Apuntes OIFem II Nivel 1

Problemas de concurso

Problema: Perder todas las partes

100 hombres fueron a la guerra. 40 perdieron el brazo, 80 una pierna y todos la cabeza. ¿Cuántos perdieron las tres cosas? Si bien no es posible dar la respuesta exacta, podemos saber que como mínimo fueron 20 y como máximo 40.

En general, si k hombres fueron a la guerra y te damos la lista de cuántos hombres perdieron cada una de las n partes del cuerpo, responde el mínimo y el máximo de hombres que pudieron perder todas las partes a la vez.

ENTRADA: Un número arbitrario de casos, pero no superior a 1000. Cada caso se da en una línea con dos enteros $k, n \leq 1000$ con el número de hombres y de partes, seguido de n enteros entre 0 y k con el número de hombres que perdieron cada una de las n partes.

SALIDA: Para cada caso escribe dos enteros, con el número mínimo y el número máximo de hombres que pudieron perder todas las partes a la vez.

Puntuación parcial

Link al problema: Perder todas las partes

Solución

Calculamos el mínimo y el máximo de hombres que perdieron todas sus partes a medida que leemos los casos de entrada. Para calcular el máximo, calculamos el mínimo entre el máximo que teníamos y la nueva entrada. No puede pasar que haya más hombres sin todas las partes en cada entrada, es decir, el máximo solo puede disminuir porque no tiene sentido que si teníamos `maximo` como el número máximo, entonces `hombres - maximo` seguro que no tendrán todas las partes.

Para calcular el mínimo, vemos que si la suma del mínimo que teníamos `minimo` y la nueva entrada `actual` no llegan al número de hombres, entonces el mínimo es 0. Sin embargo, si `minimo + actual - hombres` ≥ 0, entonces este será el nuevo mínimo. Esta situación se escribe como: `minimo = max(minimo + actual - hombres, 0);`

```cpp
#include <bits/stdc++.h>
using namespace std;

int main () {
    int hombres, partes;
    while (cin >> hombres >> partes) {
        int maximo = hombres, minimo = 0;
```

```
        int actual;
        cin >> actual;
        maximo = actual; minimo = actual;
        for (int i = 1; i < partes; ++i) {
            cin >> actual;
            maximo = min(maximo, actual);
            minimo = max(minimo + actual - hombres, 0);
        }
        cout << minimo << ' ' << maximo << '\n';
    }
}
```

Problema: Diamantes

Un príncipe muy rico tiene n diamantes. Cada diamante $1 \leq i \leq n$ tiene cierto valor vino. La tradición manda que, antes de casarse, el príncipe haga un regalo de valor exactamente V a su princesa. El príncipe quiere regalarle exactamente dos de sus diamantes, pero no sabe cómo decidir rápidamente si puede hacerlo o no. ¿Puedes ayudarlo?

Por ejemplo, si $n = 6$ y el valor de los diamantes es $5, 8, 6, 2, 6, 20$, entonces es posible realizar un regalo de valor $V = 10(8 + 2)$ o bien un regalo de valor $V = 12(6 + 6)$, pero no es posible realizar un regalo de valor $V = 9$.

ENTRADA: La entrada consiste en varios casos. Cada caso empieza con el valor V del regalo (un natural entre 1 y 10^8) y el número n de diamantes (un natural entre 1 y 10^5) en ese orden. Luego vienen n naturales entre 1 y 10^8 indicando el valor de cada diamante. Un caso con $V = n = 0$ marca el final de la entrada.

SALIDA: Para cada caso, escriba una línea con "married" o "single" según si el príncipe puede hacer el regalo o no.

Link al problema: Diamantes

Solución

Ordenamos el vector `diamantes` (de mida n) que contiene las joyas de manera que quede ordenado de forma no-decreciente. Entonces, creamos dos índices i y j que recorren el vector de los índices de 0 a $n - 1$ y de $n - 1$ a 0, respectivamente. Mientras que $i < j$, comprobamos si la suma de los valores de `diamantes` en los índices i, j da el valor esperado `x` de la joya o no. Si la suma es menor que `x`, añadimos una unidad al índice i. Si la suma es mayor que `x`, restamos una unidad al índice j.

```
#include <bits/stdc++.h>
using namespace std;

#define endl '\n'

bool can_sum(int valor, vector<int>& diamonds)
{
    int i = 0, j = (int)diamonds.size() - 1;
    while (i < j) {
        if (diamonds[i] + diamonds[j] == valor)
            return true;
        else if (diamonds[i] + diamonds[j] > valor)
            --j;
```

```
14          else if (diamonds[i] + diamonds[j] < valor)
15              ++i;
16      }
17      return false;
18  }
19
20  int main()
21  {
22      int v, n;
23      while (cin >> v >> n and v != 0) {
24          vector<int> diamonds(n);
25          for (int i = 0; i < n; ++i)
26              cin >> diamonds[i];
27          sort((diamonds).begin(), (diamonds).end());
28          if (can_sum(v, diamonds)) cout << "married" << endl;
29          else cout << "single" << endl;
30      }
31  }
```