

Algoritmos y Programación

Práctica 2
Introducción a Grafos

NetworkX

La práctica de esta semana tiene dos ejercicios.

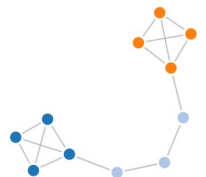
En el primer ejercicio se debe crear un grafo **no dirigido** a partir de un fichero de entrada. Para ello, utilizaremos NetworkX. Esta biblioteca debe ser estudiada. Las principales funciones a utilizar son

- `nx.Graph()`
- `G.add_node()`
- `G.add_edge()`



<https://networkx.org/documentation/stable/tutorial.html>
<https://networkx.org/documentation/stable/reference/index.html>

NetworkX is a Python package for the creation, manipulation, and study of the structure, dynamics, and functions of complex networks.

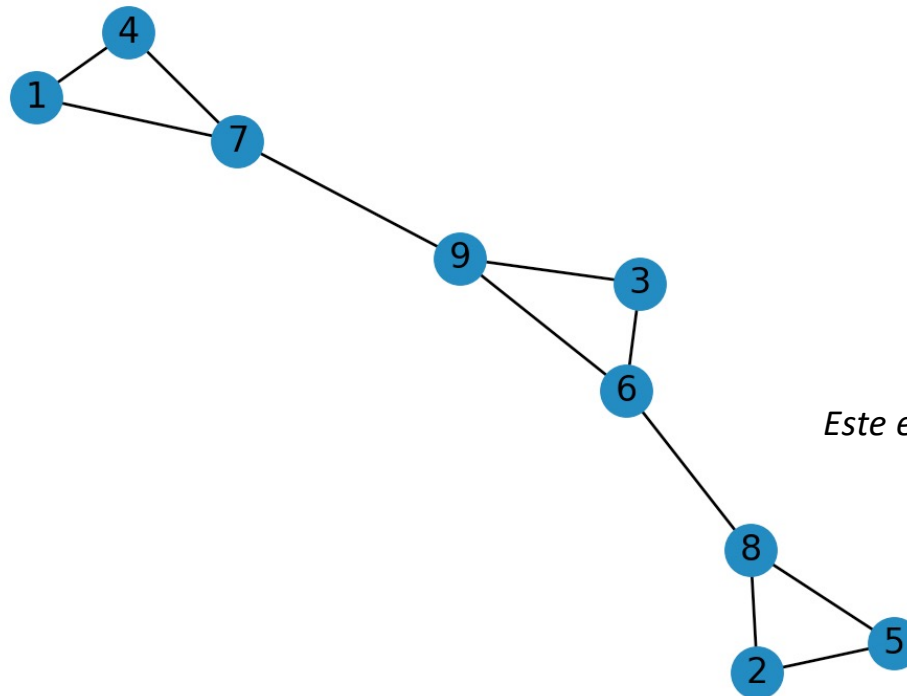


Formato del fichero de entrada

- La primera línea es un descriptor: número de vértices, número de aristas.
- El resto de las líneas son las aristas.

Ejemplo

9 11
1 4
2 8
3 6
4 7
5 2
6 9
7 1
8 5
8 6
9 7
9 3



Este ejercicio no puntúa

Ejercicio 1 de esta práctica

- Leer el fichero de entrada con el formato descrito y, utilizando NetworkX, crear el grafo **no dirigido** correspondiente.
 - Este ejercicio no puntúa; es para familiarizarnos con NetworkX

VPL: Ejercicio 1

main.py

```
1 import networkx as nx
2 from solve import *
3
4 graph = build_graph();
5
6 print("Number of nodes: " + str(graph.number_of_nodes()))
7 print("Nodes: ", graph.nodes())
8 print("Number of edges: " + str(graph.number_of_edges()))
9 print("Edges: ", graph.edges())
10
11 # Paso 3: Utilizando PyCharm añade aquí el código necesario para mostrar
12 #         gráficamente el grafo.
13 # ...
```

```
1 import networkx as nx
2
3 def build_graph():
4     """
5     Read data from the standard input and build the corresponding
6     nondirected graph without weights. Nodes numbering starts with
7     number 1 (that is, nodes are 1,2,3,...)
8     """
9     first_line = input().split()
10    num_nodes = int(first_line[0])
11    num_edges = int(first_line[1])
12
13    # Paso 1: Crear un grafo unidireccional con num_nodes
14
15
16    # Paso 2: Añadirle las aristas
17
18
19    return graph
20
```

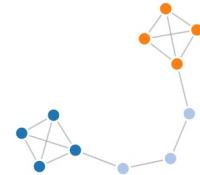
NetworkX

En el segundo ejercicio se debe crear un grafo **dirigido con pesos** a partir de un fichero de entrada.

- `nx.DiGraph()`
- `G.add_node()`
- `G.add_edge()`



NetworkX is a Python package for the creation, manipulation, and study of the structure, dynamics, and functions of complex networks.



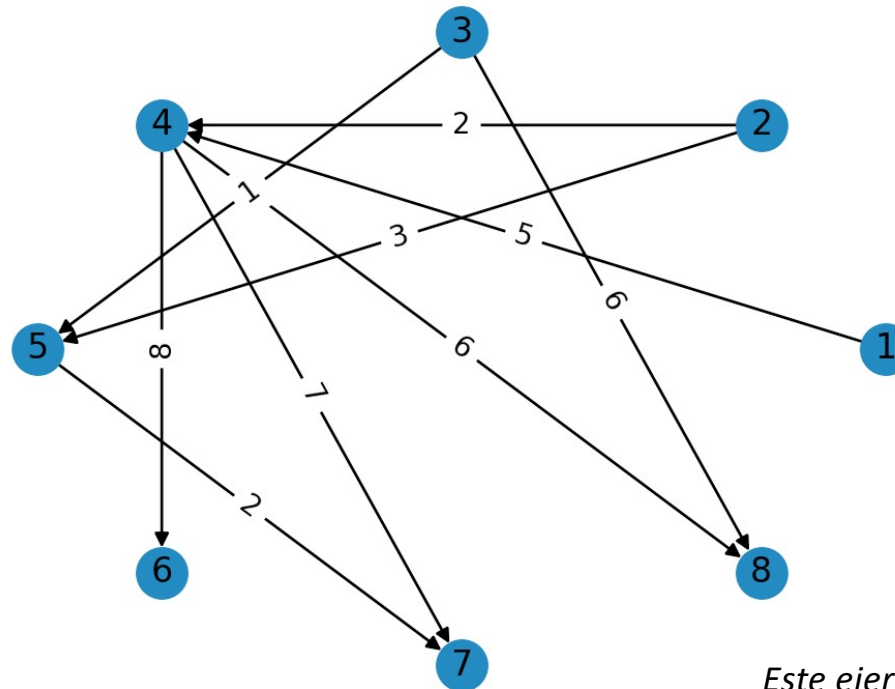
<https://networkx.org/documentation/stable/tutorial.html>
<https://networkx.org/documentation/stable/reference/index.html>

Formato del fichero de entrada

- La primera línea es un descriptor: número de vértices, número de aristas
- El resto de las líneas son las aristas con su peso

Ejemplo:

```
8 9
1 4 5
2 4 2
2 5 3
3 5 1
3 8 6
4 6 8
4 7 7
4 8 6
5 7 2
```



Este ejercicio no puntúa

VPL: Ejercicio 2

main.py

```
1 import networkx as nx
2 from solve import *
3
4 graph = build_digraph_with_weights()
5
6 print("Number of nodes: " + str(graph.number_of_nodes()))
7 print("Nodes: ", graph.nodes())
8 print("Number of edges: " + str(graph.number_of_edges()))
9 print("Edges: ", graph.edges(data=True))
10
11 # Paso 3: Utilizando PyCharm añade aqui el código necesario para mostrar
12 #         gráficamente el grafo.
13 # ...
```

solve.py

```
1 import networkx as nx
2
3 def build_digraph_with_weights():
4     """
5     Read data from the standard input and build the corresponding
6     directed graph with weights. Nodes numbering starts with number
7     1 (that is, nodes are 1,2,3,...)
8     """
9
10    first_line = input().split()
11    num_nodes = int(first_line[0])
12    num_edges = int(first_line[1])
13
14    # Paso 1: Crear grafo direccional con num_nodes
15
16
17    # Paso 2: Añadir los vértices del grafo
18
19
20    return graph
```