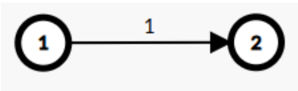
Contando entradas y salidas en una Red serie-paralela

Contribución de Román Castellarin

Descripción del problema

Mediante una secuencia de **N** operaciones, se va a construir una *red* que tendrá en total **N**+**2** nodos (representados gráficamente mediante círculos). Algunos pares de nodos estarán conectados entre sí mediante flechas **dirigidas**. Los nodos se numeran con enteros positivos, en el orden en que se van creando, y lo mismo ocurre con las flechas.

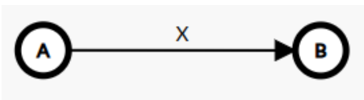
Inicialmente existen solamente dos nodos numerados **1** y **2**, y una única flecha (la número **1**) que va desde el nodo **1** hasta el **2**.



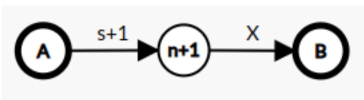
Cada una de las **N** operaciones será de uno de los siguientes dos tipos:

1. Expandir en serie la flecha **X**: Si hasta el momento la red tiene **n** nodos y **s** flechas, y la flecha **X** va desde el nodo **A** hasta el **B**, se agrega un nuevo nodo **n**+**1** en medio de la flecha **X**, de modo que la flecha **X** pasa a ir desde **n** + **1** hasta **B**, y se crea la nueva flecha **s** + **1** que va desde **A** hasta **n** + **1**.

   Es decir de

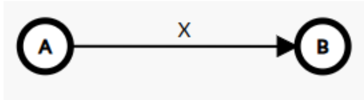
   

   Se pasa a

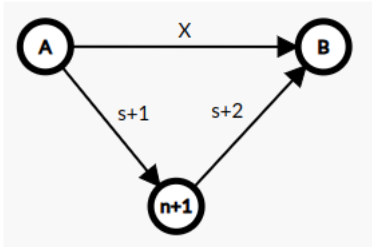
   

2. Expandir en paralelo la flecha **X**: Si hasta el momento la red tiene **n** nodos y **s** flechas, y la flecha **X** va desde el nodo **A** hasta el **B**, se agrega un nuevo nodo **n** + **1** y dos nuevas flechas, de modo que la nueva flecha **s** + **1** va desde **A** hasta **n** + **1**, y la nueva flecha **s** + **2** va desde **n** + **1** hasta **B**.

   Es decir de

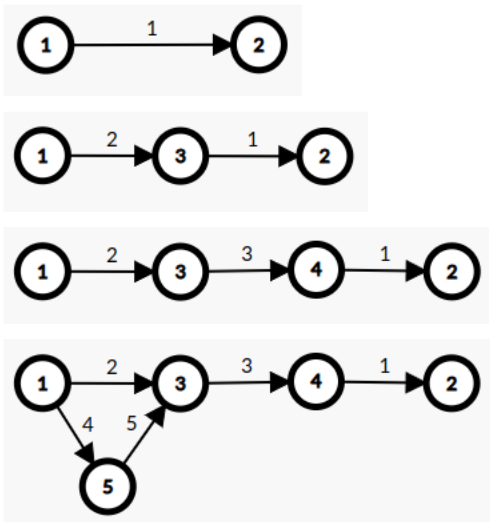
   Se pasa a

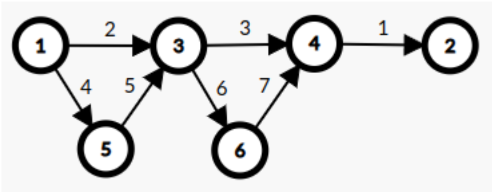
   

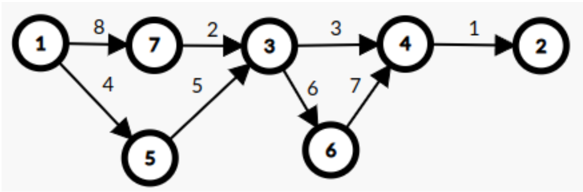
Así, la secuencia de formación de la red es la secuencia de **N** operaciones con la que se construye a partir de la flecha inicial. Por ejemplo, para la siguiente secuencia de formación con **N** = **7**:

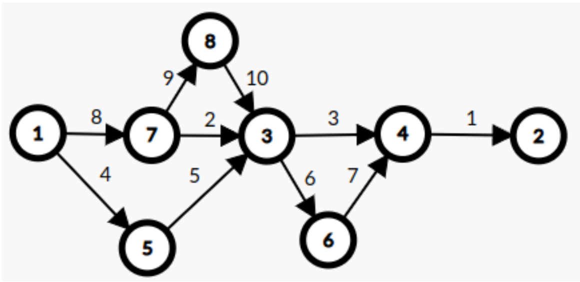
1. Expandir en serie la flecha **1**
2. Expandir en serie la flecha **1**
3. Expandir en paralelo la flecha **2**
4. Expandir en paralelo la flecha **3**
5. Expandir en serie la flecha **2**
6. Expandir en paralelo la flecha **2**
7. Expandir en paralelo la flecha **2**

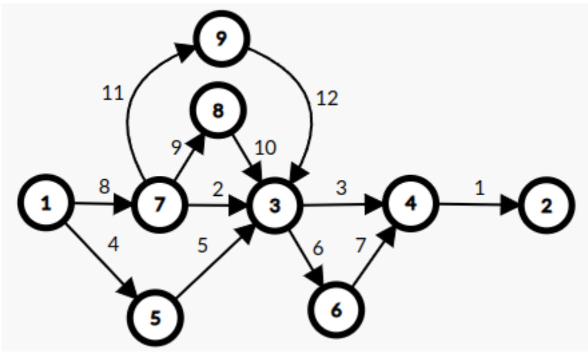
El proceso detallado hasta llegar a la red completa sería el siguiente:











Para cada nodo de la red final, definimos su *grado* como la cantidad de flechas que salen del nodo, menos la cantidad de flechas que llegan hasta el nodo. Los grados de cada nodo en el ejemplo son:

1. $\mathbf{2} - \mathbf{0} = \mathbf{2}$
2. $\mathbf{0} - \mathbf{1} = -\mathbf{1}$
3. $\mathbf{2} - \mathbf{4} = -\mathbf{2}$
4. $\mathbf{1} - \mathbf{2} = -\mathbf{1}$
5. $\mathbf{1} - \mathbf{1} = \mathbf{0}$
6. $\mathbf{1} - \mathbf{1} = \mathbf{0}$
7. $\mathbf{3} - \mathbf{1} = \mathbf{2}$
8. $\mathbf{1} - \mathbf{1} = \mathbf{0}$
9. $\mathbf{1} - \mathbf{1} = \mathbf{0}$

Dadas las **N** operaciones de la secuencia de formación de la red en orden, debes escribir una función que calcule el grado de cada nodo de la red final.

Descripción de la función

Debes implementar la función `inout(t,e)`. Sus parámetros son:

- `t`: Un arreglo de **N** enteros. **t**[**i**] es **1** si la **i**-ésima operación es expandir en serie, y **2** si la **i**-ésima operación es expandir en paralelo.
- `e`: Un arreglo de **N** enteros. **e**[**i**] es el número de flecha a expandir en la **i**-ésima operación.

La función debe retornar un arreglo de $\mathbf{N} + \mathbf{2}$ enteros: los grados de los nodos de la red final.

Evaluador

El evaluador local lee de la entrada estándar con el siguiente formato:

- El entero **N**
- **N** pares de enteros **t**[**i**], **e**[**i**]

El evaluador local escribe a la salida estándar la respuesta retornada por la función.

Restricciones

- $\mathbf{1} \leq \mathbf{N} \leq \mathbf{100}$
- $\mathbf{1} \leq \mathbf{t_i} \leq \mathbf{2}$

Ejemplo

Si se invoca al evaluador con la siguiente entrada:

```
7
1 1
1 1
2 2
2 3
1 2
2 2
2 2
```

Para un programa correcto, la salida será:

```
2 -1 -2 -1 0 0 2 0 0
```

Si en cambio la entrada es:

```
5
1 1
1 2
1 3
1 4
1 5
```

Para un programa correcto, la salida será:

```
1 -1 0 0 0 0 0
```

y si es:

```
4
2 1
2 1
2 1
2 1
```

la salida será:

```
5 -5 0 0 0 0
```

Subtareas

1. **N** = **1** (10 puntos)
2. **N** = **2** (20 puntos)
3. **e**[**i**] = **1** para todo **i** (20 puntos)
4. **t**[**i**] = **1** para todo **i**, o sea, todas operaciones de expandir en serie (15 puntos)
5. **t**[**i**] = **2** para todo **i**, o sea, todas operaciones de expandir en paralelo (20 puntos)
6. Sin más restricción (15 puntos)