素因数分解　prime factoring

素数でない数は合成数 (composite numbers) ともよばれ、素数の積として表すことができる。
たとえば 6 は合成数で、$6 = 2 \times 3$ と表すことができる。
また $12 = 2^2 \times 3$ というように表す。
このような表現を、(合成数の) 素因数分解という。
これは、たとえば 2 桁や 3 桁の「小さな」数の場合、ほとんど取るに足りないくらいのやさしい計算である。
しかし数字の桁数が大きくなるとともに飛躍的に計算量が増えて、時間がかかり、その意味で「むずかしく」なる。
100 桁を超えると、巨大なコンピュータをつかったとしても何年も、あるいは何万年もかかると予想される。
そもそも素数を求める方法が、今でもエラストテネスのふるいより本質的に進歩していないのである。
どのように出現するか、その規則性がよくわからない素数による割り算を、こつこつと「しらみつぶし」に行うほかない。
実は、このむずかしさこそが、現代の暗号で中心的な役割を果たすのである。

「情報数理入門」　藤井保憲　著　工学図書株式会社　7 頁　より

問題　与えられた整数を素因数分解するプログラムを作成してください。
例解

```
/*
  素因数分解をします。
  prime factoring
  pf.c
  yukio sugawa
  2010/11/30
*/

#include <stdio.h>

int main()
{
  int  i;
  int  n;
  char opr = '=';

  printf("number->");
  scanf("%d", &n);
  printf("%d", n);
```

```
  i = 2;
  while(n > i){
    if(n % i == 0){
      printf("%c%d", opr, i);
      opr = '*';
      n = n / i;
    }else{
      i++;
    }
  }
  printf("%c%d", opr, i);
  putchar('\n');

  return 0;
}
```