# Problem 1: Finding Complexity using Counter Method

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;
    }
}
```

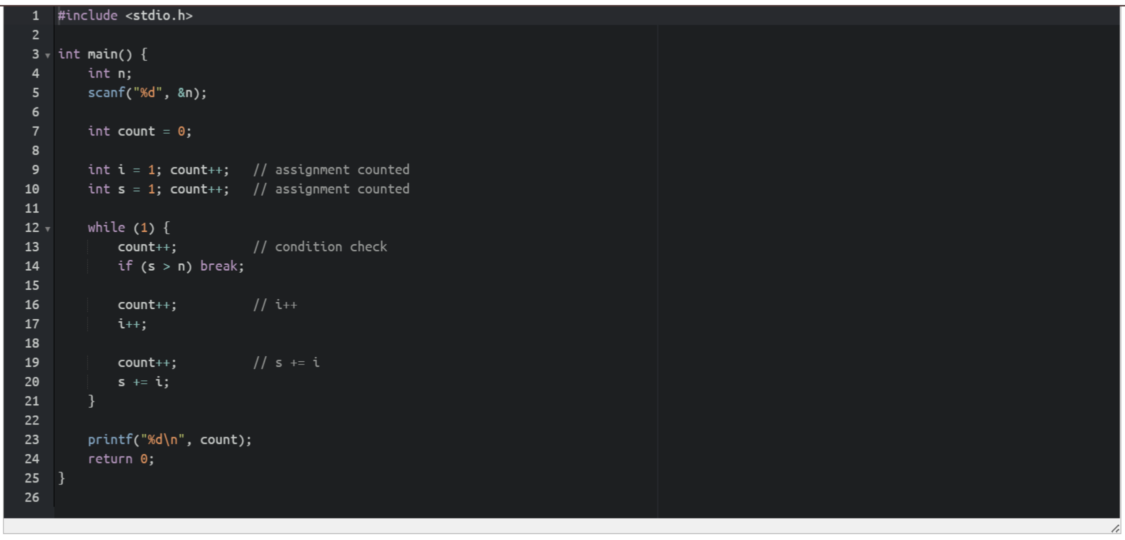
**Note:** No need of counter increment for declarations and scanf() and count variable printf() statements.

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

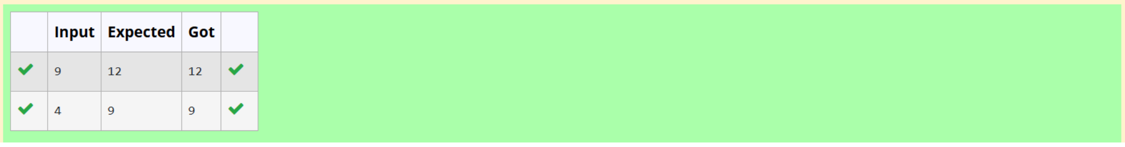
**For example:**

| Input | Result |
|---|---|
| 9 | 12 |

Answer:



```c
#include <stdio.h>

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

    int count = 0;

    int i = 1; count++;   // assignment counted
    int s = 1; count++;   // assignment counted

    while (1) {
        count++;          // condition check
        if (s > n) break;

        count++;          // i++
        i++;

        count++;          // s += i
        s += i;
    }

    printf("%d\n", count);
    return 0;
}
```



| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | 9 | 12 | 12 | ✓ |
| ✓ | 4 | 9 | 9 | ✓ |

## Problem 2: Finding Complexity using Counter method

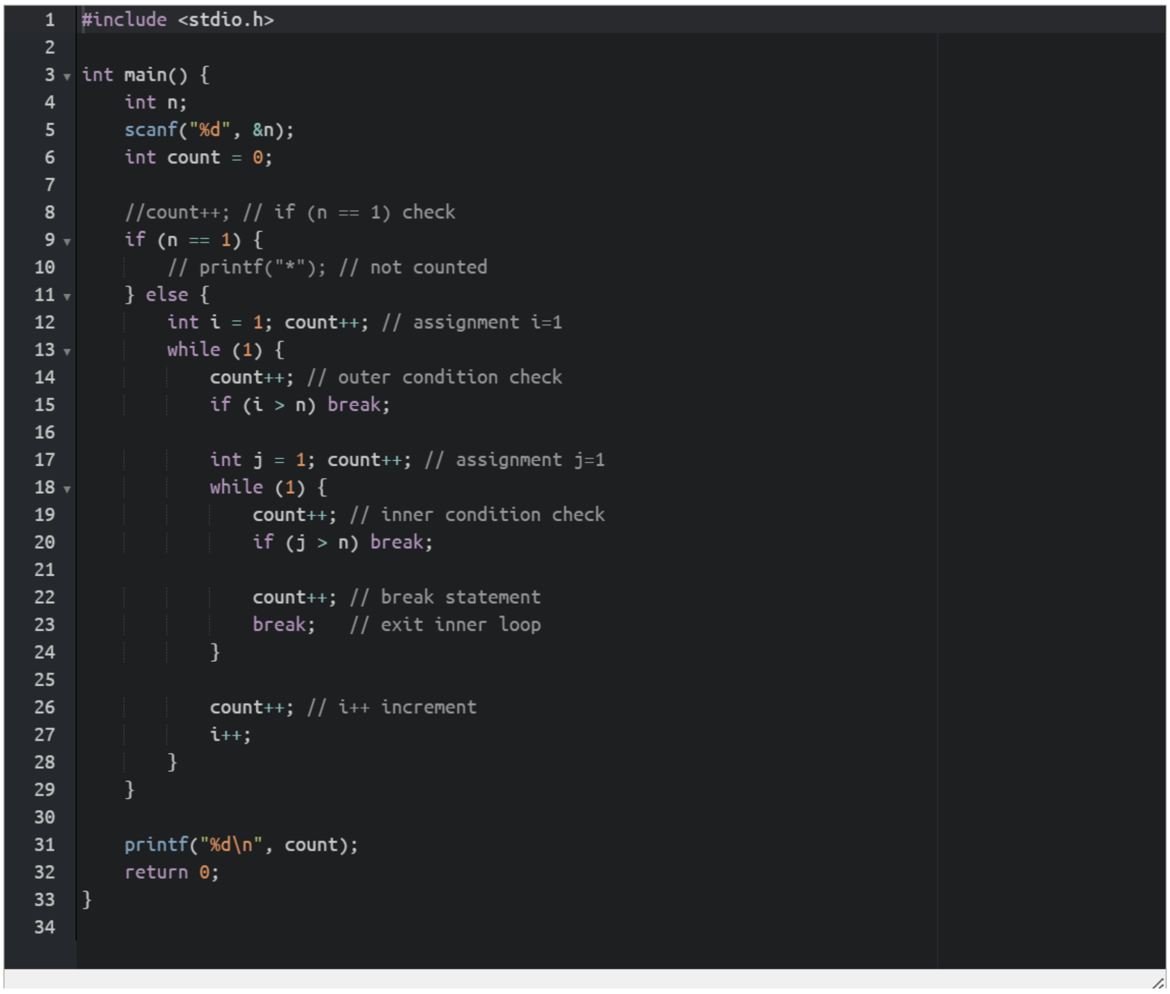
**Question:**

**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**

**Answer:**



```c
#include <stdio.h>

int main() {
    int n;
    scanf("%d", &n);
    int count = 0;

    //count++; // if (n == 1) check
    if (n == 1) {
        // printf("*"); // not counted
    } else {
        int i = 1; count++; // assignment i=1
        while (1) {
            count++; // outer condition check
            if (i > n) break;

            int j = 1; count++; // assignment j=1
            while (1) {
                count++; // inner condition check
                if (j > n) break;

                count++; // break statement
                break;   // exit inner loop
            }

            count++; // i++ increment
            i++;
        }
    }

    printf("%d\n", count);
    return 0;
}

```

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

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

# Problem 3: Finding Complexity using Counter Method

Question:

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

Answer:

```c
#include <stdio.h>

int main() {
    int num;
    scanf("%d", &num);
    int count = 0;

    for (int i = 1; ; i++) {
        count++; // loop condition
        if (i > num) break;

        count++; // if condition check
        if (num % i == 0) {
            count++; // extra for successful condition
            // printf("%d ", i); // not counted
        }
        // i++ not counted
    }

    printf("%d\n", count);
    return 0;
}

```

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

Passed all tests! ✓

**Correct**

Marks for this submission: 1.00/1.00.

## Problem 4: Finding Complexity using Counter Method

Convert the following algorithm into a program and find its time

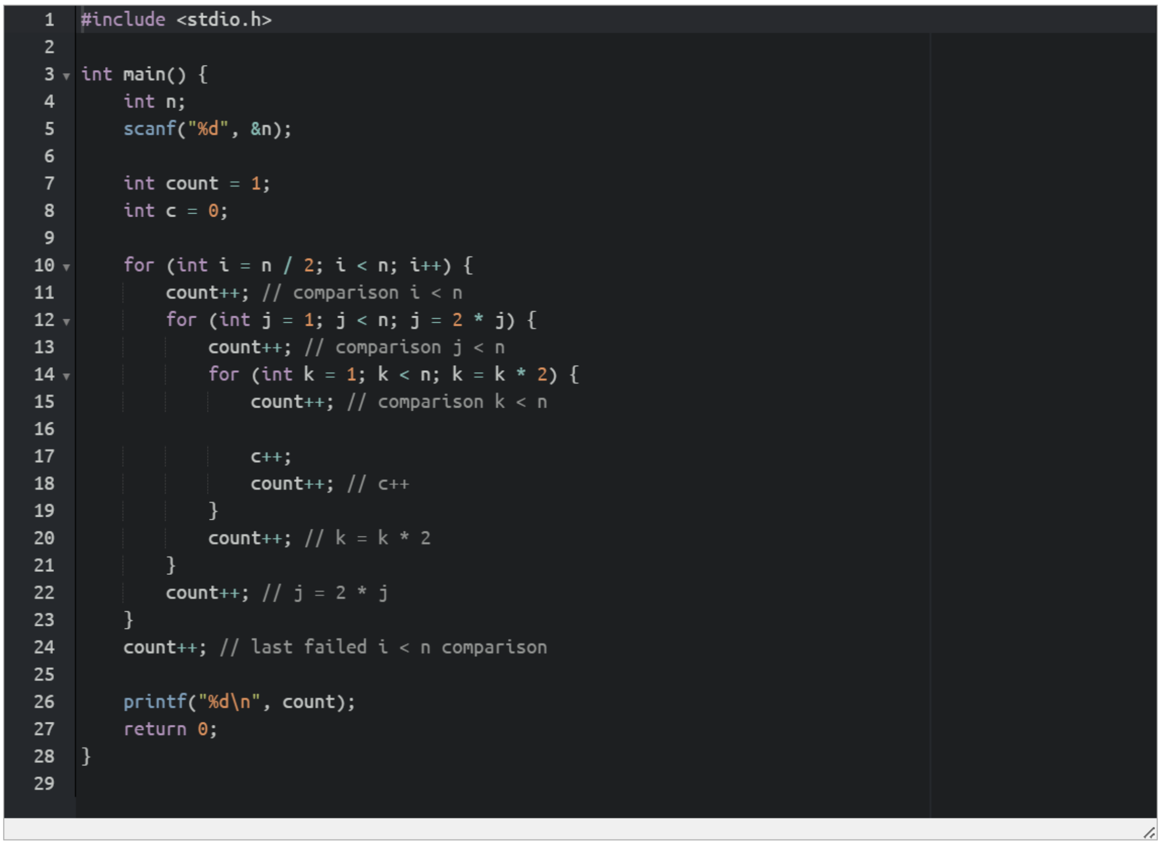
complexity using counter method.

```
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

Answer:



```c
#include <stdio.h>

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

    int count = 1;
    int c = 0;

    for (int i = n / 2; i < n; i++) {
        count++; // comparison i < n
        for (int j = 1; j < n; j = 2 * j) {
            count++; // comparison j < n
            for (int k = 1; k < n; k = k * 2) {
                count++; // comparison k < n

                c++;
                count++; // c++
            }
            count++; // k = k * 2
        }
        count++; // j = 2 * j
    }
    count++; // last failed i < n comparison

    printf("%d\n", count);
    return 0;
}

```

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

Passed all tests! ✓

**Correct**

Marks for this submission: 1.00/1.00.

## Problem 5: Finding Complexity using counter method

Question:

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

Answer:

```c
#include <stdio.h>

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

    int count = 2; // counter variable
    int rev = 0, remainder;

    while (n != 0) {
        count++; // while condition check

        remainder = n % 10;
        count++; // remainder assignment

        rev = rev * 10 + remainder;
        count++; // rev assignment

        n /= 10;
        count++; // n division and assignment
    }
    count++; // final failed while condition check

    printf("%d\n", count);
    return 0;
}
```

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

Passed all tests! ✓

**Correct**

Marks for this submission: 1.00/1.00.