```
//*************************************************************************
//*  Compilation:  javac Factors.java
//*  Execution:    java Factors N
//*
//*  Computes the prime factorization of N using brute force.
//*
//*   > java Factors 81
//*   The prime factorization of 81 is: 3 3 3 3
//*
//*   > java Factors 168
//*   The prime factorization of 168 is: 2 2 2 3 7
//*
//*   > java Factors 4444444444
//*   The prime factorization of 4444444444 is: 2 2 11 41 271 9091
//*
//*   > java Factors 4444444444444463
//*   The prime factorization of 4444444444444463 is: 4444444444444463
//*
//*   > java Factors 1000000014000000049
//*   The prime factorization of 1000000014000000049 is: 1000000007 1000000007
//*
//*   Can use these for timing tests - biggest 3, 6, 9, 12, 15, and 18 digit primes
//*   > java Factors 997
//*   > java Factors 999983
//*   > java Factors 999999937
//*   > java Factors 999999999989
//*   > java Factors 999999999999989
//*   > java Factors 999999999999999989
//*
//*************************************************************************/

public class Factors {

    public static void main(String[] args) {

        // command-line argument
        long n = Long.parseLong(args[0]);

        System.out.print("The prime factors of " + n + " are: ");

        // for each potential factor i
        for (long i = 2; i <= n / i; i++) {

            // if i is a factor of N, repeatedly divide it out
            while (n % i == 0) {
                System.out.print(i + " ");
                n = n / i;
            } // while
        } // for

        // if biggest factor occurs only once, n > 1
        if (n > 1) System.out.println(n);
        else       System.out.println();
    } // main
} // Factors
//*   Last comment
```