Pizarra

Autores: Darío Martínez Ramírez y Elena V. R.

El enunciado nos da dos listas de números a_i, b_i y nos pide que convirtamos a en una lista que contenga todos los números de b. Supongamos que ambas listas están ordenadas.

La primera observación es que si $a_i \leq b_j \leq a_{i+1}$, siempre es óptimo conseguir b_j a partir de a_i o a_{i+1}, ya que si lo consiguiéramos a partir de otro número de a, estaríamos gastando dinero para construir a_i o a_{i+1} como paso intermedio, cuando duplicarlos al principio es gratis.

Podemos por tanto resolver el problema independientemente para cada a_i, a_{i+1} y los números de b que hay en el medio, y luego sumar las respuestas. Sean estos $a_i \leq b_l \leq b_{l+1} \leq \ldots \leq b_r \leq a_{i+1}$.

Claramente la forma óptima de generar $b_l, \ldots, b_r$ es elegir algún $l \leq x \leq r+1$ y generar $b_l, \ldots, b_{x-1}$ a partir de a_i y el resto a partir de a_{i+1} (ya que de nuevo, hacer lo contrario significaría que estás generando como paso intermedio el mismo número desde a_i y a_{i+1}, que es una pérdida de dinero respecto a generarla desde un solo lado y luego duplicarlo).

Para generar $b_l, \ldots, b_{x-1}$, primero duplicamos a_i gratis, luego generamos b_l a partir de la copia (con coste $r(b_l - a_i)$), luego lo duplicamos, ahora generamos b_{l+1} a partir de la copia (con coste $r(b_{l+1} - b_l)$), luego de duplicamos, etc.

El coste de todo esto será por tanto $r((b_l - a_i) + (b_{l+1} - b_l) + \ldots + (b_{x-1} - b_{x-2})) = r(b_{x-1} - a_i)$ euros (si $x = l$ el coste será 0). Similarmente, el coste de generar $b_x, \ldots, b_r$ será $l(a_{i+1} - b_x)$ euros (si $x = r+1$ el coste será 0).

Por tanto, podemos simplemente iterar por todos los valores de x, calcular su coste, y elegir el mejor.

Hacer esto para todos los índices i nos da la respuesta correcta (también hay que añadir el coste de generar los valores de b_i más pequeños que a_1 y más grandes que a_n. La forma es básicamente la misma, pero estás obligado a generarlos todos desde el mismo lado).

```cpp
#include<bits/stdc++.h>

using namespace std;

#define int long long
#define vi vector<int>
#define vvi vector<vi>
#define pb push_back

#define sz(x) (int)(x).size()

signed main(){
    int t;
    cin >> t;
    while (t--){
        int n,m,l,r;
        cin >> n >> m >> l >> r;
        vi a(n), b(m);
        for (int &x:a) cin >> x;
        for (int &x:b) cin >> x;
        sort(a.begin(),a.end());
        sort(b.begin(),b.end());

        int ans=0;
        vvi intervals(n-1);
        int ac=0;
        if (b[0]<a[0]) ans+=l*(a[0]-b[0]);
        if (b.back()>a.back()) ans += r*(b.back()-a.back());
        for (int x:b){
            while (ac<n && x>=a[ac]) ac++; // ac is first bigger
            if (ac!=0 && ac!=n) intervals[ac-1].pb(x);
        }
        for (int i=0;i<n-1;i++) if (sz(intervals[i])){
            int k = sz(intervals[i]);
            int interans = min(l*(a[i+1]-intervals[i][0]),r*(intervals[i].back()-a[i]));
            for (int j=0;j<k-1;j++) interans = min(interans, r*(intervals[i][j]-a[i])+l*(a[i+1]-intervals[i][j+1]));
            ans+=interans;
        }

        cout << ans << '\n';
    }
}
```