9'5

Problema 53

GONZÁLEZ RAMIRO, IVÁN (TAIS50)

ID envio	Usuario/a	Hora envío	Veredicto
64244	TAIS50	2022-11-30 10:40	AC
64220	TAIS50	2022-11-30 10:17	AC

Fichero Source.cpp

```
 *
 * Indicad el nombre completo y usuario del juez de quienes habéis hecho esta solución:
 * Estudiante 1: MICHAEL PAREDES SANCHEZ TAIS74
 * Estudiante 2: IVAN GONZALEZ RAMIRO TAIS50
 *

El problema es de programacion dinamica con recorrido por diagonales.


Saliendo del pueblo i miras cual es el coste de llegar a el pueblo j saliendo de todos los
    pueblos anteriores(los k),
para saber el mejor precio lo que se hace es mirar el precio para llegar al k (i<=k<j)
(que es el mejor posible ya calculado por la recursividad) y le sumas el alquiler de ir de k
    a j.
y te quedas con el mejor precio posible entre todos las opciones de ir al pueblo j desde i
    con su posible escala en k


casos recursivos(i,j) = min(i<=k<j) {matrices(i,k)+alquiler(k,j)}
casos bases(i,i)=0;
llamada inicial =(1,N)(N el numero de pueblos)

coste de O(n^3) en tiempo y O(n*n) en espacio(matriz sol) siendo n el numero de pueblos.

 Escribe aquí un comentario general sobre la solución, explicando cómo
 se resuelve el problema
```

```
bool resuelveCaso() {

    // leemos la entrada
    int N;
    cin >> N;

    if (!cin)
        return false;

    // leemos los alquileres
    vector<vector<int>> alquiler(N, vector<int>(N, 0));
    for (int i = 0; i < N - 1; ++i) {
        for (int j = i + 1; j < N; ++j) {
```

```
            cin >> alquiler[i][j];
        }
    }
    Matriz<int> sol(N, N, 0);
    for (int d = 1; d <= N-1 ; ++d) // recorre diagonales
        for (int i = 0; i < N - d; ++i) { // recorre elementos de diagonal
            int j = i + d;
            sol[i][j] = 1000000;
            for (int k = i; k <= j-1; ++k) {
                int temp = sol[i][k] + alquiler[k][j];
                if (temp < sol[i][j]) { // es mejor partir por k
                    sol[i][j] = temp;
                }
            }
        }
    for (int i = 0; i < N - 1; ++i) {
        for (int j = i + 1; j < N; ++j) {
            cout << sol[i][j]<<"␣";
        }
        cout << "\n";
    }
    // resolver el caso



    return true;
}
```