8

Problema 34

SOTILLO SOTILLO, CLARA (TAIS96)

ID envio	Usuario/a	Hora envío	Veredicto
61368	TAIS96	2022-11-02 10:33	AC
61259	TAIS96	2022-11-02 09:19	AC

Fichero Source.cpp

```
 *
 * Indicad el nombre completo y usuario del juez de quienes habéis hecho esta solución:
 * Estudiante 1:Carlos Gomez Lopez TAIS45
 * Estudiante 2:Clara Sotillo Sotillo TAIS96
 *

El coste en el caso peor de este algoritmo es O(N log N) siendo N el numero de partidos que
    se jugaran.
Este coste se debe al uso de la funcion sort para ordenar los vectores, esta funcion tiene
    coste O(N log N).

Creamos dos vectores uno para guardar los puntos que obtendra el equipo rival y otro para
    guardar los que obtendran los Brocos.
ordenamos los puntos del equipo rival de menor a mayor y los de los Brocos de mayor a menor.
Si los puntos de los brocos en esa misma posicion son mayores que los del rival, estos se
    restan y se añaden al sumatorio.

Comprobacion algoritmo voraz:

Supongamos la puntuacion de los equipos rivales de forma que i<=j => r[i]<=r[j]
Supongamos la puntuacion de los Brocos de forma que i<=j => b[i]>=b[j]

Recorremos la puntuacion de los equipos rivales de menor a mayor y el algoritmo voraz V
    selecciona la puntuacion de los Brocos mas alta
que le proporciona la mayor suma de diferencias.

Por tanto, el aspecto de la solucion voraz es una secuencia decreciente de indices de puntos
    (que le permiten ganar) y luego el resto
son derrotas.

Ejemplo:

r: 20 30 40 80
b: 50 40 30 30

En la solucion voraz las posiciones 0, 1 son victorias y el resto son derrotas o empates.

Sea O una solución óptima y sea i la primera posición donde O y V difieren: O[i] != V[i].
Entonces la puntuacion V[i] ha sido asignada más adelante en O. Sea k la posición que cumple
    que
O[k] = V[i].
```

Esta es la situación:

```
r  E[0] <= E[1] <= ...

V  V[0]    V[1]   ... |  V[i]

      =       =   ... |  !=

O  O[0]    O[1]   ... |  O[i]  ...  O[k]=V[i]
```

CASO 1: b[V[i]] > r[i] (hay victoria en la voraz)

CASO 1.1: Si hubiese derrota o empate en la optima en el partido i, b[O[i]]<= r[i] podemos
intercambiar i y k sin perder victorias, pero tambien sin perder valor en el sumatorio de diferencias, ya que la i
va a convertirse en victoria porque b[V[i]] > r[i], pero podria darse el caso en que perdiese valor en el sumatorio
(tampoco va a ganar ambas porque es óptima).

CASO 1.2: Si hay victoria en el partido i en la optima, b[O[i]] > r[i], entonces b[O[i]] < b[V[i]] porque de los partidos
que quedan por asignar b[V[i]] es la que produce la mayor diferencia de puntos, por la estrategia voraz.
Al cambiar la posición i con la k, el partido k, va a pasar a ser derrota, ya que, si fuese victoria, lo habría detectado
el algoritmo voraz.
Al intercambiarlos en O no se reducen el número de victorias pero si se incrementa el valor en el sumatorio de difrencias:
en el partido i seguimos ganando pero con mayor diferencia mientras que en el partido k van a perder o empatar a lo mucho.

CASO 2: b[V[i]] <= r[i] (hay derrota en la voraz)

Si hay derrota en la voraz no puede existir ninguna otra combinacion de resultados que te proporcione
mayor diferencia de puntos y por tanto b[O[i]] <= r[i] ya que en caso contrario la solucion voraz la hubiese
asignado.

Tampoco hay victoria en k porque r[i] <= r[k] asi que simplemente intercambiamos dos derrotas para que se parezca mas
a la voraz.

```cpp
int resolver(vector<int>& b, vector<int>& r) {
    sort(b.begin(), b.end());
    sort(r.begin(), r.end(),greater<int>());
    int sum = 0;
```

```
    for (int i = 0; i < b.size(); i++) {

        if (b[i] > r[i]) {
            sum += b[i] - r[i];
        }

    }


    return sum;
}

bool resuelveCaso() {

    int N, aux;
    cin >> N;  // número de partidos

    if (N == 0)
        return false;

    // leer el resto del caso y resolverlo
    vector<int> b, r;

    for (int i = 0; i < N; i++) {
        cin >> aux;
        r.push_back(aux);
    }

    for (int i = 0; i < N; i++) {
        cin >> aux;
        b.push_back(aux);
    }

    cout << resolver(b, r) << "\n";
    return true;
}
```