###### 实现指数型枚举的一道例题：（2019牛客暑期多校训练营 第九场）

**题意：**

有n个数（1 <= n <= 36）的序列a， {ai} (0 < ai < 2 \* 1e17) , 问挑哪几个数字可以组成s ，

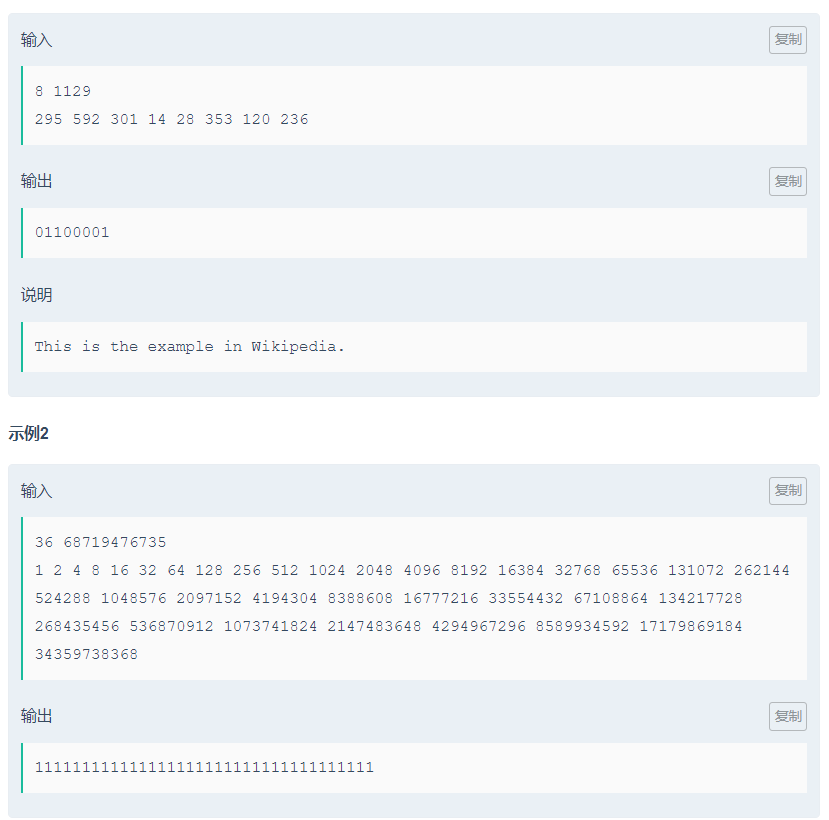
s(0 <= s < 9 \* 1e18) ，用01序列表示答案。

**样例：**

输入n 和s

然后输入n个数；

以下的第一个样例表示选择第2，3，8个数正好凑够1129



**分析：**

如果暴力使用0~2^36-1表示每种状态，共2^36 = 68719476736种状态，会超时。

但是如果巧妙的折半查找枚举,先枚举前半边的所有状态state\_l,从都不选到都选,为2^18=262144,并记录每种状态选择的数的和sum\_l。然后枚举后半边的每种状态state\_r，计算这种状态的sum\_r,并在前半边出现过的状态中查找有没有sum\_l = s - sum\_r的状态，如果有，那么这种左边的状态state\_l和当前找的右边的状态state\_r选的所有数字加起来就是s,这个也是2^18种，所以复杂度变为2\*(2^18).

**代码：**

#include <bits/stdc++.h>

**using** **namespace** std;

**typedef** unsigned **long** **long** ull;

map<ull , **int**> mp;

**int** n1 , n2 , n;

**bool** ok = 0;

ull s, a[45] ,b[45];

**void** dfs1(**int** cur ,**int** state , ull sum){

**if**(cur == n1){

        mp[sum] = state;

**return**;

    }

    dfs1(cur + 1 , state , sum);

    dfs1(cur + 1 , state | 1 << cur , sum + a[cur]);

}

**void** dfs2(**int** cur ,**int** state , ull sum){

**if**(ok) **return**;

**if**(cur == n2){

**if**(mp.find(s - sum)!=mp.end()){

**int** tmp = mp[s - sum];

            ok = 1;

**for**(**int** i = 0 ; i < n1 ; i ++){

**printf**("%d",(tmp >> i) & 1);

            }

**for**(**int** i = 0 ; i < n2 ; i ++){

**printf**("%d",(state >> i) & 1);

            }

        }

**return**;

    }

    dfs2(cur + 1 ,state , sum);

    dfs2(cur + 1 ,state | 1 << cur , sum + b[cur]);

}

**int** main(){

**scanf**("%d%llu",&n,&s);

    n1 = n / 2 , n2 = n - n/2;

**for**(**int** i = 0 ; i < n1 ; i++) **scanf**("%llu",&a[i]);

**for**(**int** i = 0 ; i < n2 ; i++) **scanf**("%llu",&b[i]);

    dfs1(0 , 0 , 0);

    dfs2(0 , 0 , 0);

**return** 0;

}