# ALGORÍTMICA PRÁCTICA 4: PROGRAMACIÓN DINÁMICA



# UNIVERSIDAD DE GRANADA

Grupo MT™

Mario Soriano Morón Miguel Torres Alonso José Teodosio Lorente Vallecillos

# Índice

- 1. Ejercicio 1
  - 1.1 Diseño de resolución por etapas y ecuación recurrente
  - 1.2 Diseño de la memoria
  - 1.3 Verificación del P.O.B.
  - 1.4 Diseño del algoritmo de cálculo de coste óptimo.
  - 1.5 Diseño del algoritmo de recuperación de la solución.
- 2. Ejercicio 2
  - 2.1 Diseño de resolución por etapas y ecuación recurrente
  - 2.2 Diseño de la memoria
  - 2.3 Verificación del P.O.B.
  - 2.4 Diseño del algoritmo de cálculo de coste óptimo.
  - 2.5 Diseño del algoritmo de recuperación de la solución.
- 3. Ejercicio 3
  - 3.1 Diseño de resolución por etapas y ecuación recurrente
  - 3.2 Diseño de la memoria
  - 3.3 Verificación del P.O.B.
  - 3.4 Diseño del algoritmo de cálculo de coste óptimo.
  - 3.5 Diseño del algoritmo de recuperación de la solución.

# 1. Ejercicio 1

#### 1.1 Diseño de resolución por etapas y ecuación recurrente

- Resolución del problema por etapas:

El problema se puede resolver por etapas. En cada etapa se considera pasar por una arista (diccionario) intermedio 'k' para cada par de nodos (idiomas) de origen y destino.

#### - Ecuación recurrente:

Para este problema podemos interpretar cada idioma como un nodo de un grafo (no dirigido) y los diccionarios como aristas que unen nodos (idiomas). Consideramos que todas las aristas tienen el mismo peso. Asumimos también que 'D' es la matriz de adyacencia del grafo, y D[i][j] es la distancia para ir directos desde el nodo 'i' al nodo 'j'. D[i][j] =  $+\infty$  si no hay arco entre 'i' y 'j'. Llamaremos  $D_k[i][j]$  al coste del camino mínimo entre 'i' y 'j', con nodos intermedios en el conjunto  $\{1...k\}$ . Si 'k' = 0, no hay nodos intermedios.

El caso base de nuestra ecuación recurrente sería  $D_0[i][j] = D[i][j]$ , es decir, no hay ningún nodo intermedio entre 'i' y 'j'.

El caso general tiene en cuenta dos posibilidades pudiendo pasar por los nodos 1 a 'k':

- Que el camino de 'i' a 'j' no pase por 'k', D<sub>k-1</sub>[i][j].
- Que el camino de 'i' a 'j' pase por 'k',  $D_{k-1}[i][k] + D_{k-1}[k][j]$ .

Por tanto, nos quedaría:

$$D_{k}[i][j] = min\{ D_{k-1}[i][j], D_{k-1}[i][k] + D_{k-1}[k][j] \}$$

#### - Función objetivo:

La función objetivo es minimizar  $D_n[i][j]$ , con 'n' el número de nodos del grafo, para todo 'i' y todo 'j'.

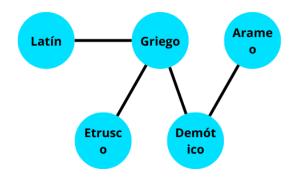
#### 1.2 Diseño de la memoria

La solución al problema se puede representar con dos tablas:

- D(i,j), que contendrá la distancia mínima entre 'i' y 'j'.
- P(i,j) = 'k', que representa que 'k' es un nodo intermedio en el camino entre 'i' y 'j'. Por tanto, para recuperar el camino tendremos que calcular también P(i,k) y P(k,j).

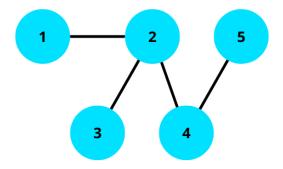
Como ejemplo vamos a realizar el proporcionado en la guía de esta práctica:

 Traducir del latín al arameo disponiendo de los siguientes diccionarios: latín-griego, griego-etrusco, etrusco-demótico, griego-demótico, demótico-arameo (Solución: sí es posible la traducción: latín-griego-demótico-arameo, realizando tres traducciones).



Lista de idiomas y enumeración para representación en grafo:

- Latín, 1.
- Griego, 2.
- Etrusco, 3.
- Demótico, 4.
- Arameo, 5.



$D_0$	1	2	3	4	5
1	0	1	8	8	∞
2	1	0	1	1	∞
3	8	1	0	1	8
4	8	1	1	0	1
5	8	8	8	1	0

D <sub>1</sub>	1	2	3	4	5
1	0	1	8	8	∞
2	1	0	1	1	∞
3	8	1	0	1	∞

4	∞	1	1	0	1
5	∞	8	8	1	0

$D_2$	1	2	3	4	5
1	0	1	2	2	8
2	1	0	1	1	∞
3	2	1	0	1	∞
4	2	1	1	0	1
5	∞	8	8	1	0

$D_3$	1	2	3	4	5
1	0	1	2	2	8
2	1	0	1	1	8
3	2	1	0	1	∞
4	2	1	1	0	1
5	8	8	8	1	0

\_

D <sub>4</sub>	1	2	3	4	5
1	0	1	2	2	3
2	1	0	1	1	2
3	2	1	0	1	2
4	2	1	1	0	1
5	3	2	2	1	0

D <sub>5</sub> (Solución)	1	2	3	4	5
1	0	1	2	2	3 (traducciones)
2	1	0	1	1	2
3	2	1	0	1	2
4	2	1	1	0	1
5	3	2	2	1	0

La tabla P(i,j) = 'k', siendo 'k' el nodo intermedio entre 'i' y 'j', quedaría así:

P (Solución)	1	2	3	4	5
1	0	0	2	2	4
2	0	0	0	0	4
3	2	0	0	0	4
4	2	0	0	0	0
5	2	4	4	0	0

Nos piden traducir de latín (1) a arameo (5).

La solución proporcionada en la guía es: latín - griego - demótico - arameo.

#### 1.3 Verificación del P.O.B.

 $D_0[i][j]$  es óptimo, porque el mejor camino de 'i' a 'j' sin pasar por ningún nodo es D[i][j].  $D_k[i][j]$  es óptimo: en caso contrario, habría otros nodos en el camino de 'i' a 'j' pasando por  $\{1...k-1\}$  tal que su coste sea menor que el considerado. Esto es imposible, dado que la ecuación recurrente siempre selecciona el menor coste.

# 1.4 Diseño del algoritmo de cálculo de coste óptimo.

 $ALGORITMO\ D\ (V = MatrAdy[1...N][1...N])$ 

*D*← matriz de N filas y columnas indexadas {1...N}

*P*← matriz de N filas y columnas indexadas {1...N}

Para cada fila i en {1...N}, hacer:

Para cada columna j en {1...N}, hacer:

D[i][j] = MatrAdy[i][j] // Distancia directa desde 'i' a 'j'.

P[i][j] = 0 // De 'i' a 'j' no se pasa por ningún nodo inicialmente.

Para cada k en {1...N}, hacer:

```
Para cada fila i en \{1...N\}, hacer:

Para cada columna j en \{1...N\}, hacer:

Si i es igual a j, hacer:

T(i,j) = 0

Fin-Si

Si D[i][j] > D[i][k] + D[k][j], entonces:

D[i][j] = D[i][k] + D[k][j]

P[i][j] = k

Fin-Si

Fin-Para

Fin-Para

Fin-Para

Devolver D, P
```

# 1.5 Diseño del algoritmo de recuperación de la solución.

ALGORITMO S = RecuperaSolucion (i, j, P(1...N, 1...N) : Tabla P resultante del algoritmo MatrAdy)

```
S \leftarrow \emptyset

k \leftarrow N

Mientras\ k <> 0,\ hacer:

k \leftarrow P(i,j)

j \leftarrow k

A \tilde{n} a dir\ 'k' a\ S

Fin-Mientras

Devolver\ S
```

#### 2. Ejercicio 2

# 2.1 Diseño de resolución por etapas y ecuación recurrente

Resolución del problema por etapas:

El problema se puede resolver por etapas. En cada etapa se considera pasar por una arista (salto) intermedio 'k' para cada par de nodos (casillas) de origen y destino.

#### - Ecuación recurrente:

Para este problema podemos interpretar cada casilla como un nodo de un grafo (dirigido) y los caminos como aristas que unen nodos (casillas). Consideramos que todas las aristas tienen peso nulo (0). Asumimos también que 'C' es la matriz de adyacencia del grafo, a su vez qué la matriz es numerada en cada nodo, empezando por el nodo de inicio, y numerando por columnas hasta el nodo fin y C[i][j] es la suma de todo el oro por la mejor ruta para ir desde el nodo 'i" hasta 'j'.  $C[i][j] = -\infty$  si no hay camino entre 'i' y 'j'. Llamaremos  $C_k[i][j]$  a la suma del oro del camino que hace máxima la suma entre 'i' y 'j', con nodos intermedios en el conjunto  $\{1...k\}$ . Si 'k' = 0, no hay nodos intermedios.

El caso base de nuestra ecuación recurrente sería  $C_0[i][j] = C[i][j]$ , es decir, no hay ningún nodo intermedio entre 'i' y 'j'.

El caso general tiene en cuenta dos posibilidades pudiendo pasar por los nodos 1 a 'k':

- Que el camino de 'i' a 'j' no pase por 'k', C<sub>k-1</sub>[i][j].
- Que el camino de 'i' a 'j' pase por 'k',  $C_{k-1}[i][k] + C_{k-1}[k][j]$ .

- k como maximo puede valer  $|\sqrt{(N^2+M^2)}|$  //redondeado hacia abajo.
- Teniendo en cuenta que L∈[V<sub>i</sub>, V<sub>i-1</sub>]
- Siendo V<sub>i</sub> los vértices, nodos o posiciones en 'i'.
- L en cada caso solo puede optar a tres valores como máximo y a uno como mínimo, siendo L=i+m ó L=i+1 (siendo m el número de filas de la matriz rectangular bidimensional), las opciones mínimas; y L=i+m+1 el caso al que opta en su máximo de valores.

Por tanto, nos quedaría:

$$C_{k}[i][j] = max\{ C_{k-1}[i][j], c[i][L] + C_{k-1}[L][j] \}$$

- Función objetivo:

La función objetivo es maximizar  $C_n[i][j]$ , con 'n' el número de nodos del grafo, para todo 'i' y todo 'j'.

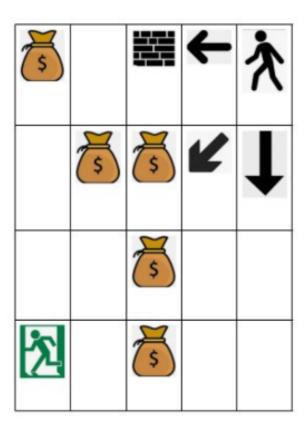
#### 2.2 Diseño de la memoria

La solución al problema se puede representar con dos tablas:

- C(i,j), que contendrá la cantidad de oro máxima por el mejor camino entre 'i' y 'j'.
- P(i,j) = 'L', que representa que 'L' es un nodo intermedio en el camino entre 'i' y 'j'.
   Por tanto, para recuperar el camino tendremos que calcular también P(i,L) y P(L,j).

Como ejemplo vamos a realizar el proporcionado en la guía de esta práctica:

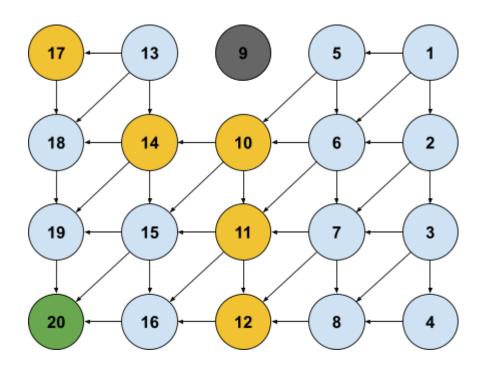
- Llegar a la salida pudiendo recoger tanto oro como sea posible con el siguiente mapa



Matriz de Adyacencia

С	1	2	3	4	5
1	1	0	-∞	0	Ι
2	0	1	1	0	0
3	0	0	1	0	0
4	s	0	1	0	0

- Numeración de los nodos:



El triángulo inferior de la tabla no se va a rellenar en ningún caso 'k' ya que el grafo es dirigido.

C <sub>0</sub>	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0	0	-8	-8	0	0	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-∞
2	-8	0	0	-8	-8	0	0	8	-8	-8	-8	8	-8	-8	8	8	-8	-8	8	-∞
3	-8	-8	0	0	8	-8	0	0	8	8	-8	-8	8	8	-8	-8	8	-8	-8	_∞
4	-8	-8	-8	0	-8	-8	-8	0	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	_∞
5	-8	-8	-8	-8	0	0	-8	-8	-8	1	-8	-8	-8	-8	-8	-8	8	-8	-8	-∞
6	-8	8	-8	-8	-8	0	0	8	-8	1	1	8	-8	-8	8	8	-8	-8	8	-∞
7	-8	-8	-8	-8	-8	-8	0	0	-8	-8	1	1	-8	-8	-8	-8	-8	-8	-8	-∞
8	-8	-8	-8	-8	-8	-8	-8	0	-8	-8	-8	1	-8	-8	-8	-8	-8	-8	-8	_∞

9	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8
10	-8	-8	-8	-8	-8	-8	-8	-8	-8	1	2	-8	-8	2	1	-8	-8	-8	-8	-8
11	8	8	-8	8	-8	8	8	-8	8	8	1	2	-8	8	1	1	8	-8	-8	-8
12	-8	-8	-8	-8	-8	-8	-8	-8	-8	8	-8	1	-8	8	-8	1	8	-8	-8	-8
13	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	0	1	-8	-8	1	0	-8	-8
14	-8	-8	-∞	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	1	1	-8	8	1	1	-∞
15	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	0	0	-8	-8	0	0
16	-8	-8	-∞	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	0	-8	-8	-8	0
17	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	8	-8	-8	1	1	-8	-∞
18	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	0	0	-80
19	-8	-8	-8	-8	-8	-8	-8	-8	-8	8	-8	-8	-8	8	-8	-8	8	8	0	0
20	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	8	8	8	8	0

C <sub>1</sub>	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0	0	0	-8	0	0	0	-8	-∞	1	1	-∞	-8	-8	-8	-8	-∞	-∞	-8	-∞
2	-8	0	0	0	-8	0	0	0	-∞	1	1	1	-8	-8	-8	-8	-8	-∞	-8	-8
3	-8	-8	0	0	-8	-8	0	0	-8	-8	1	1	-8	-8	-8	-8	-∞	-8	-8	-∞
4	-8	-8	-8	0	-8	-8	-8	0	-8	-8	-8	1	-8	-8	-8	-8	-80	-∞	-8	-∞
5	-8	-8	-8	-8	0	0	0	-8	-8	1	2	-∞	-8	2	1	-8	-8	-∞	-8	-∞
6	-8	8	-8	-8	8	0	0	0	-8	1	2	2	8	2	1	1	-8	-8	8	-∞
7	-8	-8	-8	8	-8	-8	0	0	-8	-8	1	2	-8	-8	1	1	-8	-8	-8	-∞
8	-8	-8	-8	-8	-8	-8	-8	0	-8	-8	-8	1	-8	-8	-8	1	-8	-8	8	-∞
9	-8	-8	-8	-8	-8	-8	-8	-8	-∞	-8	-8	-∞	-8	-8	-8	8	-8	-∞	8	_∞
10	-8	-8	-8	-8	-8	-8	-8	-8	-∞	1	2	3	-8	2	2	2	-8	2	2	1
11	-8	-8	-8	-8	-8	-8	-8	-8	-∞	-8	1	2	-8	-8	1	2	-∞	-∞	1	1
12	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	1	-8	-8	-8	1	-8	-8	-8	1
13	-8	-8	-8	-8	-8	-8	-8	-8	-∞	-8	-8	-∞	0	1	1	-8	1	1	1	-∞
14	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	1	1	1	-∞	1	1	1

15	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	0	0	-8	-8	0	0
16	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	0	-8	-8	-8	0
17	-8	-8	-8	8	8	8	8	-8	8	-8	-8	8	8	8	8	8	1	1	1	-80
18	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	0	0	0
19	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	0	0
20	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	8	-8	-8	0

	1	2	3	1	<b>-</b>	6	7	8	٥	10	11	10	12	14	15	16	17	10	10	20
C <sub>2</sub>	1	2	3	4	5	б	1	ð	9	10	11	12	13	14	15	16	17	18	19	20
1	0	0	0	-∞	0	0	0	-8	-∞	1	2	2	-∞	2	1	1	_∞	_∞	_∞	-∞
2	_∞	0	0	0	-∞	0	0	0	_∞	1	2	2	_∞	2	1	1	_∞	_∞	_∞	_∞
3	-∞	-&	0	0	-8	-8	0	0	-8	-8	1	1	-∞	-∞	1	1	-∞	_∞	-∞	-∞
4	-8	-8	-8	0	-8	-8	8	0	-8	-8	-∞	1	-∞	-∞	-8	1	-∞	-∞	-∞	-∞
5	8	8	8	8	0	0	0	0	-∞	1	2	-∞	-∞	2	2	2	-∞	2	2	1
6	-8	-8	-8	-8	-8	0	0	0	-8	1	2	3	-∞	2	2	1	-∞	2	2	1
7	-8	-8	-8	-8	-8	-8	0	0	-∞	-∞	1	2	-∞	-∞	1	2	-∞	_∞	1	1
8	-∞	-∞	-∞	-∞	-8	-8	-8	0	-∞	-∞	-∞	1	-∞	-∞	-8	1	-∞	_∞	-∞	1
9	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-∞	-8	-∞	-∞	-8	-∞	_∞	_∞	-∞	_∞
10	8	8	8	8	8	8	-8	-8	-∞	1	2	3	-∞	2	2	2	-∞	2	2	2
11	8	8	8	8	8	8	-8	-8	-8	-8	1	2	-∞	-∞	1	2	-∞	_∞	1	2
12	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-∞	1	-∞	-∞	-8	1	-∞	_∞	-∞	1
13	8	8	8	8	8	8	-8	-8	-∞	-∞	-∞	-8	0	1	1	1	1	1	1	1
14	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-∞	-8	-∞	1	1	1	-∞	1	1	1
15	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-∞	-∞	-∞	-∞	0	0	-∞	-∞	0	0
16	-∞	-∞	-∞	-∞	-8	-8	-∞	-8	-∞	-∞	-∞	-∞	-∞	-∞	-8	0	-∞	_∞	-∞	0
17	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-∞	-∞	-∞	-∞	-8	-∞	1	1	1	1
18	-∞	-∞	-∞	-∞	-8	-8	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	0	0	0
19	-8	-8	-8	-8	-8	-8	-8	-8	-∞	-∞	-∞	-∞	-∞	-∞	-8	-∞	_∞	_∞	0	0
20	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-∞	-8	-∞	-∞	-8	-∞	-∞	_∞	-∞	0

C <sub>3</sub>	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0	0	0	-8	0	0	0	0	-8	1	2	3	-8	2	2	2	-8	2	2	1
2	8	0	0	0	8	0	0	0	-8	1	2	3	-8	2	2	2	-8	2	2	1
3	-&	-8	0	0	-8	-8	0	0	-8	-8	1	1	-8	-8	1	2	-∞	-∞	1	1
4	-∞	-8	-∞	0	-8	-8	-8	0	-8	-8	-8	1	-8	-8	-8	1	-8	-∞	-∞	1
5	-&	-∞	-∞	-8	0	0	0	0	-∞	1	2	3	-8	2	2	3	-8	2	2	2
6	-∞	-∞	-∞	-∞	-∞	0	0	0	-∞	1	2	3	-∞	2	2	3	-∞	2	2	2
7	-8	-8	-8	-8	-8	-8	0	0	-8	-8	1	2	-8	-8	1	2	-8	-∞	1	2
8	-8	-8	-8	-8	-8	-8	-8	0	-8	-8	-8	1	-8	-8	-8	1	-8	-∞	-∞	1
9	-∞	-∞	-∞	-∞	-∞	-∞	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-∞	-8	-8	_∞
10	-8	-8	-8	-8	-8	-8	-8	-8	-8	1	2	3	-8	2	2	2	-8	2	2	3
11	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	1	2	-8	8	1	2	-8	-8	1	2
12	-∞	-∞	-∞	-∞	-∞	-∞	-8	-8	-8	-8	-8	1	-8	-8	-8	1	-8	-∞	-∞	1
13	-∞	-8	-∞	-8	-8	-8	-8	-8	-∞	-8	-8	-8	0	1	1	1	1	1	1	1
14	-8	-8	-8	-8	-8	-8	-8	-8	-∞	-8	-8	-∞	-8	1	1	1	-∞	1	1	1
15	-∞	-8	-∞	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	0	0	-8	-8	0	0
16	-8	-8	-∞	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	0	-8	_∞	_∞	0
17	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	1	1	1	1
18	-8	-8	-8	-8	-8	-8	-8	-8	-∞	-8	-8	-∞	-8	-8	-∞	-8	-8	0	0	0
19	-&	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	8	-8	8	-8	-∞	0	0
20	-8	-8	-8	-8	-8	-8	-8	-8	-∞	-8	-8	-∞	-8	-8	-∞	-8	_∞	_∞	_∞	0

C <sub>4</sub>	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0	0	0	-8	0	0	0	0	-∞	1	2	3	-8	2	2	3	-8	2	2	2
2	-8	0	0	0	-8	0	0	0	-∞	1	2	3	-8	2	2	3	-8	2	2	2
3	-8	-∞	0	0	-8	-8	0	0	_∞	_∞	1	1	-∞	-8	1	2	-∞	-8	1	2
4	-8	-8	-8	0	-8	-8	-8	0	-∞	-∞	-8	1	-8	8	8	1	-8	8	-∞	1
5	-8	-8	-8	-8	0	0	0	0	-∞	1	2	3	-8	2	2	3	-8	2	2	3
6	-8	-∞	-8	-8	-8	0	0	0	-∞	1	2	3	-∞	2	2	3	-8	2	2	3
7	-8	-∞	-∞	-8	-∞	-∞	0	0	-∞	-∞	1	2	-∞	-8	1	2	-∞	-8	1	2
8	-8	-∞	-8	-8	-∞	-∞	-8	0	-∞	-∞	-∞	1	-∞	-8	-8	1	-8	-8	_∞	1
9	-8	-∞	-8	-8	-8	-8	-8	-8	-∞	-∞	-8	-8	-∞	-8	-8	-8	-8	-8	-∞	_∞
10	-∞	_∞	-∞	-∞	_∞	_∞	-∞	_∞	_∞	1	2	3	_∞	2	2	2	_∞	2	2	3
11	-8	-∞	-∞	-8	-8	-8	-8	-∞	_∞	_∞	1	2	-8	-8	1	2	-8	-8	1	2
12	-8	_∞	-∞	-8	-∞	-∞	-8	-∞	_∞	_∞	_∞	1	_∞	-8	-8	1	-∞	-8	_∞	1
13	-∞	_∞	-∞	-∞	-∞	-∞	-∞	-∞	_∞	_∞	_∞	-∞	0	1	1	1	1	1	1	1
14	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	1	1	1	-∞	1	1	1
15	-8	-∞	-∞	-8	-∞	-∞	-8	-∞	-∞	-∞	-∞	-∞	-∞	-8	0	0	-∞	-8	0	0
16	-8	-∞	-∞	-8	-∞	-∞	-8	-∞	-∞	-∞	-∞	-8	-∞	-8	-8	0	-∞	-8	-∞	0
17	-8	_∞	-8	-8	-∞	-∞	-8	-∞	-∞	-∞	-∞	-8	-∞	-8	-8	-8	1	1	1	1
18	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	0	0	0
19	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	0	0
20	-∞	_∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	_∞	0

C <sub>5</sub>	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0	0	0	-8	0	0	0	0	-∞	1	2	3	-8	2	2	3	-8	2	2	3
2	-8	0	0	0	-8	0	0	0	-∞	1	2	3	-8	2	2	3	-8	2	2	3
3	-8	-∞	0	0	-8	-8	0	0	_∞	-∞	1	1	-∞	-8	1	2	-∞	-8	1	2
4	-8	-∞	-8	0	-8	-8	-8	0	_∞	-∞	-∞	1	-∞	-8	-8	1	-∞	-8	_∞	1
5	8	-8	-8	8	0	0	0	0	-∞	1	2	3	-8	2	2	3	-8	2	2	3
6	-8	-∞	-8	-8	-8	0	0	0	-∞	1	2	3	-∞	2	2	3	-8	2	2	3
7	-8	-∞	-∞	-8	-∞	-∞	0	0	-∞	-∞	1	2	-∞	-8	1	2	-∞	-8	1	2
8	-8	-∞	-8	-8	-8	-∞	-8	0	-∞	-∞	-∞	1	-∞	-8	-8	1	-8	-8	_∞	1
9	-8	-∞	-8	-8	-8	-8	-8	-8	-∞	-∞	-8	-8	-∞	-8	-8	-8	-8	-8	-∞	_∞
10	-∞	_∞	-∞	-∞	_∞	_∞	-∞	_∞	_∞	1	2	3	_∞	2	2	2	_∞	2	2	3
11	-8	-∞	-∞	-8	-8	-8	-8	-∞	_∞	-∞	1	2	-8	-8	1	2	-8	-8	1	2
12	-8	_∞	-∞	-8	-∞	-∞	-8	-∞	_∞	-∞	_∞	1	_∞	-8	-8	1	-∞	-8	_∞	1
13	-8	-∞	-8	-8	-∞	-∞	-8	-∞	-∞	-∞	-∞	-8	0	1	1	1	1	1	1	1
14	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	1	1	1	-∞	1	1	1
15	-8	-∞	-∞	-8	-∞	-∞	-8	-∞	-∞	-∞	-∞	-8	-∞	-8	0	0	-∞	-8	0	0
16	-8	-∞	-∞	-8	-∞	-∞	-8	-∞	-∞	-∞	-∞	-8	-∞	-8	-8	0	-∞	-8	-∞	0
17	-8	-∞	-∞	-8	-∞	-∞	-8	-∞	-∞	-∞	-∞	-8	-∞	-8	-8	-8	1	1	1	1
18	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-8	-8	-∞	0	0	0
19	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-8	-8	-∞	-∞	0	0
20	-∞	_∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	_∞	0

C <sub>6</sub>	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0	0	0	-8	0	0	0	0	-∞	1	2	3	-8	2	2	3	-8	2	2	3
2	-8	0	0	0	-8	0	0	0	-∞	1	2	3	-8	2	2	3	-∞	2	2	3
3	-8	-8	0	0	-8	-8	0	0	-∞	-8	1	1	-8	-8	1	2	-8	-8	1	2
4	-8	-8	-8	0	-8	-8	-8	0	-∞	-8	-8	1	-8	8	-8	1	-8	8	-∞	1
5	8	-8	-8	8	0	0	0	0	-∞	1	2	3	8	2	2	3	-8	2	2	3
6	-8	-∞	-8	-8	-8	0	0	0	-∞	1	2	3	-8	2	2	3	-8	2	2	3
7	-8	-8	-∞	-8	-8	-∞	0	0	-∞	-∞	1	2	-8	-8	1	2	-∞	-8	1	2
8	-8	-∞	-8	-8	-8	-∞	-8	0	-∞	-∞	-∞	1	-8	-8	-8	1	-8	-8	_∞	1
9	-8	-∞	-8	-8	-8	-8	-8	-8	-∞	-∞	-8	-8	-8	-8	-8	-8	-8	-8	-∞	_∞
10	-∞	_∞	-∞	-∞	-∞	_∞	-∞	_∞	_∞	1	2	3	-∞	2	2	2	_∞	2	2	3
11	-8	-∞	-∞	-8	-8	-8	-8	-∞	_∞	-∞	1	2	-8	-8	1	2	-8	-8	1	2
12	-8	_∞	-∞	-8	-8	-∞	-8	-∞	_∞	-∞	_∞	1	-8	-8	-8	1	-∞	-8	_∞	1
13	-∞	_∞	-∞	-∞	-∞	-∞	-∞	-∞	_∞	_∞	_∞	-∞	0	1	1	1	1	1	1	1
14	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	1	1	1	-∞	1	1	1
15	-8	-∞	-∞	-8	-8	-∞	-8	-∞	-∞	-∞	-∞	-∞	-∞	-8	0	0	-∞	-8	0	0
16	-8	-∞	-∞	-8	-8	-∞	-8	-∞	-∞	-∞	-∞	-8	-8	-8	-8	0	-∞	-8	-∞	0
17	-8	_∞	-8	-8	-8	-∞	-8	-∞	-∞	-∞	-∞	-8	-8	-8	-8	-8	1	1	1	1
18	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-8	-8	-∞	0	0	0
19	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-8	-8	-∞	-∞	0	0
20	-∞	_∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	_∞	0

La tabla P(i,j) = 'L', siendo 'L' el nodo intermedio entre 'i' y 'j', quedaría así:

P(Sol)	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0	1	2	-8	1	1	6	7	-8	6	10	11	-8	10	11	12	-8	14	14	16
2	-8	0	2	-8	1	1	6	7	-8	6	10	11	-8	10	11	12	-8	14	14	16
3	-8	-8	0	3	-8	-8	3	3	-8	-8	7	11	-8	-8	11	12	-8	-8	15	16
4	-8	-8	-8	0	-8	-8	-8	4	-8	-8	-8	8	-8	-8	-8	12	-8	-8	-8	16
5	-8	-∞	-∞	-∞	0	5	6	7	-∞	5	10	11	_∞	10	11	12	-∞	14	15	16
6	-8	_∞	_∞	_∞	_∞	0	6	7	_∞	6	10	11	_∞	10	11	12	-∞	14	15	16
7	-8	-8	-8	-8	-∞	-8	0	7	-8	-8	7	11	-∞	-8	11	12	-8	-8	15	16
8	-8	-8	-8	-8	-∞	-8	-∞	0	-∞	-∞	-8	8	-∞	-8	-8	12	-8	-8	-8	16
9	-8	-8	-8	-8	-∞	-8	-8	-8	-8	-8	-8	-8	-∞	-8	-8	-8	-8	-8	-8	-∞
10	-∞	-8	-8	-8	-∞	-∞	-8	-8	-8	0	10	11	-∞	10	11	12	-8	14	14	16
11	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	0	11	-∞	-8	11	12	-8	-8	15	16
12	-8	-8	-8	-8	-∞	-8	-8	-8	-8	-8	-8	0	-∞	-8	-∞	12	-8	-8	-8	16
13	-8	-8	-8	-8	_∞	-8	-8	-∞	-8	-8	-8	-8	0	13	14	15	13	14	14	15
14	-8	-8	-8	-8	-∞	-∞	-8	-8	-8	-8	-8	-8	-∞	0	14	15	-8	14	14	15
15	-8	-8	-8	-8	-∞	-8	-8	-8	-8	-8	-8	-8	-∞	-8	0	15	-8	-8	15	15
16	-8	-∞	-∞	-∞	-∞	-8	-∞	-∞	-∞	-∞	-∞	-∞	_∞	-8	-∞	0	-∞	-8	-8	16
17	-8	-∞	-∞	-∞	_∞	-8	-∞	-∞	-∞	-∞	-8	-∞	_∞	-8	-∞	-8	0	17	18	19
18	-8	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	0	18	19
19	-∞	-∞	-∞	-∞	-∞	-8	-∞	-∞	-∞	-∞	-8	-∞	-∞	-8	-8	-∞	-∞	-8	0	19
20	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	_∞	-∞	_∞	_∞	-∞	-∞	-∞	-∞	-∞	0

Nos piden llegar a la salida recogiendo tanto oro como sea posible. La solución del ejemplo proporcionado en la guía es: 1-6-10-11-12-16-20.

# 2.3 Verificación del P.O.B.

 $C_0[i][j]$  es óptimo, porque el mejor camino de 'i' a 'j' sin pasar por ningún nodo es C[i][j].  $C_k[i][j]$  es óptimo: en caso contrario, habría otros nodos en el camino de 'i' a 'j' pasando por  $\{1...k-1\}$  tal que su cantidad de oro sea mayor que el considerado. Esto es imposible, dado que la ecuación recurrente siempre selecciona la mayor cantidad.

# 2.4 Diseño del algoritmo de cálculo de coste óptimo.

```
Procedimiento Floyd(MatrizAdy[1..n][1..m])
Para i=1 hasta n, hacer:
       Para j=1 hasta m, hacer:
              C[i][j]= MatrizAdy[i][i]
              P[i][i]= 0
       Fin-Para
Fin-Para
Para i=1 hasta n, hacer:
       Para j=1 hasta m, hacer:
              L1=i+1
              L2=i+m
              L3=i+m+1
              CM \leftarrow max(C[i][L1]+C[L1][j], C[i][L2]+C[L2][j], C[i][L3]+C[L3][j])
              Si C[i][j] < CM, entonces:
                      Si C[i][L1]+C[L1][j] = CM, entonces:
                             C[i][j] = C[i][L1] + C[L1][j]
                             P[i][j]= L1
                      En Otro Caso Si C[i][L2]+C[L2][j] = CM, entonces:
                             C[i][j] = C[i][L2] + C[L2][j]
                             P[i][j]= L2
                      En Otro Caso, entonces:
                             C[i][j] = C[i][L3] + C[L3][j]
                              P[i][i]= L3
                      Fin-Si
              Fin-Si
       Fin-Para
Fin-Para
Devolver C,P
       2.5 Diseño del algoritmo de recuperación de la solución.
ALGORITMO S = RecuperaSolucion (i, j, P(1...N, 1...M)): Tabla P resultante del algoritmo
MatrAdy)
       S←∅
       k← N*M
       Mientras k <> 0, hacer:
              k←P(i,j)
              j←k
              Añadir 'k' a S
       Fin-Mientras
       Devolver S
```

# 3. Ejercicio 3

# 3.1 Diseño de resolución por etapas y ecuación recurrente

- Resolución del problema por etapas:

El problema se puede resolver por etapas. En cada etapa se considera pasar por una arista (salto) intermedio 'k' para cada par de nodos (casillas) de origen y destino.

#### - Ecuación recurrente:

Para este problema podemos interpretar cada casilla como un nodo de un grafo (dirigido) y los caminos como aristas que unen nodos (casillas). Consideramos que todas las aristas tienen peso nulo (0). Asumimos también que 'S' es la matriz de adyacencia del grafo, a su vez qué la matriz es numerada en cada nodo, empezando por el nodo de inicio, y numerando por columnas hasta el nodo fin y S[i][j] es la suma de la batería gastada por la mejor ruta para ir desde el nodo 'i' hasta 'j'. S[i][j] = - $\infty$  si no hay camino entre 'i' y 'j'. Llamaremos  $S_k[i][j]$  a la suma de la batería del camino que hace mínima el gasto de energía entre 'i' y 'j', con nodos intermedios en el conjunto  $\{1...k\}$ . Si 'k' = 0, no hay nodos intermedios.

El caso base de nuestra ecuación recurrente sería  $S_0[i][j] = S[i][j]$ , es decir, no hay ningún nodo intermedio entre 'i' y 'j'.

El caso general tiene en cuenta dos posibilidades pudiendo pasar por los nodos 1 a 'k':

- Que el camino de 'i' a 'j' no pase por 'k', S<sub>k-1</sub>[i][j].
- Que el camino de 'i' a 'j' pase por 'k',  $S_{k-1}[i][j] + S_{k-1}[i][j]$ .
- k como maximo puede valer |√(N²+M²)| //redondeado hacia abajo.
- Teniendo en cuenta que L∈[ V<sub>i</sub> , V<sub>i-1</sub> ]
- Siendo V<sub>i</sub> los vértices, nodos o posiciones en 'i'.
- L en cada caso solo puede optar a tres valores como mínimo y a uno como máximo, siendo L=i+m ó L=i+1 (siendo m el número de filas de la matriz rectangular bidimensional), las opciones mínimas; y L=i+m+1 el caso al que opta en su máximo de valores.

Por tanto, nos quedaría:

$$S_{k}[i][j] = min\{ S_{k-1}[i][j], S[i][L] + S_{k-1}[L][j] \}$$

#### - Función objetivo:

La función objetivo es minimizar  $S_n[i][j]$ , con 'n' el número de nodos del grafo, para todo 'i' y todo 'j'.

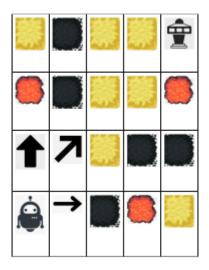
#### 3.2 Diseño de la memoria

La solución al problema se puede representar con dos tablas:

- S(f,c), que contendrá la cantidad de batería mínima por el mejor camino entre 'i' y 'j'.
- P(i,j) = 'L', que representa que 'L' es un nodo intermedio en el camino entre 'i' y 'j'.
- Por tanto, para recuperar el camino tendremos que calcular también P(i,L) y P(L,j).

Como ejemplo vamos a realizar el proporcionado en la guía de esta práctica:

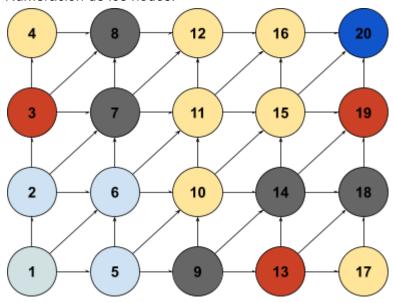
- Llegar a la base gastando la mínima batería posible con el siguiente mapa:



- Matriz de adyacencia

S	1	2	3	4	5
1	1	2	1	1	S
2	+∞	2	1	1	+∞
3	0	0	1	2	2
4	I	0	2	+∞	1

- Numeración de los nodos:



- He aquí un ejemplo de las posibles soluciones más eficientes para los valores siguientes:

- Suelo rojo: gasto = +∞

Suelo Amarillo: gasto = 1

- Suelo azul: gasto = 0

- Suelo Gris: gasto = 2

						o. gac														_
S <sub>0</sub>	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0	0	+∞	+∞	0	0	+∞	+∞	+∞	+∞	+∞	+8	+∞	+∞	+8	+∞	+∞	+∞	+∞	+∞
2	+∞	0	+ 8	+∞	+∞	0	2	+ 8	+∞	+ 8	+8	+ 8	+8	+∞	+ 8	+8	+∞	+∞	+∞	+∞
3	+∞	+∞	0	1	+∞	+∞	2	2	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞
4	+∞	+∞	+∞	0	+∞	+∞	+∞	2	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞
5	+8	+8	+∞	+8	0	0	+8	+∞	2	1	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞
6	+∞	+8	+∞	+8	+∞	0	2	+∞	+∞	1	1	+∞	+∞	+∞	+8	+∞	+∞	+∞	+∞	+∞
7	+∞	+∞	+∞	+8	+∞	+∞	0	2	+∞	+∞	1	1	+∞	+∞	+8	+∞	+∞	+∞	+∞	+∞
8	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	+∞	+∞	+∞	1	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞
9	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	1	+∞	+∞	+∞	2	+∞	+∞	+∞	+∞	+∞	+∞
10	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	1	+∞		2	1	+∞	+∞	+∞	+∞	+∞
11	+∞	+∞	+∞	+∞	+∞	+∞	+8	+∞	+∞	+∞	0	1	+∞	+∞	1	1	+∞	+∞	+∞	+∞
12	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	+∞	+∞	+∞	1	+∞	+∞	+∞	+∞
13	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	2	+∞	+∞	1	2	+∞	+∞
14	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	1	+∞	+∞	2	+∞	+∞
15	+8	+8	+∞	+8	+∞	+8	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	1	+∞	+∞	+∞	0
16	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	+∞	+∞	+∞	0
17	+8	+8	+∞	+8	+∞	+∞	+8	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	2	+∞	+∞
18	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	+∞	+∞
19	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	0
20	+∞	+∞	+∞	+8	+∞	+∞	+8	+∞	+∞	+∞	+∞	+8	+∞	+∞	+8	+∞	+8	+∞	+∞	0

S <sub>1</sub>	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0	0	+∞	+∞	0	0	2	+8	2	1	1	+8	+∞	+8	+8	+8	+8	+8	+∞	+∞
2	+∞	0	+8	*	+8	0	2	4	+∞	1	1	2	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞
3	+8	*	0	1	*8	*	2	2	+∞	+8	3	3	+∞	+8	+8	+8	+∞	+∞	+∞	+∞
4	+∞	+∞	+∞	0	+∞	+8	+8	2	+8	+8	+8	3	+∞	+8	+8	+8	+8	+8	+∞	+∞
5	+∞	+8	+8	+8	0	0	2	+ 8	2	1	2	+ 8	+∞	3	2	+ 8	+8	+∞	+∞	+∞
6	+∞	+∞	+∞	+∞	+∞	0	2	4	+∞	1	1	2	+∞	3	2	2	+∞	+∞	+∞	+∞
7	+∞	+8	+8	+8	+8	+8	0	2	+∞	+8	1	1	+∞	+8	2	2	+8	+∞	+∞	+∞
8	+∞	+8	+8	+8	+8	+8	+8	0	+∞	+8	+8	1	+∞	+8	+8	2	+∞	+∞	+∞	+∞
9	+∞	+8	+∞	+8	+8	+8	+8	+8	0	1	2	+8	+∞	2	2	+8	+∞	4	+∞	+∞
10	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+8	+∞	0	1	2	+∞	2	1	2	+∞	4	+∞	2
11	+∞	+8	+8	+8	+8	+8	+8	+8	+∞	+8	0	1	+∞	+8	1	1	+8	+∞	+∞	2
12	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+8	+∞	+8	+∞	0	+∞	+8	+8	1	+∞	+∞	+∞	2
13	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	2	3	+8	1	2	+∞	+∞
14	+8	+8	+8	+8	+8	+8	+8	+8	+8	+8	+8	+8	+∞	0	1	2	+8	2	+∞	2
15	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+8	+∞	+8	+∞	+8	+∞	+8	0	1	+8	+∞	+∞	0
16	+∞	+∞	+∞	+8	+∞	+∞	+∞	+8	+∞	+8	+∞	+8	+∞	+8	+8	0	+∞	+∞	+∞	0
17	+∞	+8	+∞	+8	+8	+8	+8	+8	+∞	+8	+∞	+8	+∞	+8	+8	+8	0	2	+∞	+∞
18	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+8	+∞	+8	+∞	+8	+∞	+8	+8	+8	+∞	0	+∞	+∞
19	+8	+8	+∞	+8	+8	+8	+8	+8	+8	+8	+8	+8	+∞	+8	+8	+8	+8	+8	0	0
20	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+8	+∞	+8	+∞	+8	+∞	+8	+8	+8	+∞	+8	+∞	0

S <sub>2</sub>	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0	0	+8	+8	0	0	2	4	2	1	1	2	+∞	3	2	2	+∞	+∞	+∞	+∞
2	+∞	0	+∞	+8	+∞	0	2	4	+∞	1	1	2	+∞	3	2	2	+∞	+∞	+∞	+∞
3	+8	+∞	0	1	+8	+∞	2	2	+∞	+∞	3	3	+∞	+∞	4	4	+∞	+∞	+∞	+∞
4	+8	+8	+8	0	+8	+8	+8	2	+∞	+∞	+∞	3	+∞	+8	+8	4	+8	+∞	+∞	+∞
5	+8	+8	+8	+ 8	0	0	2	4	2	1	2	2	+∞	3	2	2	+∞	5	+∞	2
6	+8	+∞	+8	+8	+8	0	2	4	+∞	1	1	2	+∞	3	2	2	+∞	5	+∞	2
7	+∞	+∞	+∞	+∞	+∞	+∞	0	2	+∞	+∞	1	1	+∞	+∞	2	2	+∞	+∞	+∞	2
8	+∞	+∞	+∞	+8	+8	+∞	+8	0	+∞	+∞	+∞	1	+∞	+8	+8	2	+∞	+∞	+∞	2
9	+8	+∞	+8	+8	+8	+8	+8	+∞	0	1	2	3	+∞	2	2	3	+∞	4	+∞	2
10	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	1	2	+∞	2	1	2	+∞	4	+∞	2
11	+∞	+∞	+8	+∞	+∞	+∞	+8	+∞	+∞	+∞	0	1	+∞	+8	1	1	+∞	+∞	+∞	2
12	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	+∞	+∞	+∞	1	+∞	+∞	+∞	2
13	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	2	3	4	1	2	+∞	3
14	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	1	2	+∞	2	+∞	2
15	+∞	+∞	+8	+∞	+∞	+8	+8	+∞	+∞	+∞	+∞	+∞	+∞	+8	0	1	+∞	+∞	+∞	0
16	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+8	0	+∞	+∞	+∞	0
17	+8	+8	+8	+8	+8	+8	+8	+∞	+∞	+∞	+∞	+∞	+∞	+8	+8	+8	0	2	+∞	+∞
18	+∞	+∞	+8	+8	+8	+∞	+8	+∞	+∞	+∞	+∞	+∞	+∞	+8	+8	+8	+∞	0	+∞	+∞
19	+∞	+∞	+8	+8	+8	+8	+8	+∞	+∞	+∞	+∞	+∞	+∞	+8	+8	+8	+∞	+∞	0	0
20	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0

S <sub>3</sub>	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0	0	+8	+8	0	0	2	4	2	1	1	2	+8	3	2	2	+∞	5	+∞	2
2	+8	0	+8	+8	+8	0	2	4	+8	1	1	2	+8	3	2	2	+∞	5	+∞	2
3	+8	+8	0	1	+8	+8	2	2	+8	+8	3	3	+8	+∞	4	4	+∞	+8	+∞	4
4	+∞	+∞	+∞	0	+∞	+∞	+∞	2	+∞	+8	+∞	3	+∞	+∞	+∞	4	+∞	+8	4	4
5	+8	+8	+8	+8	0	0	2	4	2	1	2	2	+8	3	2	2	+∞	5	+∞	2
6	+∞	+∞	+∞	+∞	+∞	0	2	4	+∞	1	1	2	+∞	3	2	2	+∞	5	+∞	2
7	+∞	+∞	+∞	+∞	+∞	+∞	0	2	+∞	+∞	1	1	+∞	+∞	2	2	+∞	+∞	+∞	2
8	+8	+8	+8	+8	+8	+8	+8	0	+8	+8	+8	1	+8	+8	+8	2	+∞	+8	+∞	2
9	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	1	2	3	+∞	2	2	3	+∞	4	+∞	2
10	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	1	2	+∞	2	1	2	+∞	4	+∞	2
11	+8	+8	+8	+8	+8	+8	+8	+8	+8	+8	0	1	+8	+8	1	1	+∞	+8	+∞	2
12	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	+∞	+∞	+∞	1	+∞	+∞	+∞	2
13	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	2	3	4	1	2	+∞	3
14	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	1	2	+∞	2	+∞	2
15	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	1	+∞	+∞	+∞	0
16	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	+∞	+∞	+∞	0
17	+∞	+8	+∞	+∞	+∞	+8	+8	+8	+8	+8	+8	+8	+∞	+8	+8	+8	0	2	+∞	+∞
18	+8	+∞	+8	+∞	+8	+∞	+∞	+8	+8	+8	+∞	+∞	+8	+8	+8	+8	+∞	0	+∞	+∞
19	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	0
20	+∞	+∞	+∞	+8	+∞	+8	+8	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+8	+∞	+8	+∞	0

S <sub>4</sub>	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0	0	+∞	+∞	0	0	2	4	2	1	1	2	+∞	3	2	2	+∞	5	+∞	2
2	+∞	0	+∞	+8	+∞	0	2	4	+∞	1	1	2	+∞	3	2	2	+∞	5	+∞	2
3	+8	+8	0	1	+8	+8	2	2	+∞	+8	3	3	+8	+8	4	4	+∞	+8	+8	4
4	+8	+8	+8	0	+8	+8	+8	2	+∞	+8	+8	3	+8	+8	+8	4	+∞	+8	4	4
5	+∞	+8	+8	+8	0	0	2	4	2	1	2	2	+8	3	2	2	+∞	5	+8	2
6	+8	+8	+8	+8	+8	0	2	4	+∞	1	1	2	+8	3	2	2	+∞	5	+8	2
7	*8	+8	+8	+8	+8	+8	0	2	+∞	+8	1	1	+8	+8	2	2	+∞	+8	+8	2
8	+8	+8	+8	+8	+8	+8	+8	0	+∞	+8	+8	1	+8	+8	+8	2	+∞	+8	+8	2
9	+8	+8	+8	+8	+8	+8	+8	+8	0	1	2	3	+8	2	2	3	+∞	4	+8	2
10	+8	+8	+8	+8	+8	+8	+8	+∞	+∞	0	1	2	+8	2	1	2	+∞	4	+8	2
11	+8	+8	+8	+8	+8	+8	+8	+∞	+∞	+8	0	1	+8	+8	1	1	+∞	+8	+∞	2
12	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	+∞	+∞	+∞	1	+∞	+∞	+∞	2
13	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	2	3	4	1	2	+∞	3
14	+∞	+∞	+∞	+8	+∞	+∞	+8	+∞	+∞	+8	+8	+∞	+∞	0	1	2	+∞	2	+8	2
15	+8	+8	+8	+8	+8	+8	+8	+∞	+∞	+8	+8	+∞	+8	+8	0	1	+∞	+8	+8	0
16	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	+∞	+∞	+∞	0
17	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+8	+8	+∞	+∞	+8	+∞	+∞	0	2	+8	+∞
18	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	+∞	+∞
19	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+8	+∞	+∞	+∞	+8	+∞	+∞	+∞	+8	0	0
20	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+8	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+8	+8	0

S <sub>5</sub>	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0	0	+∞	+∞	0	0	2	4	2	1	1	2	+∞	3	2	2	+∞	5	+∞	2
2	+∞	0	+∞	+8	+∞	0	2	4	+∞	1	1	2	+∞	3	2	2	+∞	5	+∞	2
3	+8	+8	0	1	+8	+8	2	2	+∞	+8	3	3	+∞	+8	4	4	+∞	+8	+8	4
4	+8	+8	+∞	0	+8	+8	+8	2	+∞	+8	+8	3	+∞	+8	+8	4	+∞	+8	4	4
5	+∞	+∞	+∞	+8	0	0	2	4	2	1	2	2	+∞	3	2	2	+∞	5	+8	2
6	+8	+8	+8	+8	+8	0	2	4	+∞	1	1	2	+∞	3	2	2	+∞	5	+8	2
7	+8	+8	+8	+8	+8	+8	0	2	+∞	+8	1	1	+∞	+8	2	2	+∞	+8	+8	2
8	+8	+8	+8	+8	+8	+8	+8	0	+∞	+8	+8	1	+∞	+8	+8	2	+∞	+8	+8	2
9	+8	+8	+∞	+8	+8	+8	+8	+8	0	1	2	3	+∞	2	2	3	+∞	4	+8	2
10	+8	+8	+∞	+8	+8	+8	+8	+∞	+∞	0	1	2	+∞	2	1	2	+∞	4	+8	2
11	+∞	+8	+∞	+8	+8	+8	+8	+∞	+∞	+8	0	1	+∞	+8	1	1	+∞	+8	+8	2
12	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	+∞	+∞	+∞	1	+∞	+∞	+∞	2
13	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	2	3	4	1	2	+∞	3
14	+∞	+8	+∞	+8	+∞	+∞	+8	+∞	+∞	+8	+8	+∞	+∞	0	1	2	+∞	2	+8	2
15	+8	+8	+∞	+8	+8	+8	+8	+∞	+∞	+8	+8	+∞	+∞	+8	0	1	+∞	+8	+8	0
16	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	+∞	+∞	+∞	0
17	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+8	+8	+∞	+∞	+8	+∞	+∞	0	2	+8	+∞
18	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	+∞	+∞
19	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+8	+∞	+∞	+∞	+8	+∞	+∞	+∞	+8	0	0
20	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+8	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+8	+8	0

S <sub>6</sub>	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0	0	+∞	+∞	0	0	2	4	2	1	1	2	+∞	3	2	2	+∞	5	+∞	2
2	+∞	0	+∞	+8	+∞	0	2	4	+∞	1	1	2	+∞	3	2	2	+∞	5	+∞	2
3	+8	+∞	0	1	+∞	+∞	2	2	+∞	+8	3	3	+∞	+8	4	4	+∞	+8	+∞	4
4	+8	+∞	+∞	0	+∞	+∞	+8	2	+∞	+8	+∞	3	+∞	+8	+8	4	+∞	+8	4	4
5	+8	+∞	+∞	+8	0	0	2	4	2	1	2	2	+∞	3	2	2	+∞	5	+∞	2
6	+8	+∞	+8	+8	+8	0	2	4	+∞	1	1	2	+∞	3	2	2	+∞	5	+∞	2
7	+8	+∞	+∞	+8	+∞	+∞	0	2	+∞	+8	1	1	+∞	+8	2	2	+∞	+8	+∞	2
8	+8	+∞	+∞	+8	+∞	+∞	+8	0	+∞	+8	+∞	1	+∞	+8	+8	2	+∞	+8	+∞	2
9	+8	+∞	+8	+8	+8	+8	+8	+8	0	1	2	3	+∞	2	2	3	+∞	4	+∞	2
10	+8	+8	+8	+8	+∞	+∞	+8	+∞	+∞	0	1	2	+∞	2	1	2	+∞	4	+∞	2
11	+8	+∞	+∞	+8	+∞	+∞	+8	+∞	+∞	+8	0	1	+∞	+8	1	1	+∞	+8	+∞	2
12	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	+∞	+∞	+∞	1	+∞	+∞	+∞	2
13	+∞	+∞	+∞	+8	+∞	+∞	+8	+∞	+∞	+8	+∞	+∞	0	2	3	4	1	2	+∞	3
14	+8	+8	+8	+8	+∞	+∞	+8	+∞	+∞	+8	+∞	+∞	+∞	0	1	2	+∞	2	+∞	2
15	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+8	+∞	+∞	+∞	+∞	0	1	+∞	+8	+∞	0
16	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	+∞	+∞	+∞	0
17	+8	+8	+8	+8	+8	+8	+8	+8	+8	+8	+∞	+∞	+∞	+8	+∞	+∞	0	2	+∞	+∞
18	+8	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	+∞	+∞
19	+8	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+8	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+8	0	0
20	+∞	+∞	+∞	+8	+∞	+∞	+8	+∞	+∞	+8	+∞	+∞	+∞	+∞	+8	+∞	+∞	+8	+∞	0

La tabla P(i,j) = 'L', siendo 'L' el nodo intermedio entre 'i' y 'j', quedaría así:

P(Sol)	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0	0	+∞	+∞	1	0	2	4	2	5	1	2	+∞	3	10	2	+∞	5	+∞	15
2	+∞	0	+∞	+∞	+∞	1	2	4	+8	1	1	2	+8	3	2	2	+8	5	+8	2
3	+∞	+8	0	1	+∞	+∞	2	2	+8	+8	3	3	+∞	+8	4	3	+8	+8	+8	4
4	+∞	+∞	+∞	0	+∞	+∞	+∞	2	+∞	+∞	+∞	3	+∞	+∞	+∞	4	+∞	+∞	+∞	4
5	+∞	+∞	+∞	+∞	0	0	2	4	2	1	1	2	+∞	3	2	2	+∞	5	+∞	2
6	+∞	+∞	+∞	+∞	+∞	0	2	4	+∞	1	1	2	+∞	3	2	2	+∞	5	+∞	2
7	+∞	+∞	+∞	+∞	+∞	+∞	0	2	+∞	+∞	1	1	+∞	+∞	2	2	+∞	+∞	+∞	2
8	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	+∞	+∞	+∞	1	+∞	+∞	+∞	2	+∞	+∞	+∞	2
9	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	1	2	3	+∞	2	2	3	+∞	4	+∞	2
10	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	1	2	+∞	2	1	2	+∞	4	+∞	1
11	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	1	+∞	+∞	1	1	+∞	+∞	+∞	1
12	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	+∞	+∞	+∞	1	+∞	+∞	+∞	1
13	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	2	3	4	+∞	2	+∞	3
14	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	1	2	+∞	2	+∞	1
15	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	1	+∞	+∞	+∞	0
16	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	+∞	+∞	+∞	0
17	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	2	+∞	+∞
18	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	+∞	+∞
19	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0	0
20	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	0

Nos piden llegar a la salida gastando la mínima cantidad de batería posible. La solución del ejemplo proporcionado en la guía es: 1-5-10-15-20.

#### 3.3 Verificación del P.O.B.

 $S_0[i][j]$  es óptimo, porque el mejor camino de 'i' a 'j' sin pasar por ningún nodo es S[i][j].  $S_k[i][j]$  es óptimo: en caso contrario, habría otros nodos en el camino de 'i' a 'j' pasando por  $\{1...k-1\}$  tal que su cantidad de energía sea menor que la considerada. Esto es imposible, dado que la ecuación recurrente siempre selecciona la mayor cantidad.

```
3.4 Diseño del algoritmo de cálculo de coste óptimo.
Procedimiento Floyd(MatrizAdy[1..n][1..m])
Para i=1 hasta n, hacer:
       Para j=1 hasta m, hacer:
               S[i][j]= MatrizAdy[i][i]
               P[i][i] = 0
       Fin-Para
Fin-Para
Para i=1 hasta n, hacer:
       Para j=1 hasta m, hacer:
               L1=i+1
               L2=i+m
               L3=i+m+1
               SM \leftarrow min(S[i][L1]+S[L1][j], S[i][L2]+S[L2][j], S[i][L3]+S[L3][j])
               Si S[i][i] < SM, entonces:
                      Si S[i][L1]+S[L1][j] = SM, entonces:
                              S[i][j] = S[i][L1] + S[L1][j]
                              P[i][j]= L1
                      En Otro Caso Si S[i][L2]+S[L2][j] = SM, entonces:
                              S[i][j] = S[i][L2] + S[L2][j]
                              P[i][i]= L2
                      En Otro Caso, entonces:
                              S[i][j] = S[i][L3] + S[L3][j]
                              P[i][i]= L3
                      Fin-Si
               Fin-Si
       Fin-Para
Fin-Para
Devolver S.P.
       3.5 Diseño del algoritmo de recuperación de la solución.
ALGORITMO S = RecuperaSolucion (i, j, P(1...N, 1...M)): Tabla P resultante del algoritmo
MatrAdy)
       S←Ø
       k← N*M
       Mientras k <> 0, hacer:
               k←P(i,i)
              j←k
               Añadir 'k' a S
```

Fin-Mientras Devolver S