АНАЛИЗ НА РЕШЕНИЕТО НА ЗАДАЧА
РАЗЛИКИ ОТ КУБОВЕ

Ако n се представя като разлика от третите степени на две естествени числа a и b, то, съгласно известната формула:

$$n = a^3 - b^3 = (a - b)(a^2 + ab + b^2),\ a > b.$$

Нека разгледаме всевъзможните представяния на n като произведение на два различни естествени множителя p и q, т. е., $n = pq$, $p < q$. Тогава за някои p и q трябва да е изпълнено съответно $a - b = p$ и $a^2 + ab + b^2 = q$. Ако повдигнем първото равенство на втора степен $(a - b)^2 = p^2$, т. е. $a^2 - 2ab + b^2 = p^2$ и го извадим от второто, получаваме зависимостта $3ab = q - p^2$, която заедно с $a - b = p$ и $a > b$ ни дава необходимо и достатъчно условие за съществуване на нужното представяне, както и алгоритъм за неговото получаване.

Нататък трябва само да се помисли върху някаква по-ефективна реализация на този алгоритъм. Можем да използваме ефективно каноничното представяне на естествено число като произведение на степените на прости множители. Така получаването на различните делители се изчерпва с различните комбинации на степени на съставящите прости множители.

Има някои достатъчни условия за липса на решение. Можем да разгледаме, например, остатъците при делението на 7. Кубовете имат само три възможни остатъка по модул 7: 0, 1 и 6 (≡-1). При изваждане по двойки, от тези остатъци не могат да се получат числата 3 и 4. Следователно, ако n дава остатък 3 или 4 по модул 7, представяне като разлика от кубове не съществува.

Реализация:

```
#include <iostream>
#include <math.h>
using namespace std;
typedef struct
{long long n;
 char cnt;
 long long p[64];
 char d[64];
}Number;
void makeNumber(long long a,Number &N)
{ long long p=3;
  N.n=a;
  N.cnt=0;
  if (!(a&1))
  {N.p[0]=2;
   N.d[0]=0;
   do
   {N.d[0]++;
    a>>=1;
   }while (!(a&1));
   N.cnt++;
  }
  while (p<=sqrt(a))
  { if (!(a%p))
    {N.p[N.cnt]=p;
     N.d[N.cnt]=0;
     do
     {N.d[N.cnt]++;
      a/=p;
     }while (!(a%p));
     N.cnt++;
    }
    p+=2;
  }
  if (a>1)
```

```
  {N.p[N.cnt]=a;
   N.d[N.cnt]=1;
   N.cnt++;
  }
}
bool nextFact(Number N, char *t, long long &a, long long &b)
{int i;
 do
 {for(i=0;i<N.cnt;i++) if (t[i]==N.d[i]) t[i]=0;
                       else break;
  if (i==N.cnt) return false;
  t[i]++;
  a=1;
  for (int j=0;j<N.cnt;j++)
   for (int k=0;k<t[j];k++) a*=N.p[j];
  b=N.n/a;
  if (a<b) return true;
 }while (true);
}
bool Find(long long n,long long &a,long long &b)
{long long p,q,t;
 Number N,T;
 char c[64],tt[64];
 p=n%7;
 if (p==3 || p==4) return false;
 makeNumber(n,N);
 for (int i=1;i<N.cnt;i++) c[i]=0;
 c[0]=-1;
 while (nextFact(N,c,p,q))
 {t=q-p*p;
  if (t>0 && !(t%3))
  {makeNumber(t/3,T);
   for (int i=1;i<T.cnt;i++) tt[i]=0;
   tt[0]=-1;
   while (nextFact(T,tt,b,a))
    if (a-b==p) return true;
  }
 }
 return false;
}
int main()
{
  long long a,b,n;
  cin>>n;
  if (Find(n,a,b)) cout<<a<<' '<<b<<endl;
  else cout<<"NO\n";
  cin>>n;
  if (Find(n,a,b)) cout<<a<<' '<<b<<endl;
  else cout<<"NO\n";
  return 0;
}
```

Автор: Павлин Пеев