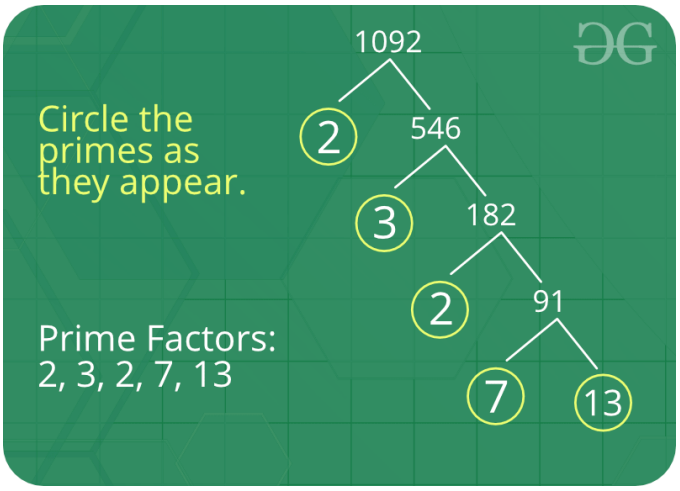
Prime Factorization

Prime factor is the factor of the given number which is a prime number. 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 is a prime factorization algorithm, particularly fast for a large 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.

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".

```
#include<iostream>
using namespace std;

bool isPrime(int n) // function to check if the number is pri
{
    for(int i = 2 ; i < n ; i ++)
    {
        if(n%i == 0)
        {
            return false;
        }
    }
    return true;
}

void primeFactors(int n) // n passed from main is 232
{
    for(int i = 2; i <= n ; i ++)
    {
        if(isPrime(i)) // if i is prime, the control goes int
        {
            int x = i ; // x = 2
            while( n%x == 0 ) // control goes in while block
            {
                cout << i << " " ; // print(i)
                x = x*i ;  // update the value of x
             }
        }
    }
    return;
}

int main()
{
    int n = 232;
    primeFactors(n);
```

```
    return 0;
}
```

Output

```
2 2 2 29
```