Question **1**

Correct

Marked out of 5.00

Flag question

Write a program that takes as parameter an integer n.

You have to print the number of zeros at the end of the factorial of n.

For example, 3! = 6. The number of zeros are 0. 5! = 120. The number of zeros at the end are 1.

Note: n! < 10^5

Example Input:

3

Output:

0

Example Input:

60

Output:

14

Example Input:

100

Output:

24

Example Input:

1024

Output:

253

**For example:**

| Input | Result |
|---|---|
| 3 | 0 |
| 60 | 14 |
| 100 | 24 |
| 1024 | 253 |

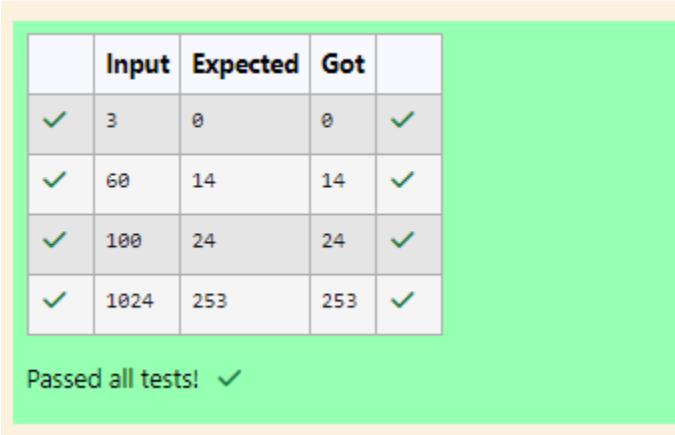
```
import java.util.Scanner;
public class TrailingZeros{
    public static int
    countTrailingZeros(int n){
        int count=0;
        for (int i = 5; n / i >= 1; i*=5    )
            count += n / i;

        return count;
    }

    public static void main(String[] args)
    {
        Scanner scanner =  new Scanner(System.in);

        int n = scanner.nextInt();
```

```
        int result=countTrailingZeros(n);
        System.out.println(result);
        scanner.close();
    }
}
```



| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | 3 | 0 | 0 | ✓ |
| ✓ | 60 | 14 | 14 | ✓ |
| ✓ | 100 | 24 | 24 | ✓ |
| ✓ | 1024 | 253 | 253 | ✓ |

Passed all tests! ✓

Question 2
Correct
Marked out of 5.00
Flag question

You and your friend are movie fans and want to predict if the movie is going to be a hit!
The movie's success formula depends on 2 parameters:
the acting power of the actor (range 0 to 10)
the critic's rating of the movie (range 0 to 10)
The movie is a hit if the acting power is excellent (more than 8) or the rating is excellent (more than 8). This holds true except if either the acting power is poor (less than 2) or rating is poor (less than 2), then the movie is a flop. Otherwise the movie is average.
Write a program that takes 2 integers:
the first integer is the acting power
second integer is the critic's rating.
You have to print Yes if the movie is a hit, Maybe if the movie is average and No if the movie is flop.
Example input:
9 5
Output:
Yes
Example input:
1 9
Output:
No
Example input:
6 4
Output:
Maybe

**For example:**

| Input | Result |
|---|---|
| 9 5 | Yes |
| 1 9 | No |
| 6 4 | Maybe |

```
import java.util.Scanner;
public class MoviePrediction{
    public static void main(String[] args)
    {
        Scanner scanner = new Scanner(System.in);
        int actingPower=scanner.nextInt();
        int criticsRating=scanner.nextInt();
        if(actingPower < 2 || criticsRating < 2){
            System.out.println("No");
```

```
    }
    else if(actingPower>8 || criticsRating>8){
        System.out.println("Yes");
    }
    else{
        System.out.println("Maybe");
    }
    scanner.close();
}
```

| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | 9 5 | Yes | Yes | ✓ |
| ✓ | 1 9 | No | No | ✓ |
| ✓ | 6 4 | Maybe | Maybe | ✓ |

Passed all tests! ✓

Question **3**

Correct

Marked out of 5.00

Flag question

Consider the following sequence:

1st term: 1

2nd term: 1 2 1

3rd term: 1 2 1 3 1 2 1

4th term: 1 2 1 3 1 2 1 4 1 2 1 3 1 2 1

And so on. Write a program that takes as parameter an integer n and prints the nth terms of this sequence.

Example Input:

1

Output:

1

Example Input:

4

Output:

1 2 1 3 1 2 1 4 1 2 1 3 1 2 1

**For example:**

| Input | Result |
|---|---|
| 1 | 1 |
| 2 | 1 2 1 |
| 3 | 1 2 1 3 1 2 1 |
| 4 | 1 2 1 3 1 2 1 4 1 2 1 3 1 2 1 |

```
import java.util.Scanner;
public class SequenceGenerator{
    public static String generateTerm(int n){
        if(n==1){
```

```
            return "1";

        }else{
            String previousTerm = generateTerm(n-1);
            return previousTerm+" " + n + " "+previousTerm;
        }
    }
    public static void main(String[] args){
        Scanner scanner= new Scanner(System.in);
        int n=scanner.nextInt();
        System.out.println(generateTerm(n));
        scanner.close();
    }
}
```

| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | 1 | 1 | 1 | ✓ |
| ✓ | 2 | 1 2 1 | 1 2 1 | ✓ |
| ✓ | 3 | 1 2 1 3 1 2 1 | 1 2 1 3 1 2 1 | ✓ |
| ✓ | 4 | 1 2 1 3 1 2 1 4 1 2 1 3 1 2 1 | 1 2 1 3 1 2 1 4 1 2 1 3 1 2 1 | ✓ |

Passed all tests! ✓