Ex. No: 2 **Date: 20.08.24**

Register No.: 230701368 **Name: AL UMA**

Finding Time Complexity of Algorithms

2.A Finding Complexity Using Counter Method

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

ALGORITHM:

int main()

{ set count to 0

Declare n

Read n from the user

Initialize i=0

Increment count by 1

Initialize s=0

Increment count by 1

While(s<=n)

{

Increment count by 1

i++

Increment count by 1

s+=i

Increment count by 1

}

Increment count by 1

Print count

}

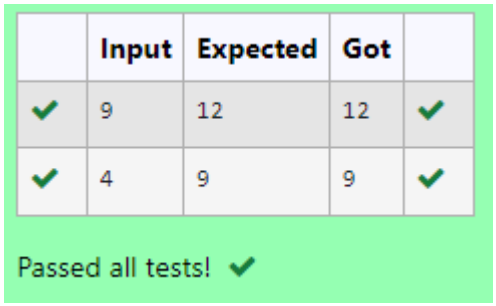
PROGRAM:

```
#include<stdio.h>
int main()
{
    int count=0;
    int n;
    scanf("%d",&n);
    int i=1;
    count++;
    int s=1;
    count++;
    while(s<=n)
```

```
    {
        count++;
        i++;
        count++;
        s+=i;
        count++;

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

OUTPUT:



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

Passed all tests! ✔

2.B Finding Complexity Using Counter Method

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:

```
int main()
{
set count to 0
declare n;
read n
if(n==1){
increment count by 1;
increment count by 1
}
else{
increment  count by 0
for(int i=1;i<=n;i++)
        {
            Increment count by 1
            for(int j=1;j<=n;j++)
            {
                Increment count by 1
                Increment count by 1
                Increment count by 1
                break;
            }Increment count by 1
        }Increment count by 1
    }
    Print count
}
```

PROGRAM:

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

        count++;

    }
    else
    {
        count++;
        for(int i=1;i<=n;i++)
        {
            count++;
            for(int j=1;j<=n;j++)
            {
                count++;
                count++;
                count++;
                break;
            }count++;
        }count++;
    }
    printf("%d",count);
```

}

OUTPUT:

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

2.C Finding Complexity Using Counter Method

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

```
#include<stdio.h>

int main()

{

   Declare num;

   Read num

   Intialize count to 0

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

   {

      Increment count by 1
```

```
        Increment count by 1
        if(num%i==0)
        {
          Increment count by 1
        }

    }Increment count by 1
    Print count
}
```

PROGRAM:

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

2.D Finding Complexity Using Counter Method

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

ALGORITHM:

int main()

{

Declare n;

Read n

Initialize count to 0

Initialize c to 0

Increment count by 1

for(int i=n/2;i<n;i++)

{

```
        Increment count by 1
        for(int j=1;j<n;j=2*j)
        {
        Increment count by 1
        for(int k=1;k<n;k=k*2){
        Increment count by 1
        c++;
        Increment count by 1
} Increment count by 1
      } Increment count by 1
   } Increment count by 1
   Print count
}
```

PROGRAM

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

OUTPUT:

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

2.E Finding Complexity Using Counter Method

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:

int main()

{

Declare n,remainder,rev

Read n

Initialize count to 0

Initialize count to 0

Increment count by 1

while(n!=0)

```
    {
        Increment count by 1
        remainder=n%10;
        Increment count by 1
        rev=rev*10+remainder;
        Increment count by 1
        n/=10;
        Increment count by 1
    } Increment count by 1
    Increment count by 1
    Print count
}
```

PROGRAM

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

OUTPUT:

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