

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:

```

1 #include <stdio.h>
2
3 int main() {
4     int n;
5     scanf("%d", &n);
6
7     int count = 0;
8
9     int i = 1; count++; // assignment counted
10    int s = 1; count++; // assignment counted
11
12    while (1) {
13        count++; // condition check
14        if (s > n) break;
15
16        count++; // i++
17        i++;
18
19        count++; // s += i
20        s += i;
21    }
22
23    printf("%d\n", count);
24    return 0;
25 }
26

```

	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 :

```
1 #include <stdio.h>
2
3 int main() {
4     int n;
5     scanf("%d", &n);
6     int count = 0;
7
8     //count++; // if (n == 1) check
9     if (n == 1) {
10         // printf("*"); // not counted
11     } else {
12         int i = 1; count++; // assignment i=1
13         while (1) {
14             count++; // outer condition check
15             if (i > n) break;
16
17             int j = 1; count++; // assignment j=1
18             while (1) {
19                 count++; // inner condition check
20                 if (j > n) break;
21
22                 count++; // break statement
23                 break; // exit inner loop
24             }
25
26             count++; // i++ increment
27             i++;
28         }
29     }
30
31     printf("%d\n", count);
32     return 0;
33 }
34
```

	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:

```

1 #include <stdio.h>
2
3 int main() {
4     int num;
5     scanf("%d", &num);
6     int count = 0;
7
8     for (int i = 1; ; i++) {
9         count++; // loop condition
10        if (i > num) break;
11
12        count++; // if condition check
13        if (num % i == 0) {
14            count++; // extra for successful condition
15            // printf("%d ", i); // not counted
16        }
17        // i++ not counted
18    }
19
20    printf("%d\n", count);
21    return 0;
22 }
23

```

	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:

```

1 #include <stdio.h>
2
3 int main() {
4     int n;
5     scanf("%d", &n);
6
7     int count = 1;
8     int c = 0;
9
10    for (int i = n / 2; i < n; i++) {
11        count++; // comparison i < n
12        for (int j = 1; j < n; j = 2 * j) {
13            count++; // comparison j < n
14            for (int k = 1; k < n; k = k * 2) {
15                count++; // comparison k < n
16
17                c++;
18                count++; // c++
19            }
20            count++; // k = k * 2
21        }
22        count++; // j = 2 * j
23    }
24    count++; // last failed i < n comparison
25
26    printf("%d\n", count);
27    return 0;
28 }
29

```

	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:

```

1  #include <stdio.h>
2
3  int main() {
4      int n;
5      scanf("%d", &n);
6
7      int count = 2; // counter variable
8      int rev = 0, remainder;
9
10     while (n != 0) {
11         count++; // while condition check
12
13         remainder = n % 10;
14         count++; // remainder assignment
15
16         rev = rev * 10 + remainder;
17         count++; // rev assignment
18
19         n /= 10;
20         count++; // n division and assignment
21     }
22     count++; // final failed while condition check
23
24     printf("%d\n", count);
25     return 0;
26 }
27

```

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

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.