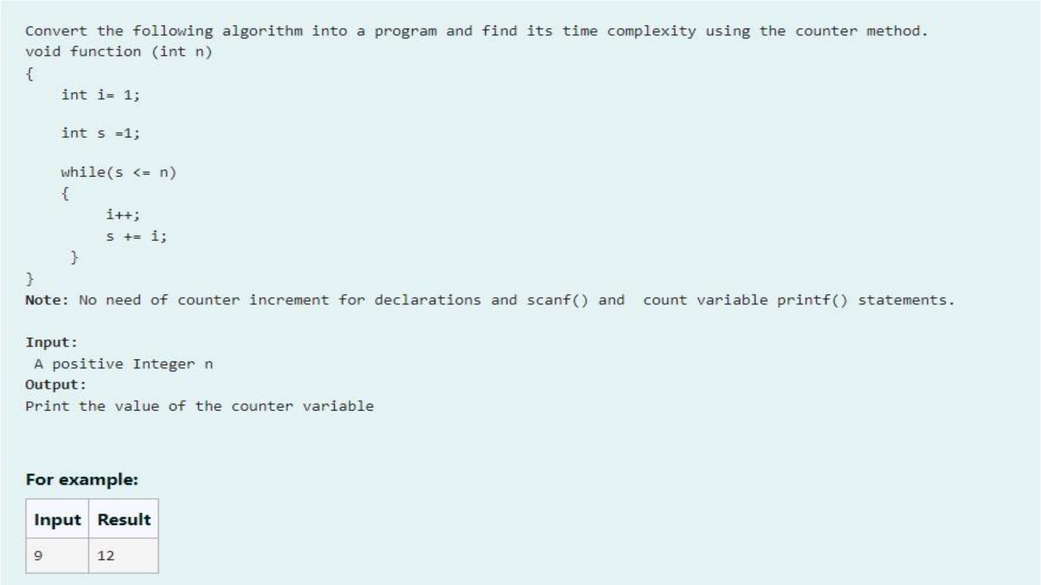
NAME: B.SREEMATHI
ROLL NO: 230701333
CLASS: CSE-F

TIME COMPLEXITY

QUESTION 2.A

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

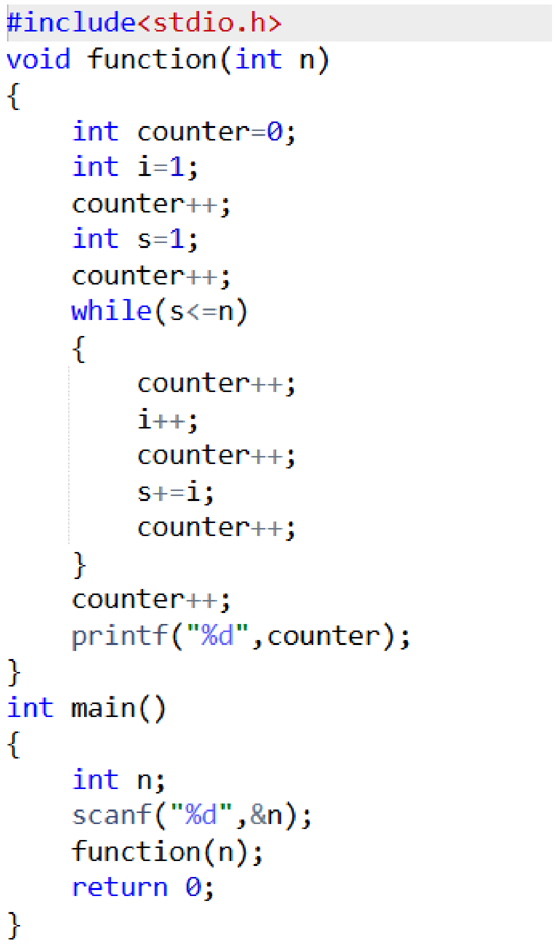
For example:

Input	Result
9	12

ALGORITHM:

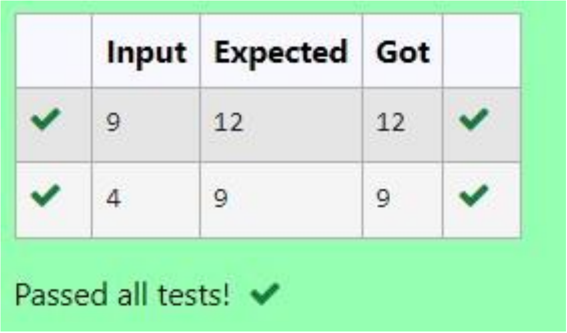
1. Start
2. Input n
3. Set counter = 0, i = 1, and s = 1
4. Increment counter
5. While s ≤ n:
6. Increment counter
7. Increment i
8. Increment counter
9. Add i to s (s += i)
10. Increment counter
11. Increment counter
12. Print counter
13. Stop

PROGRAM:



```c
#include<stdio.h>
void function(int n)
{
    int counter=0;
    int i=1;
    counter++;
    int s=1;
    counter++;
    while(s<=n)
    {
        counter++;
        i++;
        counter++;
        s+=i;
        counter++;
    }
    counter++;
    printf("%d",counter);
}
int main()
{
    int n;
    scanf("%d",&n);
    function(n);
    return 0;
}
```

OUTPUT :



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

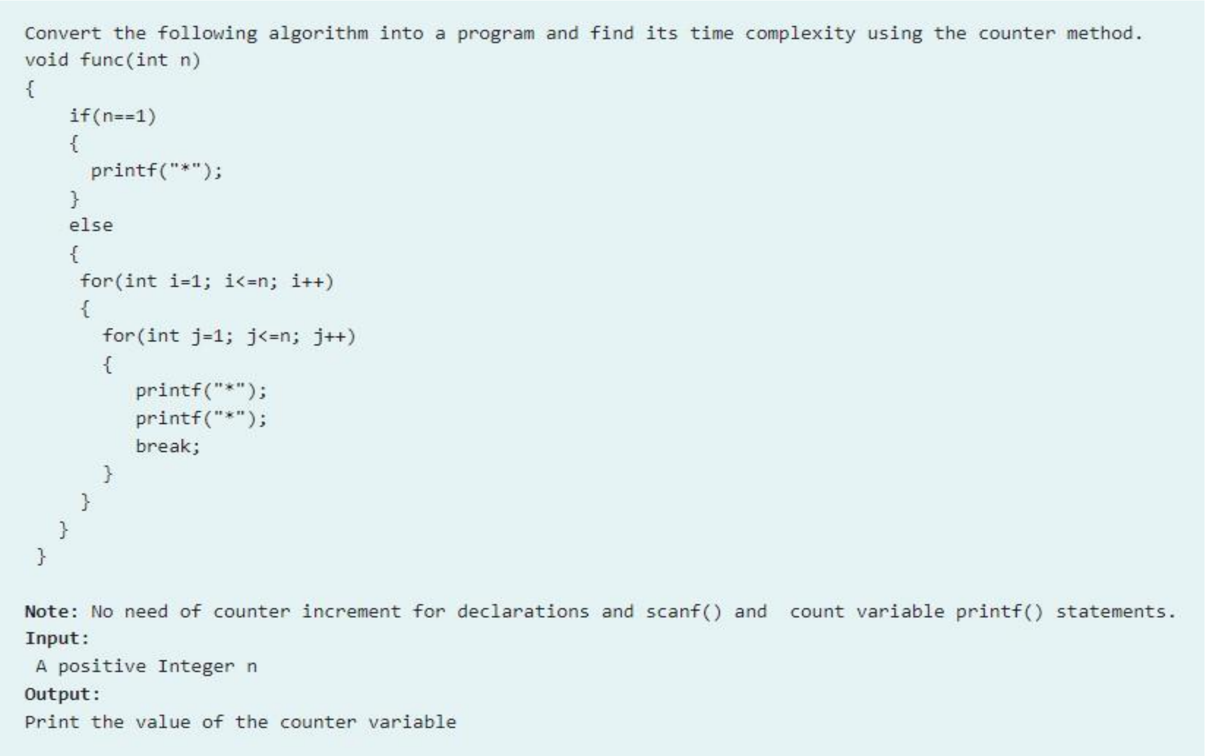
Passed all tests! ✓

RESULT:

The above code is executed successfully and gives expected output.

QUESTION 2.b

AIM:



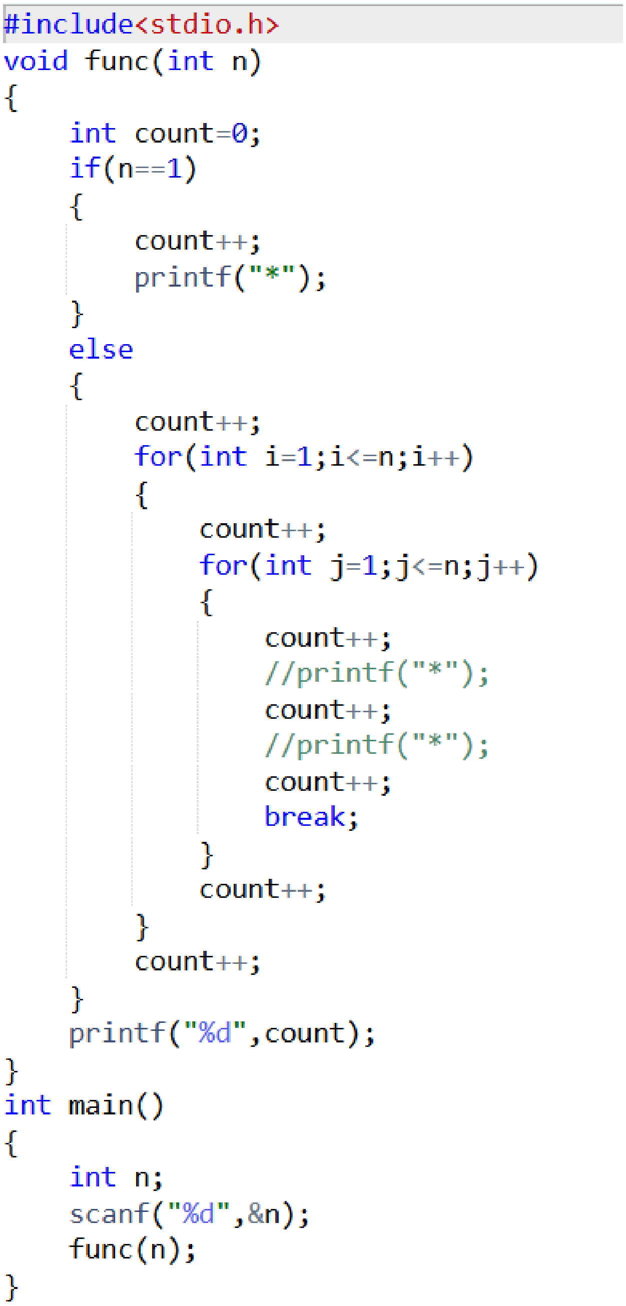
```
Convert the following algorithm into a program and find its time complexity using the counter method.
void func(int n)
{
    if(n==1)
    {
      printf("*");
    }
    else
    {
     for(int i=1; i<=n; i++)
     {
       for(int j=1; j<=n; j++)
       {
          printf("*");
          printf("*");
          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:

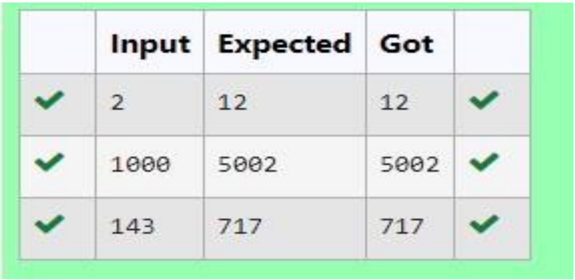
1. Start
2. Input n
3. Set count = 0
4. If n == 1:
5. Increment count
6. Print *
7. Else:
8. Increment count
9. For i = 1 to n:
    a. Increment count
    b. For j = 1 to n:
        i. Increment count three times
        ii. Break
    c. Increment count
10. Increment count
11. Print count
12. Stop

PROGRAM:



```
#include<stdio.h>
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:



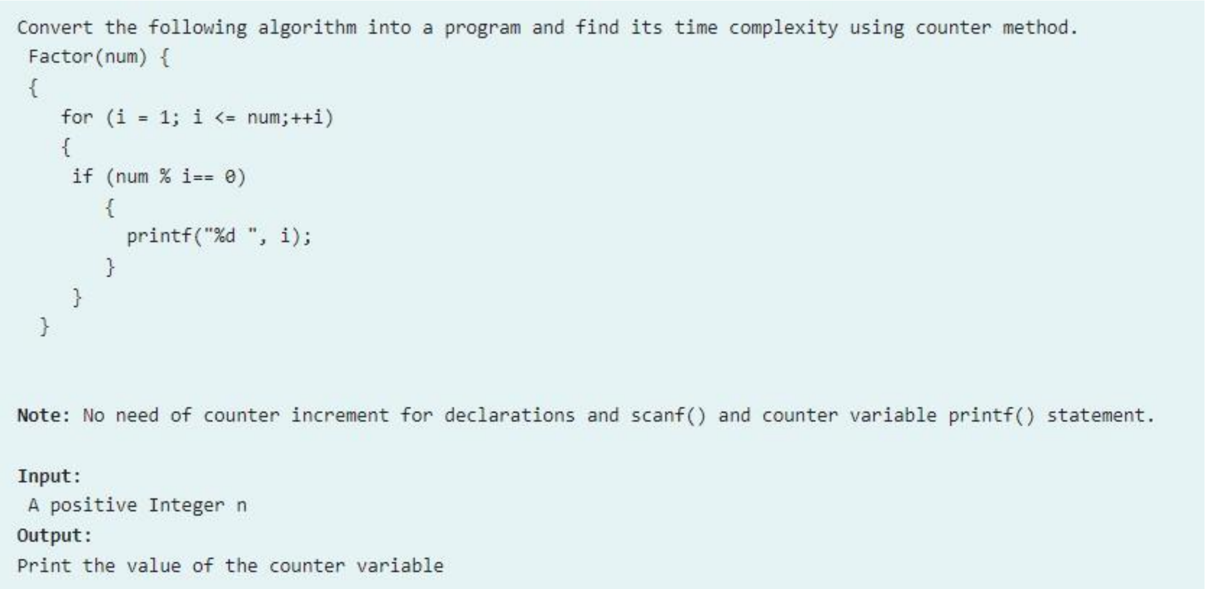
	Input	Expected	Got	
✓	2	12	12	✓
✓	1000	5002	5002	✓
✓	143	717	717	✓

RESULT:

The above code is executed successfully and gives expected output.

QUESTION 2.C

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);
        }
     }
  }
```

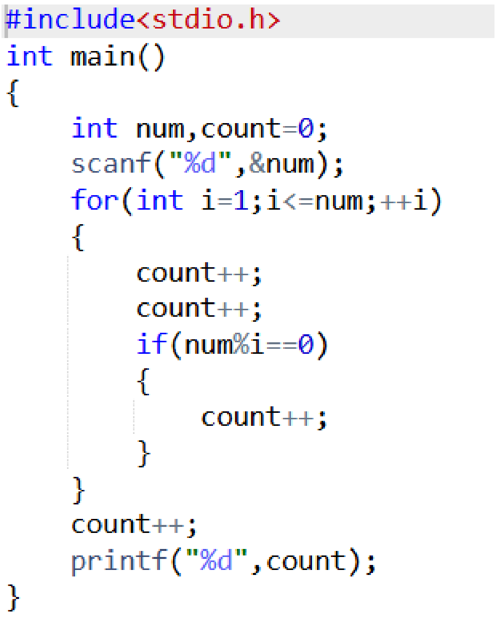
Note: No need of counter increment for declarations and scanf() and counter variable printf() statement.

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

ALGORITHM:

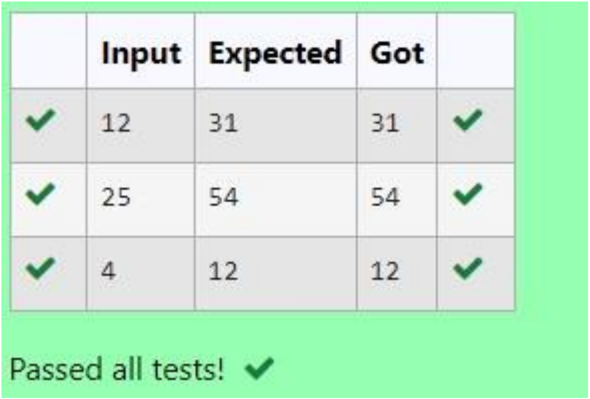
1. Start
2. Input num
3. Set count = 0
4. For i = 1 to num:
5. Increment count twice
6. If num % i == 0, increment count
7. Increment count
8. Print count
9. Stop**.**

PROGRAM:



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

OUTPUT:



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

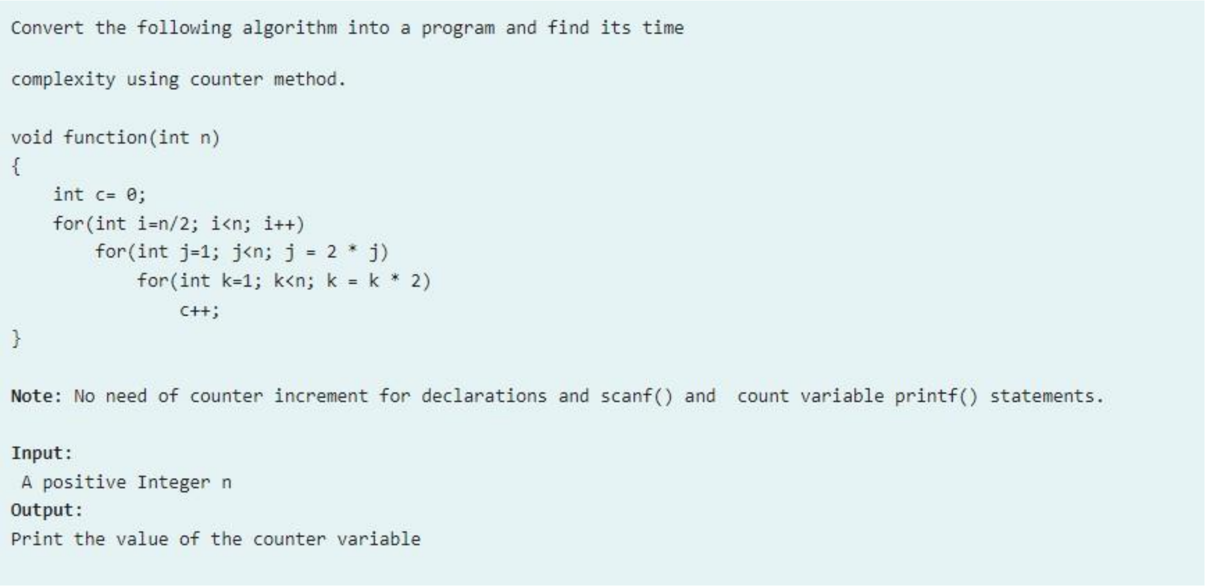
Passed all tests! ✓

RESULT:

The above code is executed successfully and gives expected output.

QUESTION 2.D

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.

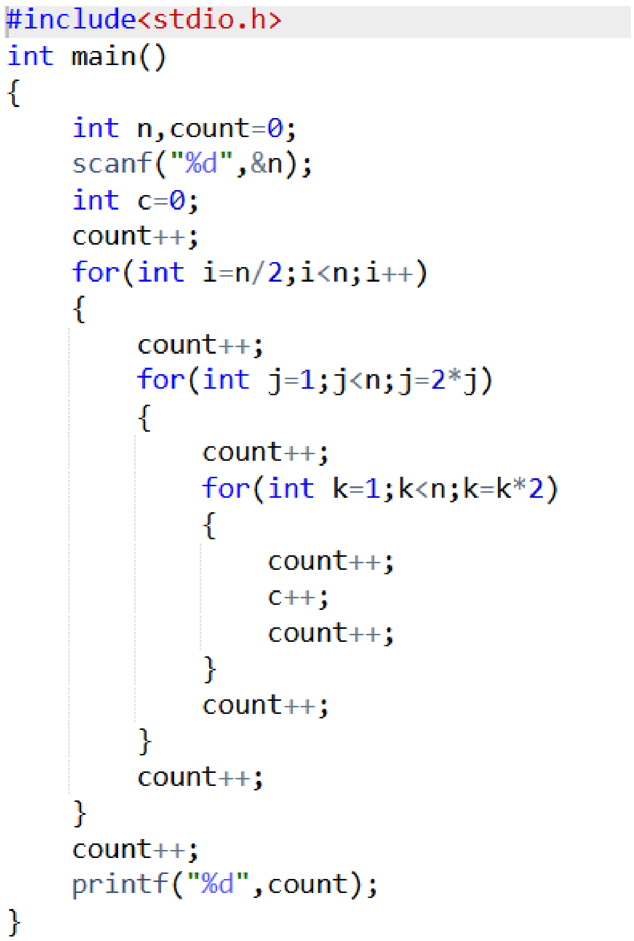
Input:
A positive Integer n

Output:
Print the value of the counter variable

ALGORITM:

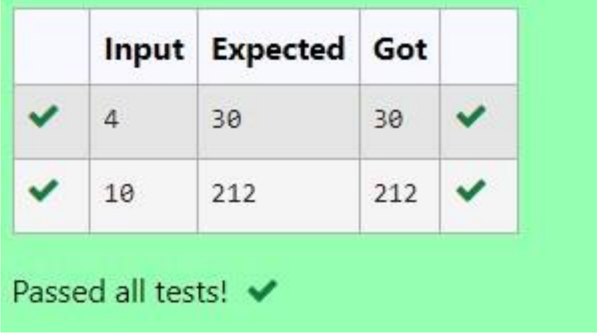
```
1. Start
2. Input n
3. Set count = 0 and c = 0
4. Increment count
5. For i = n/2 to n-1:
      a. Increment count
      b. For j = 1 to n, doubling j:
            i. Increment count
           ii. For k = 1 to n, doubling k:
                  1. Increment count
                  2. Increment c
                  3. Increment count
          iii. Increment count
      c. Increment count
6. Increment count
7. Print count
8. Stop
```

PROGRAM:



```c
#include<stdio.h>
int main()
{
    int n,count=0;
    scanf("%d",&n);
    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);
}
```

OUTPUT:



	Input	Expected	Got	
✓	4	30	30	✓
✓	10	212	212	✓

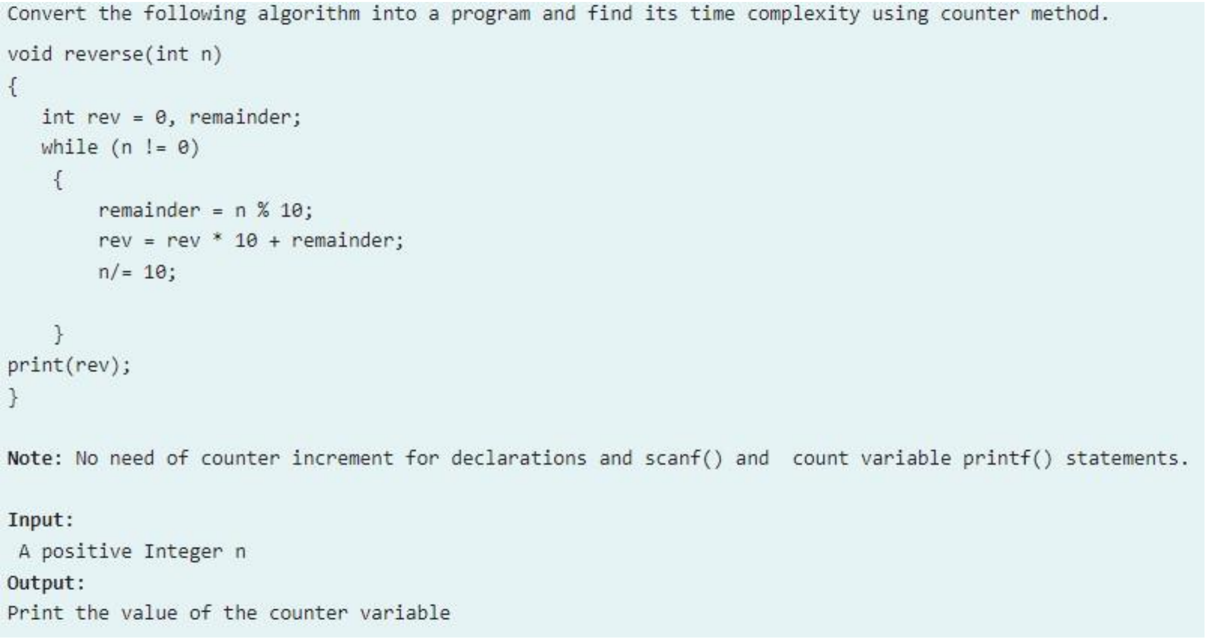
Passed all tests! ✓

RESULT:

The above code is executed successfully and gives expected output.

QUESTION 2.E

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);
}
```

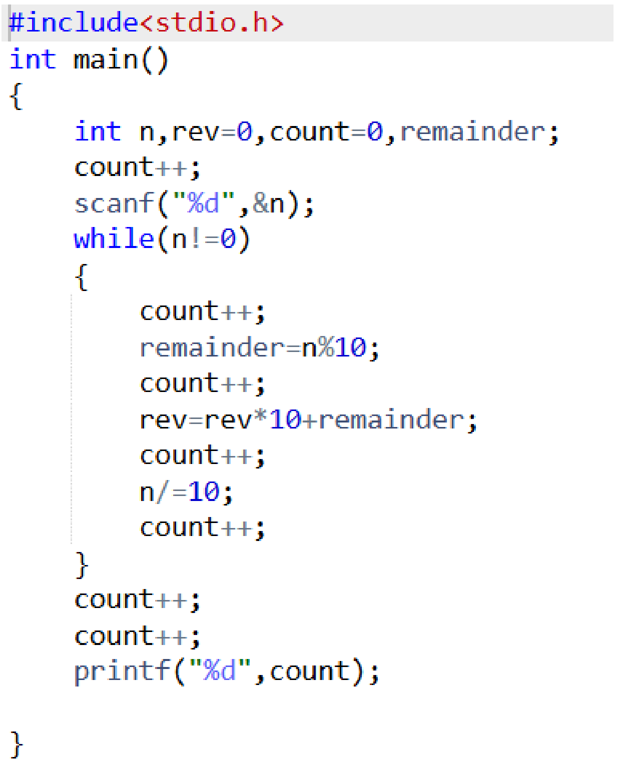
Note: No need of counter increment for declarations and scanf() and count variable printf() statements.

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

ALGORITHM:

1. Start
2. Input n
3. Set rev = 0, count = 0
4. Increment count
5. While n ≠ 0:
6. Increment count
7. Compute remainder = n % 10 and increment count
8. Update rev = rev * 10 + remainder and increment count
9. Update n = n / 10 and increment count
10. Increment count twice
11. Print count
12. Stop

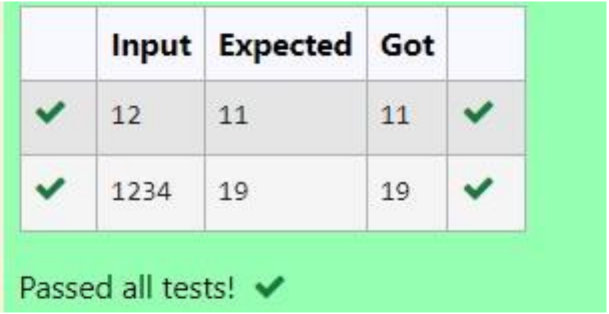
PROGRAM:



```
#include<stdio.h>
int main()
{
    int n,rev=0,count=0,remainder;
    count++;
    scanf("%d",&n);
    while(n!=0)
    {
        count++;
        remainder=n%10;
        count++;
        rev=rev*10+remainder;
        count++;
        n/=10;
        count++;
    }
    count++;
    count++;
    printf("%d",count);

}
```

OUTPUT:



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

Passed all tests! ✓

RESULT:

The above code is executed successfully and gives expected output.