# Soluciones entrenos OIE

## Garage

Un parking tiene $N$ plazas, numeradas de 1 a $N$. El parking abre cada mañana sin ningún coche, y opera de la siguiente manera a lo largo del día. Cuando un coche llega al parking, si no hay ningún puesto libre, el coche espera en la entrada hasta que haya sitio. Si hay algún espacio libre, el coche va, de entre las plazas de parking libres, a la que tenga el menor número. Si llegan más coches mientras algún coche está esperando, se ponen en fila, en el orden en que han llegado. En el momento en que un coche sale del parking, el primer coche de la cola de espera aparca en el sitio que ha quedado vacío.

El coste de aparcar en una plaza, en dólares, es el peso del coche, en kilogramos, multiplicado por la tarifa de esa plaza de parking. El coste no depende del tiempo que pase en coche en el parking.

Sabemos que llegarán al parking $M$ coches y sabemos el orden de las llegadas y salidas. Calcula cuantos dólares se recaudarán.

Escribe un programa que, dadas las tarifas de cada una de las plazas de parking, el peso de cada coche y el orden de llegadas y salidas, determine cuanto se recaudará.

### Input Format

La entrada empieza con dos enteros, $1 \leq N \leq 100$ y $1 \leq M \leq 2000$ separados por un espacio.

Las siguientes $N$ líneas contienen las tarifas de las plazas de parking. La $i$-ésima de estas líneas contiene un entero $1 \leq R_i \leq 100$, la tarifa de la $i$-ésima plaza de parking en dólares por kilogramo.

Las siguientes $M$ líneas contienen el peso de los coches que van a llegar al parking. Los coches están numerados de 1 a $M$ sin ningún orden en particular. La $k$-ésima de estas líneas contiene un entero $1 \leq W_k \leq 10000$ el peso del coche $k$ en kilogramos.

Las siguientes $2M$ líneas contienen las llegadas y salidas de los coches en orden cronológico. Un entero positivo $i$ indica que el coche $i$ llega al parking. Un entero negativo $-i$ indica que el coche $i$ sale del parking. Se garantiza que ningún coche se tendrá que ir mientras esté en la cola ni se irá antes de haber llegado al parking. Además, todos los coches desde 1 hasta $M$ aparecerán exactamente 2 veces en esta secuencia, primero llegando al parking y luego saliendo de este.

### Output format

Escribe una sola línea con un entero: el número total de dólares que se recaudará hoy.

# Solución

Simplemente simulamos el proceso. Cuando llega un coche miramos si hay sitio. Si lo hay lo ponemos en el primero que encontremos, si no, lo ponemos en la cola de espera (en una queue). Cuando salga un coche, si hay coches esperando en la cola, cogemos el primero y lo ponemos en la plaza que se acaba de vaciar.

Para saber que plazas están ocupadas y cuales libres, podemos tener un vector, de modo que Ocupado[i] sea 1 si la $i$-ésima plaza de parking está ocupada y 0 si está libre. Además, deberemos saber que plaza ocupaba un coche cuando éste se vaya del parking. Con lo que podemos tener otro vector con las posiciones que ocupan los coches.

# Código

## C++

```cpp
#include <iostream>
#include <vector>
#include <queue>

using namespace std;
typedef vector <int> vi;

int main (){
    int n, m;
    cin >> n >> m;

    vi Tarifa(n), Peso(m+1); //Tarifas de las plazas y peso de cada coche
    for (int i = 0; i < n; ++i) cin >> Tarifa[i];
    for (int i = 1; i <= m; ++i) cin >> Peso[i];

    vi Ocupado(n, 0), Posicion(m+1, -1); //Ocupado[i] es 1 si la plaza i está ocupada, si no es 0. Posicion[k] es la plaza en la que está aparcado el coche k o -1 si no está aparcado
    queue <int> cola; //La cola de espera
    int solucion = 0; //Guardamos aquí la solución

    for (int k = 0; k < 2*m; ++k){
        int x;
        cin >> x;

        if (x > 0){ //Entra un coche
            for (int i = 0; i < n and Posicion[x] == -1; ++i){
                if (Ocupado[i] == 0){ //Hemos encontrado un sitio
                    Posicion[x] = i;
                    Ocupado[i] = 1;
                    solucion += Peso[x] * Tarifa[i]; //Actualizamos la solución
                }
            }
            if (Posicion[x] == -1) cola.push(x); //No hay sitios, lo ponemos en la cola
        }
        else{ //Sale un coche
            int pos = Posicion[-x];
            Ocupado[pos] = 0;

            if (not cola.empty()){ //Si hay coches en la cola
                int y = cola.front(); cola.pop();
                Posicion[y] = pos; //El primer coche de la cola lo sustituye
                Ocupado[pos] = 1;
                solucion += Peso[y] * Tarifa[pos]; //Actualizamos la solución
            }
        }
    }
    cout << solucion << endl;
}
```