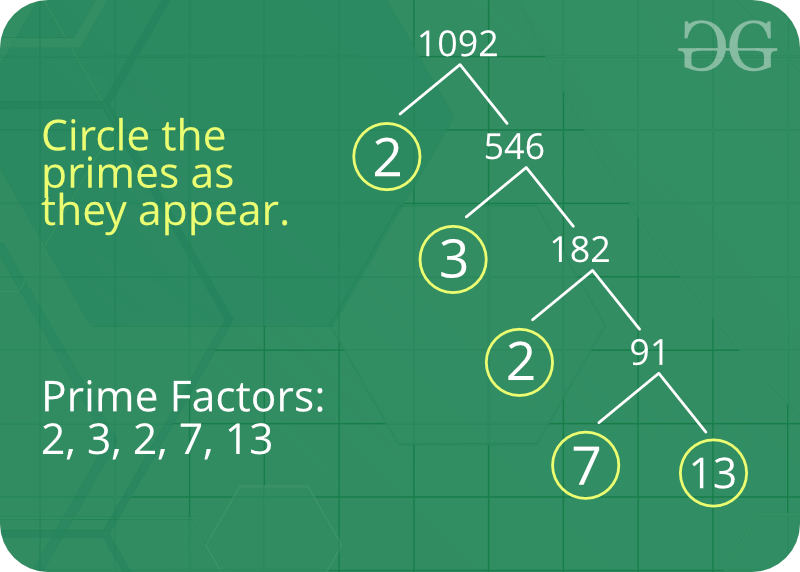
***Prime Factors***

Prime factor is the factor of the given number which is a [prime number](https://www.geeksforgeeks.org/prime-numbers/). Factors are the numbers you multiply together to get another number. In simple words, prime factor is finding which prime numbers multiply together to make the original number.

**Example:** The prime factors of 15 are 3 and 5 (because 3×5=15, and 3 and 5 are prime numbers).



**Some interesting fact about Prime Factor :**

1. There is only one (unique!) set of prime factors for any number.
2. In order to maintain this property of unique prime factorizations, it is necessary that the number one, 1, be categorized as neither prime nor composite.
3. Prime factorizations can help us with divisibility, simplifying fractions, and finding common denominators for fractions.
4. [Pollard’s Rho](https://www.geeksforgeeks.org/pollards-rho-algorithm-prime-factorization/) is a prime factorization algorithm, particularly fast for a large [composite number](https://www.geeksforgeeks.org/composite-number/) with small prime factors.
5. Cryptography is the study of secret codes. Prime Factorization is very important to people who try to make (or break) secret codes based on numbers.

**How to print a prime factor of a number?**  
**Naive solution:**   
Given a number n, write a function to print all prime factors of n. For example, if the input number is 12, then output should be “2 2 3” and if the input number is 315, then output should be “3 3 5 7”.

Following are the steps to find all prime factors:

1. While *n*is divisible by 2, print 2 and divide n by 2.
2. After step 1, *n* must be odd. Now start a loop from i = 3 to square root of *n*. While *i* divides *n*, print *i* and divide *n* by *i*, increment *i* by 2 and continue.
3. If *n* is a prime number and is greater than 2, then *n* will not become 1 by above two steps. So print *n* if it is greater than 2.

C++Java

// Program to print all prime factors

import java.io.\*;

import java.lang.Math;

class GFG {

// A function to print all prime factors

// of a given number n

public static void primeFactors(int n)

{

// Print the number of 2s that divide n

while (n % 2 == 0) {

System.out.print(2 + " ");

n /= 2;

}

// n must be odd at this point. So we can

// skip one element (Note i = i +2)

for (int i = 3; i <= Math.sqrt(n); i += 2) {

// While i divides n, print i and divide n

while (n % i == 0) {

System.out.print(i + " ");

n /= i;

}

}

// This condition is to handle the case when

// n is a prime number greater than 2

if (n > 2)

System.out.print(n);

}

public static void main(String[] args)

{

int n = 315;

primeFactors(n);

}

}

**Output:**

3 3 5 7

Time Complexity: O(sqrt(n))

Auxiliary Space: O(1)

**More efficient solution:**

C++Java

/\*package whatever //do not write package name here \*/

import java.io.\*;

import java.util.\*;

class GFG {

static void printPrimeFactors(int n)

{

if(n <= 1)

return;

while(n % 2 == 0)

{

System.out.print(2+" ");

n = n / 2;

}

while(n % 3 == 0)

{

System.out.print(3+" ");

n = n / 3;

}

for(int i=5; i\*i<=n; i=i+6)

{

while(n % i == 0)

{

System.out.print(i+" ");

n = n / i;

}

while(n % (i + 2) == 0)

{

System.out.print((i + 2)+" ");

n = n / (i + 2);

}

}

if(n > 3)

System.out.print(n+" ");

System.out.println();

}

public static void main (String[] args) {

int n = 315;

printPrimeFactors(n);

}

}

**Output:**

3 3 5 7