```cpp
//日期：2018/ 时间：
//质因子分解：将一个正整数n写成一个或多个质数的乘积180==2*2*3*3*5
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

struct factor{
    int x,cnt;      //x为质因子，cnt为其个数
}fac[10];//因为2*3*5*7*11*13*17*19*23*29已经超过了int的范围，因此fac数组的大小只需要开到10就行了

const int maxn = 100010;
bool p[maxn]={0};
int prime[maxn]={0},pNum=0;     //素数和素数个数

void eFind_Prime(){
    for(int i=2;i<maxn;i++){
        if(p[i] == false){
            prime[pNum++]=i;
            for(int j=i+i;j<maxn;j+=i){
                //i的倍数全部不是素数。
                p[j] = true;
            }
        }
    }
}

/*      质因子分解的思路
对于一个正整数n来说，如果它存在[2,n]内的质因子，要么这些质因子都小于等于sqrt(n)，
                                                要么只存在一个大于sqrt(n)的质因子，而其余质因子都小于等于sqrt(n)


*/

int main(){
    eFind_Prime();
    int n,num=0;        //num为n的不同质因子的个数
    scanf("%d",&n);

    if(n==1)    printf("1=1"); //特殊判断
    else{
        printf("%d=",n);
        int sqr = (int)sqrt(1.0*n);

        //枚举根号n内的质因子
        for(int i=0; i<pNum && prime[i]<=sqr; i++){
            if(n%prime[i] == 0){        //如果prime[i]是n的质因子
                fac[num].x = prime[i];      //记录该因子
                fac[num].cnt = 0;
                while(n%prime[i] == 0){ //计算出质因子prime[i]的个数
                    fac[num].cnt++;
                    n/=prime[i];
                }
                num++;                  //不同质因子个数加一
            }
```

```
55              if(n == 1) break;           //及时退出循环，节省时间
56          }
57          if(n != 1){                     //如果无法被根号n以内的质因子除尽
58              fac[num].x = n;
59              fac[num++].cnt = 1;
60          }
61
62          for(int i=0;i<num;i++){
63              if(i>0) printf("*");
64              printf("%d",fac[i].x);
65              if(fac[i].cnt > 1)
66                  printf("^%d",fac[i].cnt);
67          }
68
69      }
70
71      return 0;
72  }
73
```