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

ALGORITHM:

PROGRAM:

```
#include <stdio.h>

void function (int n)

{

   int count = 0;


   int i= 1;

   count++;
```

```
    int s =1;
    count++;

    while(s <= n)
    {
        i++;
        s += i;
        count++;
        count++;
        count++;
    }
    count++;
    printf("%d",count);
}
int main()
{
    int n;
    scanf("%d",&n);
    function(n);
}
```

RESULT:

	Input	Expected	Got	
✔	9	12	12	✔
✔	4	9	9	✔

Passed all tests! ✔

Finding Complexity using Counter method

1)

```
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:

PROGRAM:

```
#include <stdio.h>
void func(int n)
{
   int count = 0;
   count++;
   if(n==1)
   {
    //printf("*");

   }
   else
   {
    for(int i=1; i<=n; i++)
    {
      count++;
      for(int j=1; j<=n; j++)
      {
```

```
        count++;
        //printf("*");
        count++;
        //printf("*");
        count++;

        break;
        count++;
      }
     count++;
    }
    count++;
  }



  printf("%d",count);
 }

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

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

Passed all tests! ✔

Problem 3: Finding Complexity using Counter Method

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.

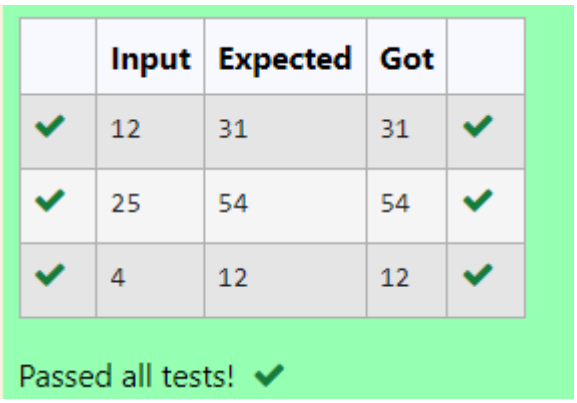
Input:
A positive Integer n
Output:
Print the value of the counter variable

PROGRAM:

```
#include<stdio.h>
int c=0;
void Factor(int num)
 {
   for (int i = 1; i <= num;++i)
   {
      c++;
      if (num % i== 0)
      {
         c++;
```

```
            //printf("%d ", i);
        }
        c++;
    }
    c++;
 }
int main()
{
   int n;
   scanf("%d",&n);
   Factor(n);
   printf("%d",c);
}
```

RESULT:



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

Passed all tests! ✔

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

ALGORITHM:

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){
            count++;
            c++;
            count++;
         }
```

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

RESULT:

	Input	Expected	Got	
✔	4	30	30	✔
✔	10	212	212	✔

Passed all tests! ✔

Problem 5: Finding Complexity using counter method

5)

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

PROGRAM:

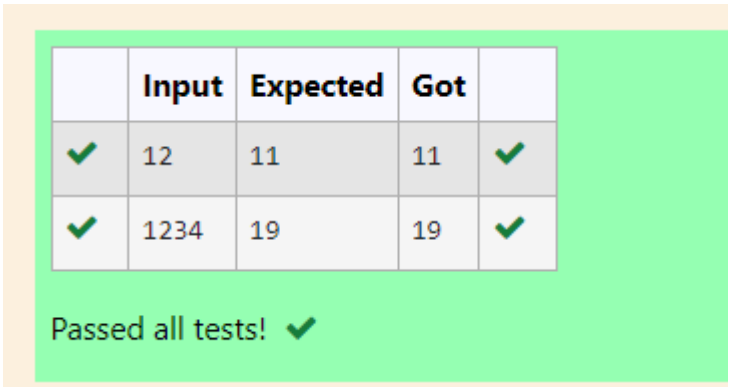
```
#include<stdio.h>
void reverse(int n)
{
  int count=0;
  int rev = 0, remainder;
  count++;
  while (n != 0)
  {
     count++;
     remainder = n % 10;
     count++;
     rev = rev * 10 + remainder;
     count++;
     n/= 10;
     count++;


  }
```

```
    count++;
//print(rev);
count++;
printf("%d",count);
}

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

RESULT:



	Input	Expected	Got	
✔	12	11	11	✔
✔	1234	19	19	✔

Passed all tests! ✔

1-Number of Zeros in a Given Array

Problem Statement

Given an array of 1s and 0s this has all 1s first followed by all 0s. Aim is to find the number of 0s. Write a program using Divide and Conquer to Count the number of zeroes in the given array.

Input Format

First Line Contains Integer m – Size of array

Next m lines Contains m numbers – Elements of an array

Output Format

First Line Contains Integer – Number of zeroes present in the given array.