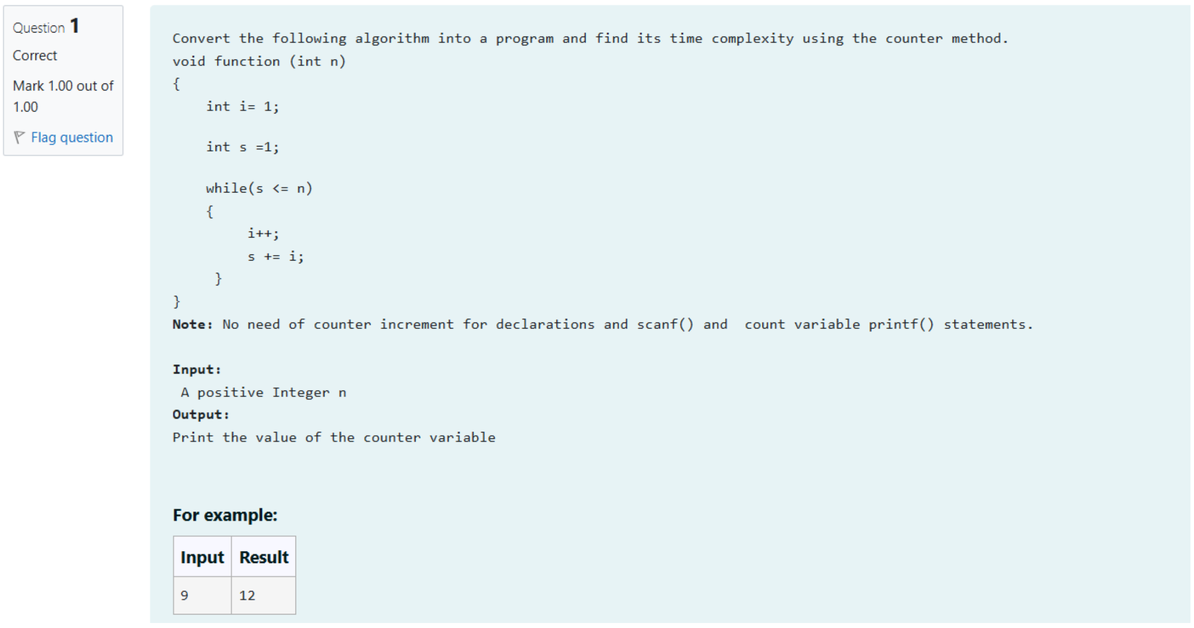
FINDING TIME COMPLEXITY OF ALGORITHMS



Question **1**

Correct

Mark 1.00 out of 1.00

Flag question

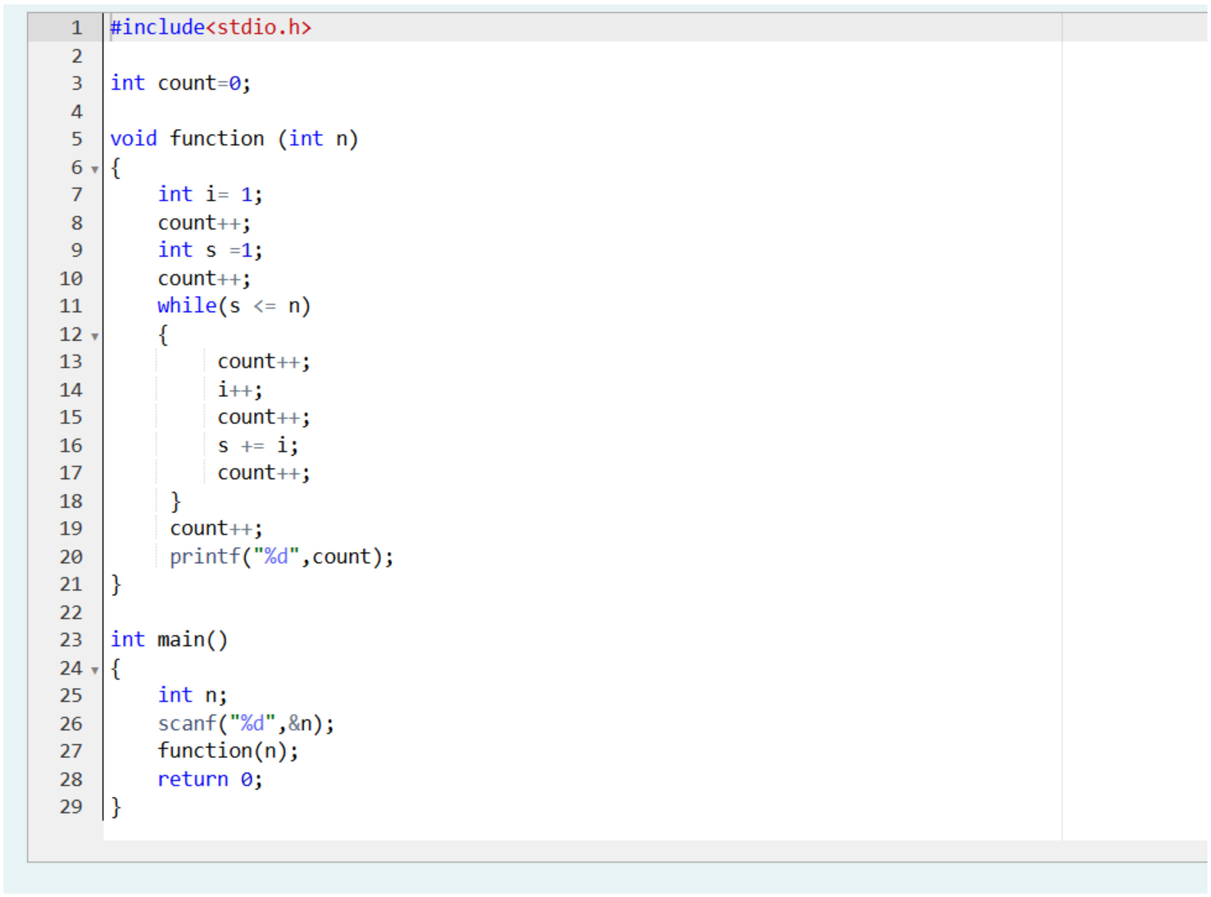
```
Convert the following algorithm into a program and find its time complexity using the counter method.
void function (int n)
{
    int i= 1;

    int s =1;

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

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

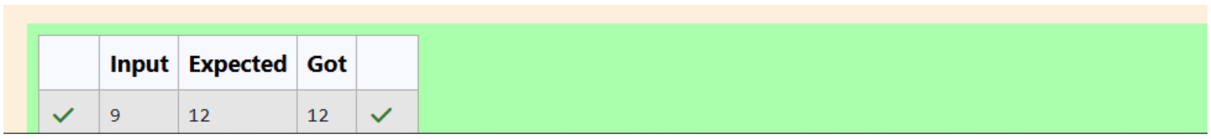
Input:
 A positive Integer n
Output:
Print the value of the counter variable

For example:

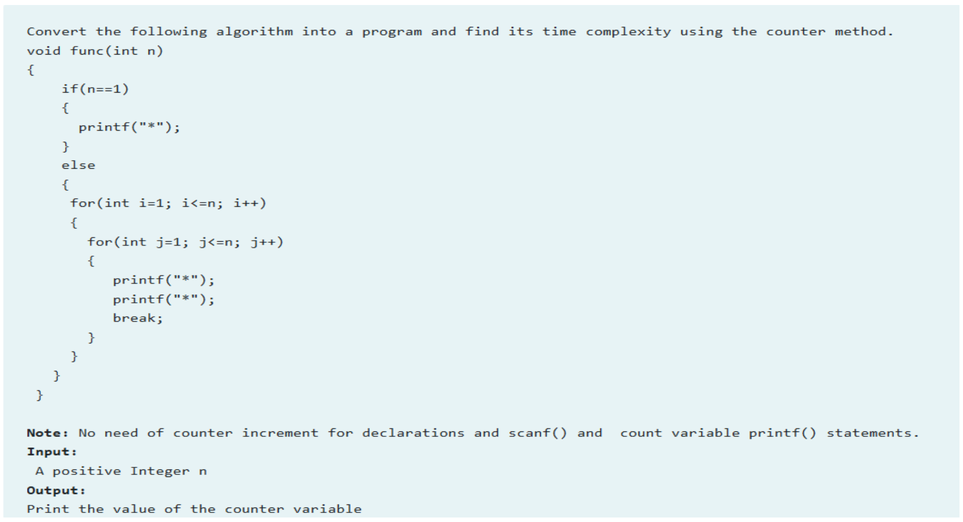
Input	Result
9	12



```c
#include<stdio.h>

int count=0;

void function (int n)
{
    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);
    return 0;
}
```

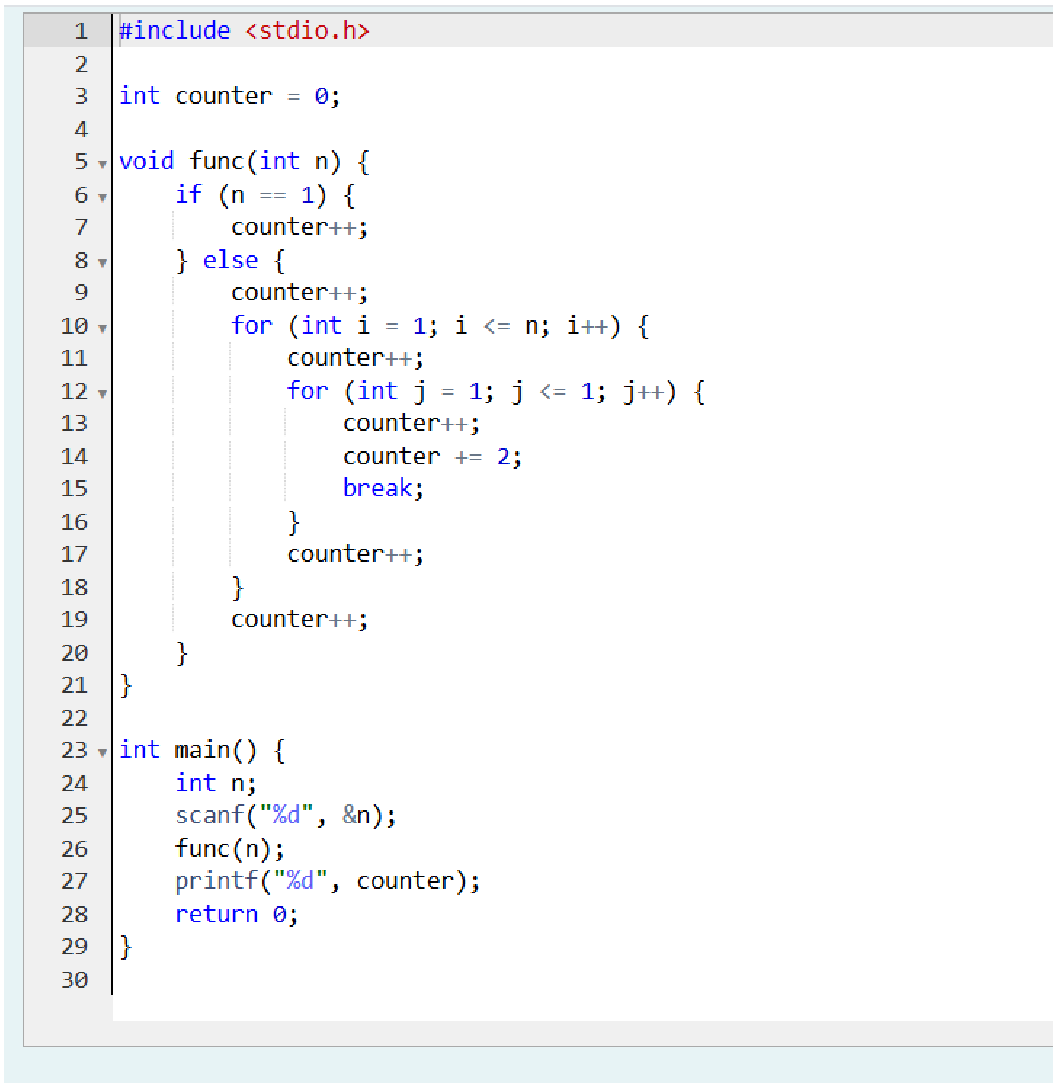


	Input	Expected	Got	
✓	9	12	12	✓

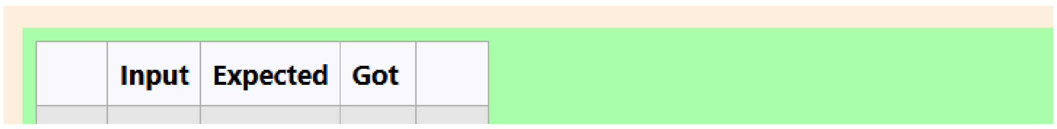


```
Convert the following algorithm into a program and find its time complexity using the counter method.
void func(int n)
{
    if(n==1)
    {
      printf("*");
    }
    else
    {
     for(int i=1; i<=n; i++)
     {
       for(int j=1; j<=n; j++)
       {
          printf("*");
          printf("*");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

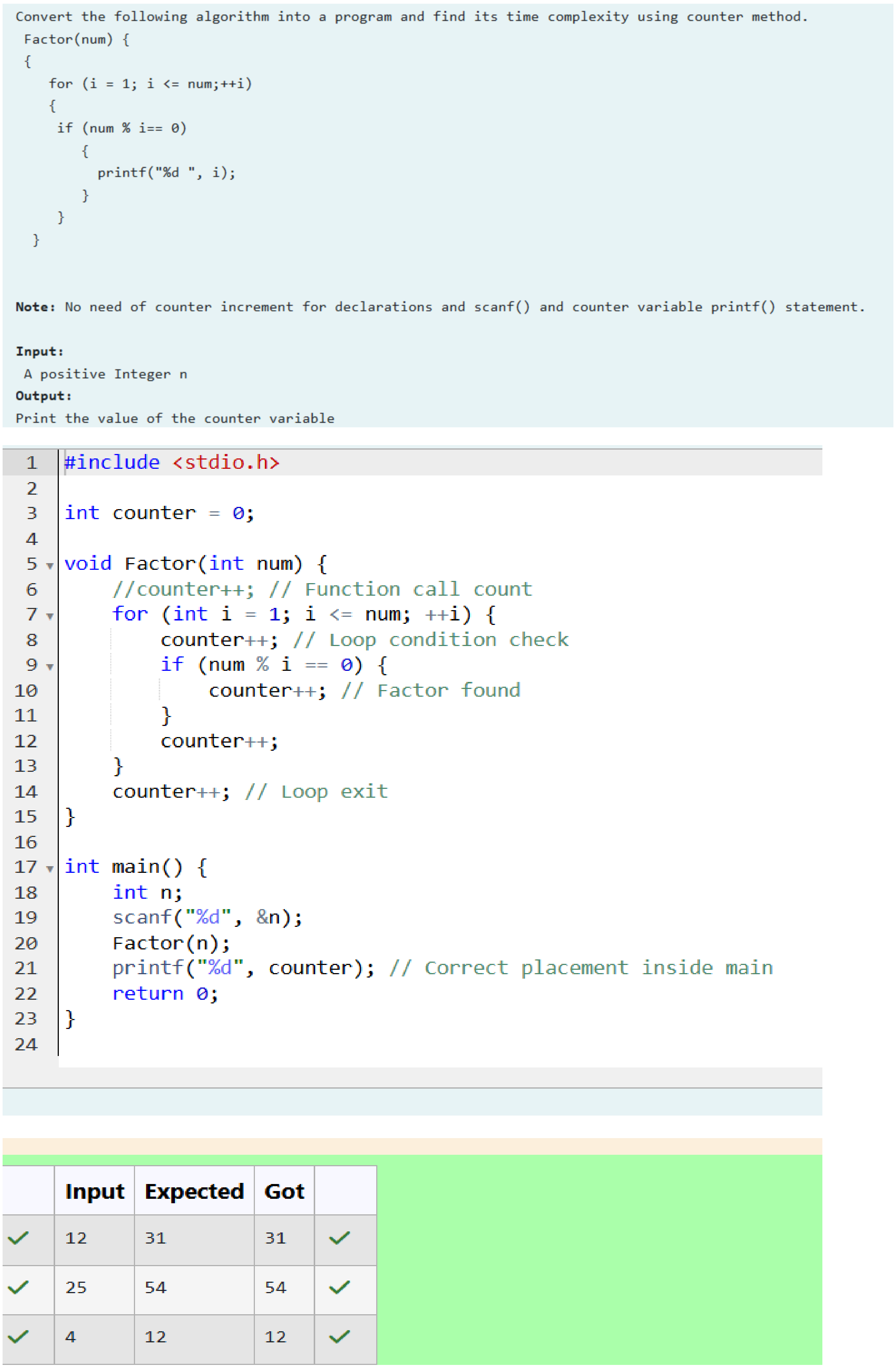
Note: No need of counter increment for declarations and scanf() and count variable printf() statements.
Input:
 A positive Integer n
Output:
Print the value of the counter variable



```c
#include <stdio.h>

int counter = 0;

void func(int n) {
    if (n == 1) {
        counter++;
    } else {
        counter++;
        for (int i = 1; i <= n; i++) {
            counter++;
            for (int j = 1; j <= 1; j++) {
                counter++;
                counter += 2;
                break;
            }
            counter++;
        }
        counter++;
    }
}

int main() {
    int n;
    scanf("%d", &n);
    func(n);
    printf("%d", counter);
    return 0;
}

```



	Input	Expected	Got	



```
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);
        }
     }
  }
```

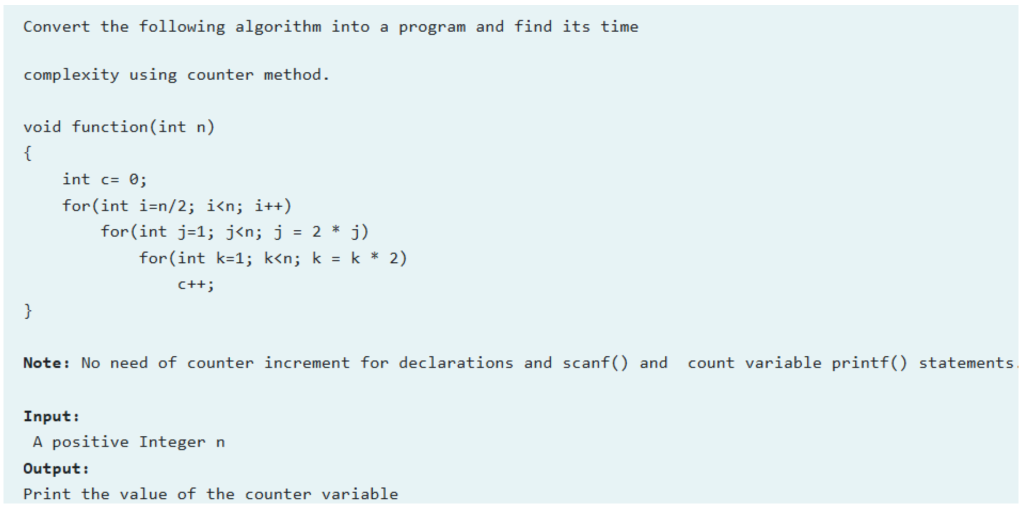
Note: No need of counter increment for declarations and scanf() and counter variable printf() statement.

Input:
 A positive Integer n

Output:
Print the value of the counter variable

```c
#include <stdio.h>

int counter = 0;

void Factor(int num) {
    //counter++; // Function call count
    for (int i = 1; i <= num; ++i) {
        counter++; // Loop condition check
        if (num % i == 0) {
            counter++; // Factor found
        }
        counter++;
    }
    counter++; // Loop exit
}

int main() {
    int n;
    scanf("%d", &n);
    Factor(n);
    printf("%d", counter); // Correct placement inside main
    return 0;
}

```

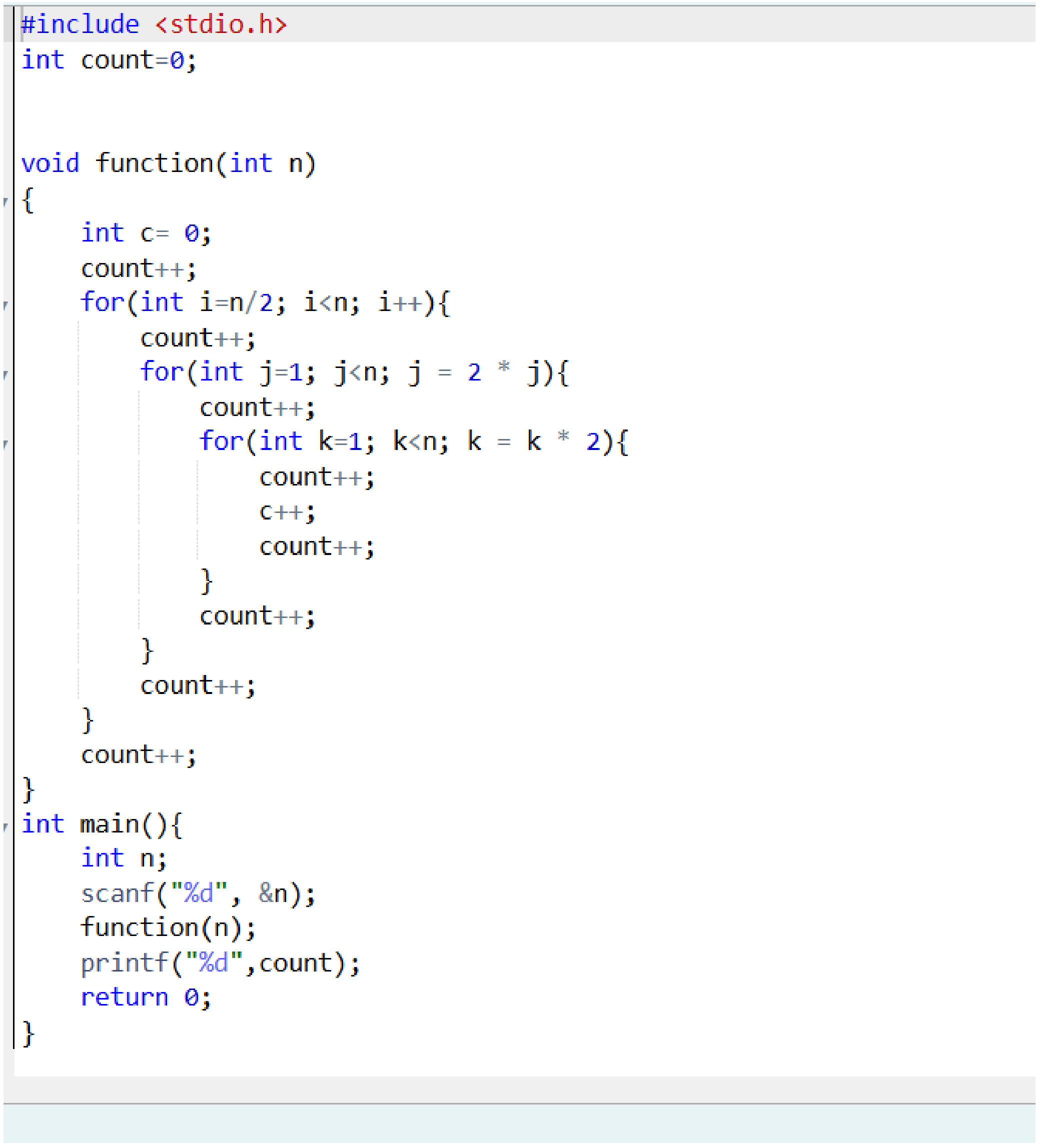
	Input	Expected	Got	
✓	12	31	31	✓
✓	25	54	54	✓
✓	4	12	12	✓



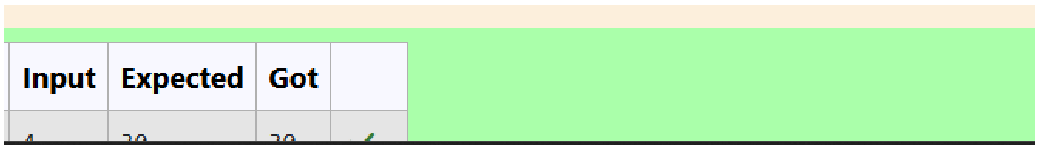
```
Convert the following algorithm into a program and find its time

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



```c
#include <stdio.h>
int count=0;


void function(int 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++;
}
int main(){
    int n;
    scanf("%d", &n);
    function(n);
    printf("%d",count);
    return 0;
}
```

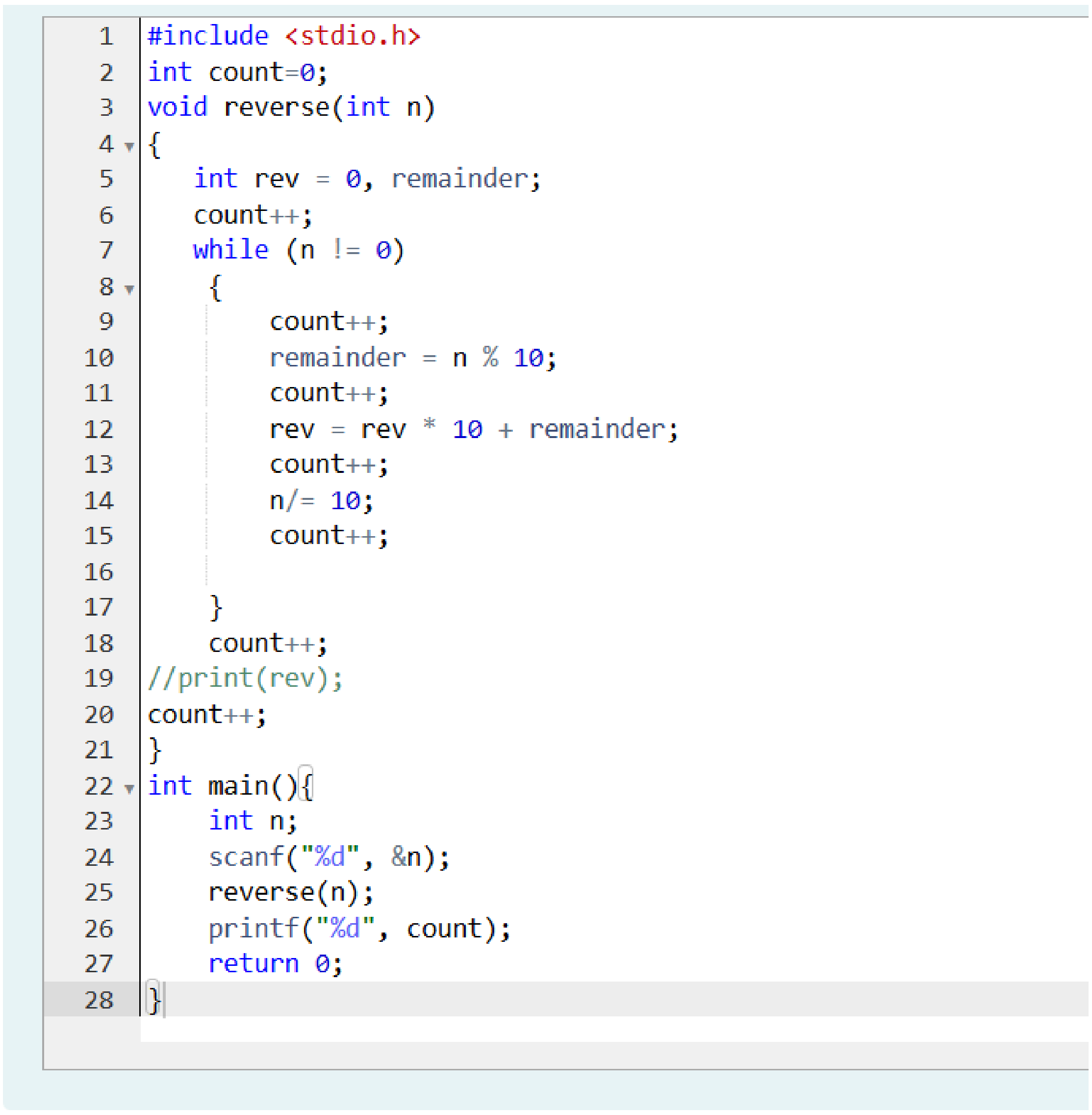


Input	Expected	Got	
4	30	30	✓

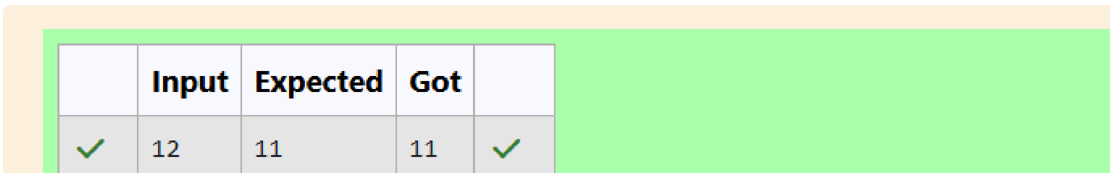
```
Convert the following algorithm into a program and find its time complexity using counter method.
void reverse(int n)
{
   int rev = 0, remainder;
   while (n != 0)
    {
        remainder = n % 10;
        rev = rev * 10 + remainder;
        n/= 10;

    }
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



```c
#include <stdio.h>
int count=0;
void reverse(int n)
{
    int rev = 0, remainder;
    count++;
    while (n != 0)
     {
         count++;
         remainder = n % 10;
         count++;
         rev = rev * 10 + remainder;
         count++;
         n/= 10;
         count++;

     }
     count++;
//print(rev);
count++;
}
int main(){
     int n;
     scanf("%d", &n);
     reverse(n);
     printf("%d", count);
     return 0;
}
```



	Input	Expected	Got	
✓	12	11	11	✓