**★★☆☆☆**

**題組：Problem Set Archive** **with Online Judge**

**題號：12455-bars**

**解題者：蔡昌燁**

**解題日期：2024年3月14日**

**題意：**

給予幾根長度不同鐵桿，要湊出我們需要的長度(可焊接，但不可裁切)，若能湊出需要的長度則輸出YES，不行則輸出NO。

**題意範例:**

給予4個鐵桿長度分別為{10,12,5,7}，要湊出25，輸出為NO

**解法：**

Cin >> 次數(times)、目標長度(OL)、鐵桿數量(Q)、鐵桿長度(儲存於陣列)

Int sum (加總長度)

Int array[Q]

 Bool b[Q] ，儲存值為0代表不加，1代表加

 使用recurrence方法，search函式

1.先算出sum。

2.判斷sum == object。

 3.若true則結束recurrence，false則繼續。

 4.若所有加總都結束還是找無，則cout << NO，並結束recurrence

**解法範例：**

Times = 1

 OL = 7

Array[0,1,2,3,4,5,6] = 12 4 3 5 11 9

b[0,1,2,3,4,5,6] = 0 0 0 0 0 0

* 1 0 0 0 0 0 -> sum = 12
* 0 1 0 0 0 0 -> sum = 4
* 0 0 1 0 0 0 -> sum = 3
* 0 0 0 1 0 0 -> sum = 5
* 0 0 0 0 1 0 -> sum = 11
* 0 0 0 0 0 1 -> sum = 9
* 1 1 0 0 0 0 -> sum = 16
* 0 1 1 0 0 0 -> sum = 7 = OL

Cout << “YES”;

**討論：**

(1)這題是一個sum of subset的問題

(2) for迴圈也可執行計算不需要recurrence

 (3)優化方法：若有很多數值要排列的話，可以建立一個

 bool dp[object]，若x數有被填過，則dp[x] = true，沒被填過

 則 dp[x]=false，超過object則忽略，以範例說明：

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| dp | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| dp[i] | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 |

 若有好幾筆資料需要檢驗答案時可將時間複雜度 O(2^n)，降低至O(n)。

**程式：**

#include <iostream>

using namespace std;

int search(bool b[],int digit,int object,int a[]){ //b儲存1/0，digit儲存數字數量，object是目標，a[]是各length長度

 int temp=0;

 for(int i=0;i<digit;i++){ //把b[i]是1 的 a[i]全部加起來

 if(b[i]==1){

 temp+=a[i];

 }

 }

 if(temp== object){ //如果找到符合的答案，回傳yes並結束迴圈

 cout << "YES" << endl;

 return 0;

 }

 //判斷是否全部加總完了

 int judge=1;

 for(int i=0;i<digit;i++){

 if(b[i] == 0) judge = 0;

 }

 if(judge == 1){

 cout << "NO" << endl;

 return 0;

 }

 //如果還沒找到答案 ， 把其中一個b[]變成1

 for(int i=0;i<digit;i++){

 if(b[i] == 0){

 b[i]=1;

 if(0 == search(b,digit,object,a))

 return 0;

 }

 b[i]=0;

 }

}

int main(){

 int times;

 cin >> times;

 for(int i=0;i<times;i++){

 int object=0,digit=0; //object是要湊出來的值,digit是數字數量

 cin >> object >> digit;

 int array[digit]; // array存各個數字的長度

 for(int k=0;k<digit;k++){ //讀入各length長度

 cin >> array[k] ;

 }

 bool yn[digit];

 for(int j=0;j<digit;j++){yn[j]=0;} //初始化bool陣列

 search(yn,digit,object,array);

 }

 return 0;

}