Exercices représentation correction

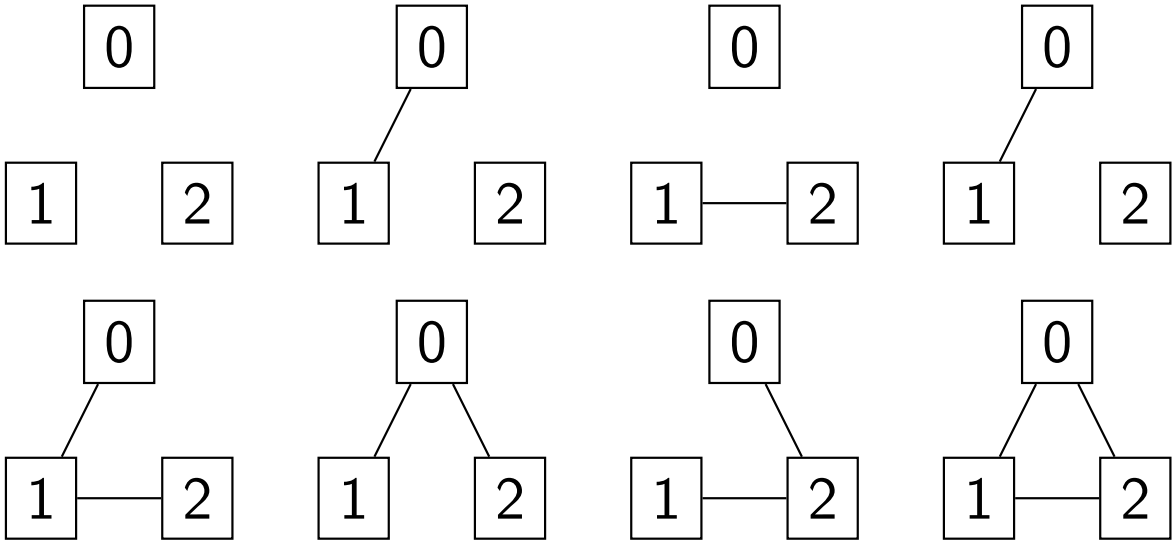
Christophe Viroulaud

Terminale - NSI

Algo 15

Sommaire

Exercice 1



Sommaire

Exercice 2

degrés :

- A : 4
- B : 2
- C : 2
- D : 3
- E : 2
- F : 1

$$\sum_{s \in S} deg(s) = 14 = 2.A$$

1. Matrice d'adjacence

$$\begin{pmatrix} 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

2. Dictionnaire d'adjacence

```
1 g = {"A": ["B", "C", "D", "F"],
2     "B": ["A", "D"],
3     "C": ["A", "E"],
4     "D": ["A", "B", "E"],
5     "E": ["C", "D"],
6     "F": ["A"]}
```

Sommaire

Exercice 3

Le graphe est complet.

```
1 mat = [ [0, 1, 1, 1, 1],
2         [1, 0, 1, 1, 1],
3         [1, 1, 0, 1, 1],
4         [1, 1, 1, 0, 1],
5         [1, 1, 1, 1, 0]]
```

Code 1 – Matrice d'adjance

```python
def ordre(mat: list) -> int:
    return len(mat)
```

```
>>> ordre(mat)
5
```

Code 2 – Appel

```python
def est_complet(mat: list) -> bool:
    """
    vérifie si chaque sommet est relié à tous
    les autres (sauf lui même)
    """
    for ligne in range(ordre(mat)):
        for col in range(ordre(mat)):
            if (ligne != col and mat[ligne][col] == 0) or \
                    (ligne == col and mat[ligne][col] == 1):
                return False
    return True
```

```
>>> est_complet(mat)
True
```

Code 3 – Appel

Remarque

Les parenthèses ne sont pas obligatoires ici : la porte **`and`** est prioritaire devant **`or`**. Il est conseillé de les ajouter tout de même en cas de doute.

Sommaire

Exercice 4

- 0 : $d^+ = 2, d^- = 0$
- 1 : $d^+ = 1, d^- = 0$
- 2 : $d^+ = 1, d^- = 1$
- 3 : $d^+ = 0, d^- = 3$
- 4 : $d^+ = 1, d^- = 2$
- 5 : $d^+ = 1, d^- = 2$

```
1 suivants = [[3, 5], [3], [4], [], [3, 5], [2]]
```

```
1 def degres_sortants(liste: list, s: int) -> int:
2     return len(liste[s])
```

```
1 >>> degres_sortants(suivants, 4)
2 2
```

Code 4 – Appel

```python
def degres_entrants(liste: list, s: int) -> int:
    deg = 0
    for liste_successeurs in liste:
        for successeurs in liste_successeurs:
            if successeurs == s:
                deg = deg+1
    return deg
```

```
>>> degres_entrants(suivants, 4)
1
```

Code 5 – Appel