**[FINDING TIME COMPLEXITY OF ALGORITHMS](http://118.185.187.137/moodle/course/view.php?id=158" \l "section-1)**

**AIM:**

To implement a C program that prints an asterisk pattern based on input n and analyzes its time complexity using a counter method.

**PROGRAM:**

#include<stdio.h>

int main()

{

int n;

scanf("%d",&n);

int count=0;

count++;

if(n==1)

{

//printf("\*");

}

else

{

count++;

for(int i=1; i<=n; i++)

{

count++;

for(int j=1; j<=n; j++)

{

count++;

//printf("\*");

//printf("\*");

break;

count++;

}

count++;

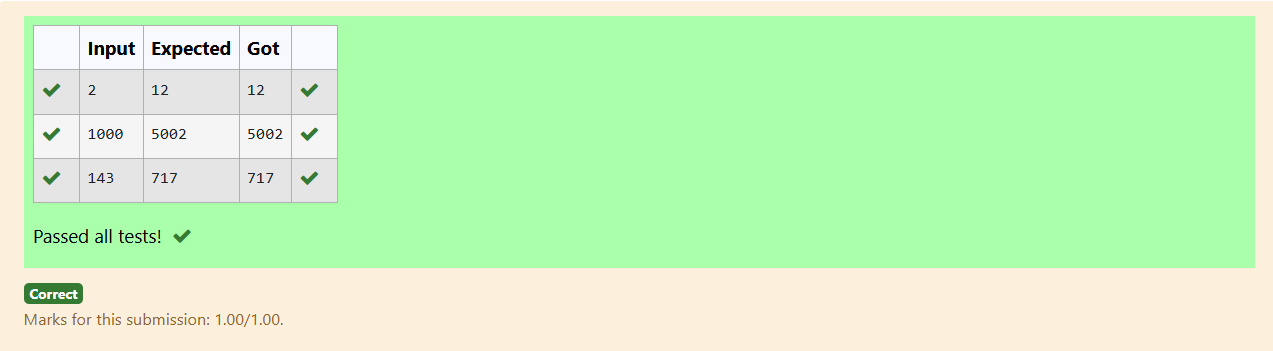
}

}

printf("%d",(5\*n)+2);

}

**OUTPUT:**



**AIM:**

To implement a C program that finds and prints all factors of a given number and analyzes its time complexity using a counter method.

**PROGRAM:**

#include<stdio.h>

int main()

{

int c=0;

int num;

c++;

scanf("%d",&num);

for (int i = 1; i <= num;++i)

{

c+=2;

if (num % i== 0)

{

c++;

//printf("%d ", i);

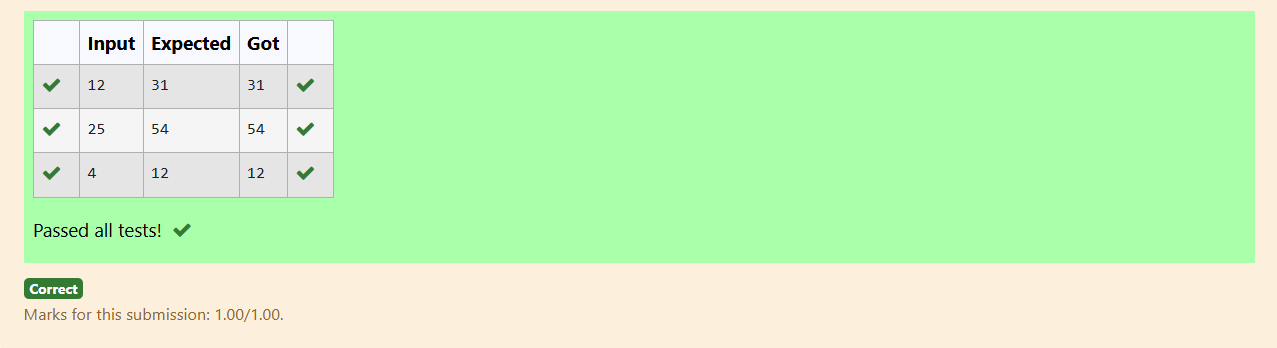
}

}

printf("%d",c);

}

**OUTPUT:**



**AIM:**

To implement a C program that executes a nested loop structure for a given integer n and analyzes its time complexity using a counter method.

**PROGRAM:**

#include<stdio.h>

int main()

{

int count=0;

int c= 0;

int n;

scanf("%d",&n);

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++;

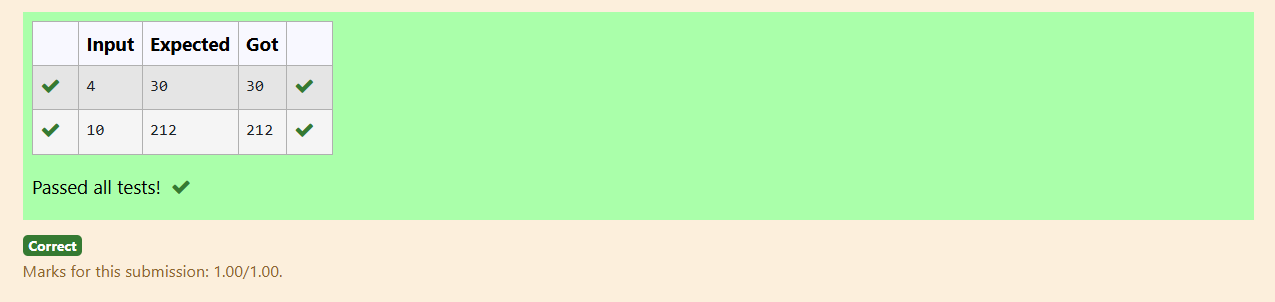
}

count++;

printf("%d",count);

}

**OUTPUT:**



**AIM:**

To implement a C program that reverses the digits of a given number n and analyzes its time complexity using a counter method.

**PROGRAM:**

#include<stdio.h>

int main()

{

int n;

scanf("%d",&n);

int count=0;

count++;

int rev = 0, remainder;

count++;

while (n != 0)

{

count++;

remainder = n % 10;

count++;

rev = rev \* 10 + remainder;

count++;

n/= 10;

count++;

}

count++;

//print(rev);

printf("%d",count);

}

**OUTPUT:**

