```
#include<stdio.h> void
reverse(int n)
{
    int count=0; int
    rev=0;
    count++;

    int remainder;
    count++; while(n!=0)
    {
        count++; remainder=n%10;
        count++;
        rev=rev*10+remainder;
        count++; n/=10; count++;
    }count++; printf("%d",count);
}

int main()
{
```

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

OUTPUT:

	Input	Expected	Got	
✓	12	11	11	✓
✓	1234	19	19	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

RESULT:

Hence the above program has been executed successfully.