Solution du devoir 3

Auteur : Alexandre Blondin Massé

1. Supposez que vous avez n billes identiques devant vous. Vous souhaitez diviser ces n billes en k paquets. Par exemple, si $n = 5$ et $k = 3$, vous pourriez diviser les $n = 5$ billes comme suit : 2 paquets de 2 billes et 1 paquet de 1 bille, pour un total de $k = 3$ paquets, qu'on représente par le triplet $(2, 2, 1)$, appelé *partage*. Plus précisément, un *partage* de n billes en k paquets est un k-tuplet $(n_1, n_2, \ldots, n_k)$ tel que $n_1 + n_2 + \ldots + n_k = n$ et $n_1 \geq n_2 \geq \ldots \geq n_k \geq 1$.

   Dénotons par $P(n, k)$ l'ensemble de tous les partages de n billes en k paquets, où chaque paquet contient au moins 1 bille et soit $p(n, k) = |P(n, k)|$.

   (a) (2 points) Montrez que $p(8, 3) = 5$ en calculant $P(8, 3)$.

   Solution: On remarque que

   $$P(8, 3) = \{(6, 1, 1), (5, 2, 1), (4, 3, 1), (4, 2, 2), (3, 3, 2)\},$$

   de sorte que $p(8, 3) = 5$.

   (b) (3 points) Calculez les valeurs de $p(n, k)$ pour $1 \leq k \leq n \leq 5$. Présentez votre solution sous forme de tableau.

   Solution: On obtient

$n \setminus k$	1	2	3	4	5
1	1	1	1	1	1
2		1	1	2	2
3			1	1	2
4				1	1
5					1

   (c) (5 points) Les nombres $p(n, k)$ vérifient les égalités suivantes :

   (i) $p(n, k) = 0$ si $n < k$;

   (ii) $p(n, n) = p(n, 1) = 1$;

   (iii) $p(n, k) = p(n - 1, k - 1) + p(n - k, k)$.

   Donnez un argument combinatoire qui justifie les égalités (i), (ii) et (iii). *Indice :* Étant donné un partage, il y a deux possibilités : (1) il existe au moins un paquet ayant exactement une bille ou (2) tous les paquets ont au moins deux billes.

Solution: (i) Il est impossible de partager n billes en k paquets si $k > n$, ce qui entraîne que $p(n, k) = 0$ dans ce cas.

(ii) La seule façon de partager n billes en n paquets est de composer n paquets de 1 bille. De la même façon, il n'y a qu'une seule façon de partager n billes en 1 paquet : c'est de prendre toutes les billes dans le même paquet.

(iii) On souhaite maintenant compter le nombre de façons de former k paquets avec n billes dans le cas général. Comme le suggère l'indice, on remarque qu'un tel partage doit vérifier exactement une des deux conditions données dans l'indice. Par le principe de la somme, il suffit donc de compter le nombre de paquets ayant exactement une bille d'une part, puis le nombre de paquets ayant au moins deux billes d'autres part.

Comptons d'abord les partages dans lesquels il y a au moins 1 paquet de 1 bille. Il reste alors $n - 1$ billes qu'on doit répartir dans $k - 1$ paquets, de sorte qu'il y en a justement $p(n - 1, k - 1)$.

Il ne nous reste qu'à compter le nombre de partages dans lesquels tous les paquets contiennent au moins 2 billes. On peut alors enlever 1 bille de chacun des k paquets (il reste alors $n - k$) billes et alors on se retrouve à compter le nombre de partages de $n - k$ billes en k paquets, qui est donné par $p(n - k, k)$.

(d) (10 points) Écrivez un générateur dans SageMath (ou Python) qui, étant donnés deux entiers positifs n et k tels que $1 \leq k \leq n$, génère les éléments de l'ensemble $P(n, k)$. Par exemple, si votre fonction est de la forme

```
def paquets(n, k):
    #
    # A compléter
    #
    # Vous devez utilisez le mot réservé
    #
    #   yield
    #
    # au moins une fois dans votre code
    # pour obtenir un générateur
    #
```

alors on s'attend à ce que le bout de code suivant

```
for paquet in paquets(9, 4):
    print paquet
```

affiche

```
(6, 1, 1, 1)
(5, 2, 1, 1)
(4, 3, 1, 1)
(4, 2, 2, 1)
(3, 3, 2, 1)
(3, 2, 2, 2)
```

Note : Les équations données à la sous-question (c) cachent un algorithme récursif. *Suggestion :* Comme votre générateur retourne des k-tuplets, certaines fonctions sur les k-tuplets pourraient vous être utiles :

- Il est possible de construire un k-tuplet en compréhension, comme pour les listes et les ensembles. Par exemple `tuple(1 for _ in range(n))` construit le n-tuplet $(1, 1, \ldots, 1)$. De la même façon, `tuple(i + 1 for i in paquet)` construit un nouveau k-tuplet à partir de `paquet` en ajoutant 1 à chaque élément du k-tuplet `paquet`.
- Il est possible de concaténer des k-tuplets à l'aide de l'opérateur `+`. Par exemple, `(1, 2, 3) + (4, 5, 6)` retourne `(1, 2, 3, 4, 5, 6)`
- Pour construire un 1-tuplet, une particularité syntaxique de Python est qu'il faut ajouter une virgule à la fin (sinon, l'interpréteur croit qu'il s'agit d'une expression parenthésée et non d'un k-tuplet). Par exemple, le 1-tuplet (6) est représenté par `(6,)` en Python.

Solution: Il suffit de diviser les cas selon ceux décrits par la sous-question précédente.

```python
# -*- coding: utf-8 -*-

def paquets(n, k):
    r"""
    Retourne un générateur des partages de `n` billes en `k`
    paquets.

    Un partage de `n` billes en `k` paquets est un `k`-tuplet
    de nombres naturels non nuls décroissant dont la somme est
    `n`.
    """
    if n >= k:
        if n == k:
            yield tuple(1 for _ in range(n))
        elif k == 1:
            yield (n,)
        else:
            for paquet in paquets(n - 1, k - 1):
                yield paquet + (1,)
            for paquet in paquets(n - k, k):
                yield tuple(i + 1 for i in paquet)

# Test
for paquet in paquets(9, 4):
    print paquet

# Résultat
# --------
# (6, 1, 1, 1)
# (5, 2, 1, 1)
```

```
# (4, 3, 1, 1)
# (4, 2, 2, 1)
# (3, 3, 2, 1)
# (3, 2, 2, 2)
```

2. Dans cette question, nous nous intéressons à construire deux relations sur les éléments de $\mathbb{N}^2$ à partir de deux fonctions mathématiques très importantes. La première est la fonction signe sign : $\mathbb{R} \to \{-1, 0, 1\}$ définie par

$$\text{sign}(z) = \begin{cases} -1, & \text{si } z < 0; \\ 0, & \text{si } z = 0; \\ 1, & \text{si } z > 0. \end{cases}$$

La seconde est la troncature trunc : $\mathbb{R} \to \mathbb{Z}$, définie par

$$\text{trunc}(z) = \text{sign}(z)\lfloor |z| \rfloor,$$

qui est semblable à la fonction plancher $\lfloor \cdot \rfloor$, mais qui a un comportement différent sur les valeurs négatives.

De plus, considérons les trois relations suivantes définies sur $\mathbb{N}^2$.

1. La relation $\leftrightarrow$, définie par

$$(x, y) \leftrightarrow (x', y') \quad \text{si et seulement si} \quad (x - x')^2 + (y - y')^2 = 1. \tag{1}$$

On dit alors que (x, y) et (x', y') sont *voisins*.

2. La relation $\to$, définie par

$$(x, y) \to (x', y') \quad \text{si et seulement si} \quad (x, y) = (\lfloor x'/2 \rfloor, \lfloor y'/2 \rfloor) \tag{2}$$

On dit alors que (x, y) est le *parent* de (x', y').

3. La relation $\Leftrightarrow$ définie par

$$p \Leftrightarrow p' \quad \text{si et seulement si} \quad \text{il existe } p'' \in \mathbb{N}^2 \text{ tel que } p'' \to p \text{ et } p'' \to p', \tag{3}$$

On dit alors que p et p' sont *frères*, puisqu'ils ont le même parent.

(a) (2 points) Montrez que $\text{sign}(-z) = -\text{sign}(z)$ pour tout $z \in \mathbb{R}$.

(b) (2 points) Montrez que $\text{trunc}(-z) = -\text{trunc}(z)$ pour tout $z \in \mathbb{R}$.

(c) (6 points) Pour chaque propriété parmi la réflexivité, l'irréflexivité, la transitivité, la symétrie, l'antisymétrie et l'asymétrie, indiquez si la relation $\leftrightarrow$ satisfait ou non la propriété. Dans chaque cas, justifiez.

(d) (6 points) Pour chaque propriété parmi la réflexivité, l'irréflexivité, la transitivité, la symétrie, l'antisymétrie et l'asymétrie, indiquez si la relation $\to$ satisfait ou non la propriété. Dans chaque cas, justifiez.

(e) (4 points) Montrez que la relation $\Leftrightarrow$ est une relation d'équivalence.

(f) (5 points) Dessinez le graphe des relations $\to$ et $\leftrightarrow$ si on se restreint à l'ensemble de sommets $V = \{(x, y) \mid 0 \leq x, y \leq 7\}$.

(g) (5 points) Calculez les classes d'équivalence de la relation $\Leftrightarrow$ lorsqu'on se restreint à l'ensemble de sommets V mentionné à la sous-question (f). Ajoutez au dessin fait en (f) ces classes d'équivalence.

Solution: (a) Il y a trois cas à considérer. Si $z < 0$, alors

$$\text{sign}(-z) = 1 = -(-1) = -\text{sign}(z).$$

Si $z = 0$, alors

$$\text{sign}(-z) = \text{sign}(0) = 0 = -0 = -\text{sign}(0) = -\text{sign}(z).$$

Finalement, si $z > 0$, alors

$$\text{sign}(-z) = -1 = -\text{sign}(z),$$

tel que voulu.

(b) On a

$$\begin{aligned}
\text{trunc}(-z) &= \text{sign}(-z)\lfloor|-z|\rfloor && \text{(par définition de trunc)} \\
&= -\text{sign}(z)\lfloor|z|\rfloor && \text{(par (a) et propriété de valeur absolue)} \\
&= -\text{trunc}(z) && \text{(par définition de trunc)}
\end{aligned}$$

(c) La relation $\leftrightarrow$ n'est pas réflexive. Par exemple, si on prend $(x, y) = (0, 0)$, alors on voit bien que $(0 - 0)^2 + (0 - 0)^2 = 0 \neq 1$.

Elle est irréflexive, puisque pour tout $(x, y) \in \mathbb{N}^2$, on a justement $(x-x)^2+(y-y)^2 = 0 \neq 1$.

Elle n'est pas transitive. Par exemple, $(0, 0) \leftrightarrow (1, 0)$ et $(1, 0) \leftrightarrow (2, 0)$, puisque $(0 - 1)^2 + (0 - 0)^2 = 1$ et $(1 - 2)^2 + (0 - 0)^2 = 1$, mais $(0 - 2)^2 + (0 - 0)^2 = 4 \neq 1$.

Elle est symétrique, puisque pour toute paire de couples $(x, y), (x', y') \in \mathbb{N}^2$, si $(x, y) \leftrightarrow (x', y')$, alors

$$\begin{aligned}
(x - x')^2 + (y - y')^2 = 1 &\Rightarrow [(-1)(x - x')]^2 + [(-1)(y - y')]^2 = 1 \\
&\Rightarrow (-x + x')^2 + (-y + y')^2 = 1 \\
&\Rightarrow (x' - x)^2 + (y' - y)^2 = 1,
\end{aligned}$$

de sorte que $(x', y') \leftrightarrow (x, y)$.

Elle n'est pas antisymétrique. Par exemple, $(0,0) \leftrightarrow (1,0)$ et $(1,0) \leftrightarrow (0,0)$, mais $(1,0) \neq (0,0)$.

Elle n'est pas asymétrique, puisqu'elle est symétrique et non vide.

(d) La relation $\rightarrow$ n'est pas réflexive. Par exemple, $(1,0) \nrightarrow (1,0)$, puisque

$$(\lfloor 1/2 \rfloor, \lfloor 1/2 \rfloor) = (0,0) \neq (1,0).$$

Elle n'est pas irréflexive non plus, puisque $(0,0) \rightarrow (0,0)$. C'est en fait le seul couple en relation avec lui-même pour la relation $\rightarrow$.

Elle n'est pas transitive. Par exemple, $(0,0) \rightarrow (1,0)$ et $(1,0) \rightarrow (2,0)$, mais $(0,0) \nrightarrow (2,0)$.

Elle n'est pas symétrique, puisque par exemple $(0,0) \rightarrow (1,0)$, mais $(1,0) \nrightarrow (0,0)$.

Elle est antisymétrique. En effet, supposons que $(x,y) \rightarrow (x',y')$ et $(x',y') = (x,y)$. Alors

$$(x,y) = (\lfloor x'/2 \rfloor, \lfloor y'/2 \rfloor) \quad \text{et} \quad (x',y') = (\lfloor x/2 \rfloor, \lfloor y/2 \rfloor).$$

Plus précisément, il existe des nombres $q_1, q_2, q_3, q_4 \geq 0$ et r_1, r_2, r_3, r_4 (les quotients et les restes), avec $0 \leq r_1, r_2, r_3, r_4 \leq 1$, tels que

$$\begin{aligned} x &= q_1 x' + r_1, & (4) \\ y &= q_2 y' + r_2, & (5) \\ x' &= q_3 x + r_3, & (6) \\ y' &= q_4 y + r_4. & (7) \end{aligned}$$

Les equations (4) et (6) entraînent

$$x = q_1(q_3 x + r_3) + r_1 = q_1 q_3 x + q_1 r_3 + r_1.$$

Or, comme tous les quotients et les restes sont entiers et non négatifs, la seule possibilité est que $r_1 = r_3 = 0$ et alors $q_1 = q_3 = 1$, ce qui implique $x = x'$. Par un raisonnement similaire avec les équations (5) et (7), on montre que $y = y'$.

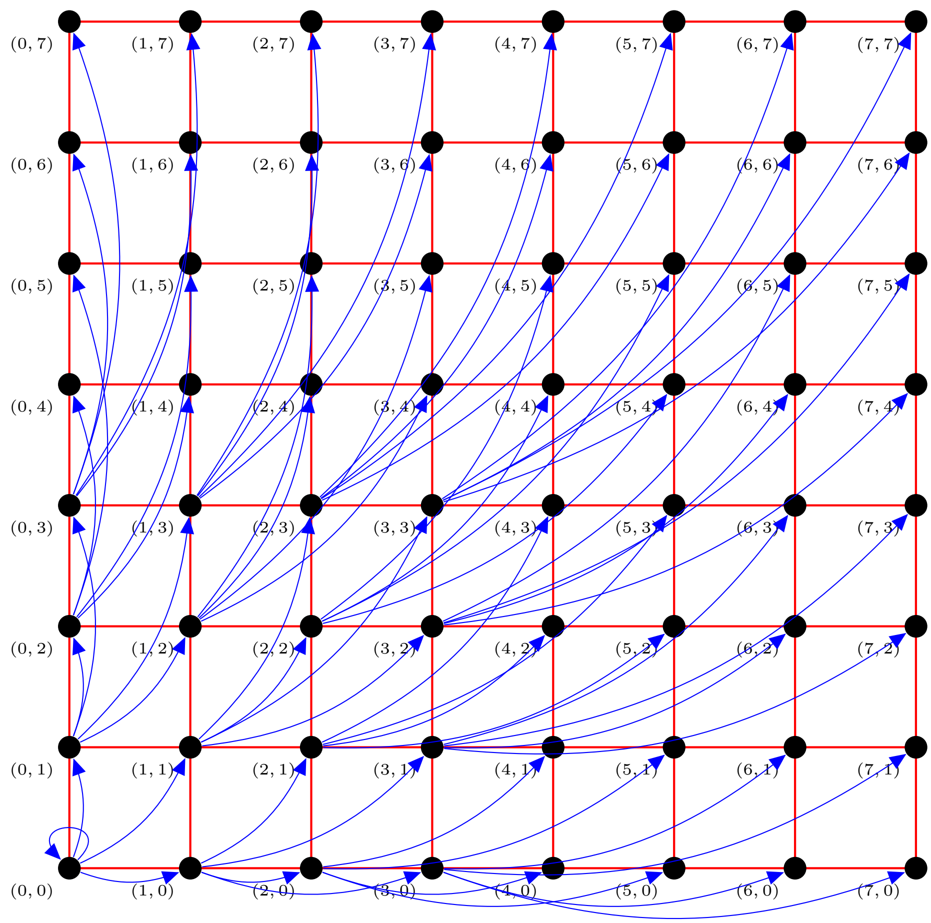
La relation n'est pas asymétrique, puisque $(0,0) \rightarrow (0,0)$.

(e) Réflexivité. Pour tout $p \in \mathbb{N}^2$, on a bien $p \Leftrightarrow p$ puisque chaque point de $\mathbb{N}^2$ a bien un parent aussi dans $\mathbb{N}^2$. En effet, si $p = (x,y)$, alors $p' = (\lfloor x/2 \rfloor, \lfloor y/2 \rfloor) \in \mathbb{N}^2$ et $p' \rightarrow p$.

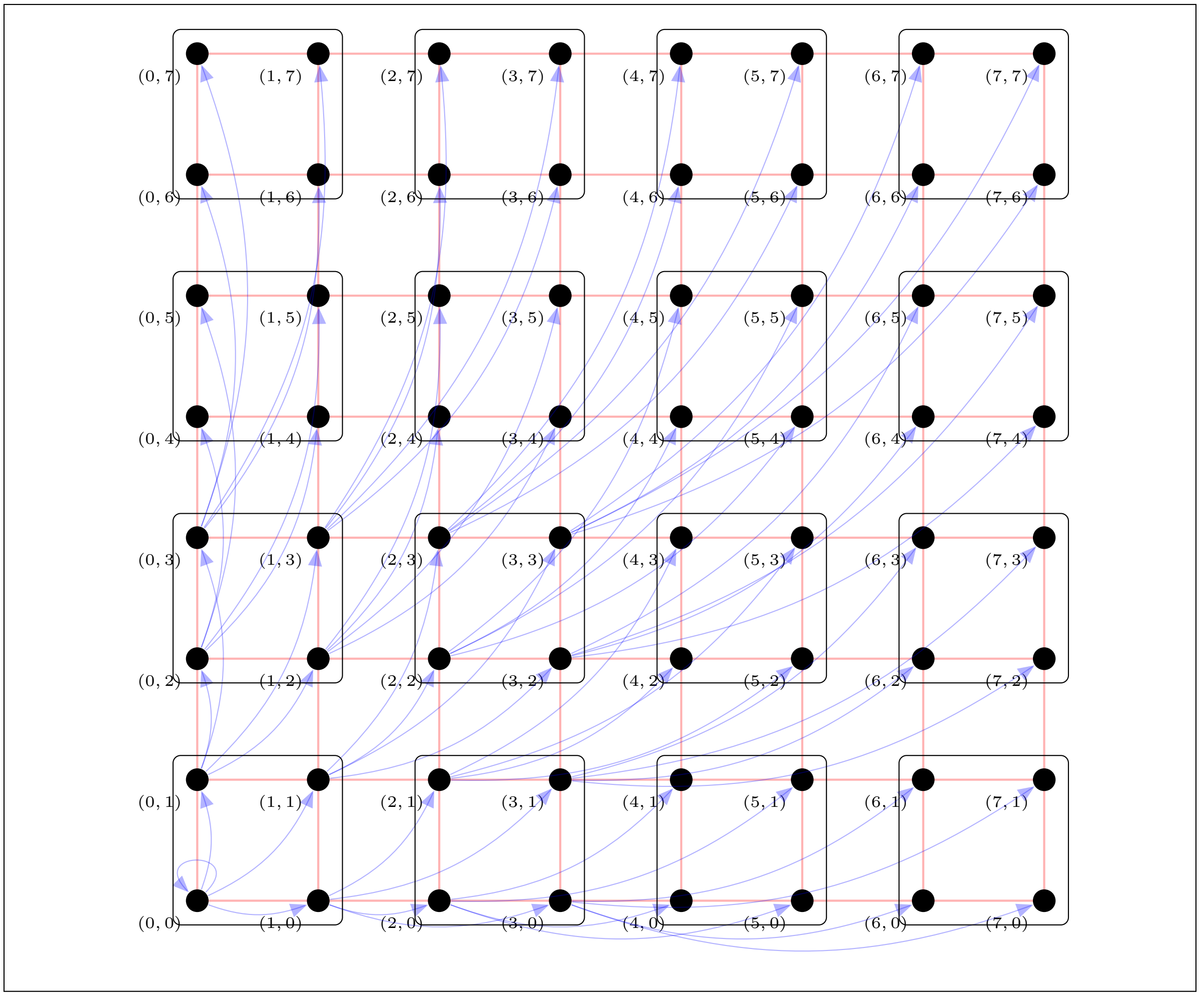
Symétrie. Si $p \Leftrightarrow p'$, alors il existe un point p'' qui est parent de p et de p'. Il est donc aussi parent de p' et de p. Autrement dit, comme l'opérateur logique $\wedge$ (et) est commutatif, la relation $\Leftrightarrow$ aussi doit être symétrique.

Transitivité. Supposons que $p \Leftrightarrow p'$ et $p' \Leftrightarrow p''$. Alors il existe des points q et q' tels que $q \to p$, $q \to p'$, $q' \to p'$ et $q' \to p''$. On remarque en particulier que q et q' sont uniques. Autrement dit, la relation $\to$ définit (x, y) comme une *fonction* de (x', y'), et on sait qu'en appliquant une fonction à une valeur, le résultat est nécessairement unique, par définition de fonction. Par conséquent, nous avons que $q = q'$, ce qui entraîne $p \Leftrightarrow p''$.

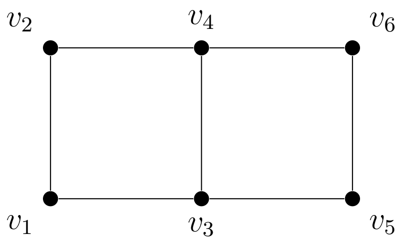
(f) Il est possible de dessiner les graphes des deux relations dans un seul graphe, puisque l'ensemble des sommets est le même. Dans l'image ci-bas, on identifie la relation $\to$ par des flèches bleues et la relation $\leftrightarrow$ par des arêtes rouges. On obtient ceci :



(g) Les classes d'équivalence sont identifiés par des rectangles (arrondis) noirs dans l'image ci-bas.



3. Un *automorphisme* de graphes est un isomorphisme d'un graphe G avec lui-même. Intuitivement, un automorphisme correspond à une symétrie d'un graphe. Par exemple, prenons le graphe non orienté ci-bas :



Ce graphe possède quatre axes de symétrie décrits par les quatre automorphismes suivants :

Fonction	v_1	v_2	v_3	v_4	v_5	v_6
f_1	v_1	v_2	v_3	v_4	v_5	v_6
f_2	v_2	v_1	v_4	v_3	v_6	v_5
f_3	v_5	v_6	v_3	v_4	v_1	v_2
f_4	v_6	v_5	v_4	v_3	v_2	v_1

Pour chacun des graphes suivants, donnez son nombre d'automorphismes. Justifiez.

(a) (4 points) Le graphe complet K_n, pour $n \geq 1$;

> **Solution:** Tous les sommets sont interchangeables. Par conséquent, on a n choix pour le premier sommet, $n-1$ choix pour le deuxième, $n-2$ choix pour le troisième, etc. On a donc $n!$ automorphismes.

(b) (4 points) Le graphe biparti complet $K_{m,n}$, pour $m, n \geq 1$;

> **Solution:** Il y a deux cas à considérer. Si $m = n$, alors on a $2n$ choix pour le premier sommet (on prend n'importe quel sommet), puis on a $n-1$ choix pour le deuxième (il doit être dans la même partie que le premier sommet choisi), $n-2$ choix pour le troisième, etc. Une fois que tous les sommets d'une même partie ont été choisis, il nous reste $n!$ choix pour la seconde partie. En tout, il y a donc $2n!n!$ choix.
>
> Dans le cas où $m \neq n$, alors on a $m!$ choix pour une partie et $n!$ pour l'autre, ce qui donne $m!n!$ choix en tout.

(c) (4 points) Le cycle C_n, pour $n \geq 3$;

> **Solution:** On avait fait cet exemple en classe. Il y a n choix pour le premier sommet, puis ensuite 2 choix pour le deuxième (qui détermine alors le sens de parcours du cycle). Une fois ces deux sommets choisis, les autres sont entièrement déterminés. Il y a donc $2n$ automorphismes.

(d) (4 points) L'hypercube Q_n, pour $n \geq 1$;

> **Solution:** Celui-là est plus difficile. Tout d'abord, on a 2^n choix pour le premier sommet. Une fois choisi, on a $n!$ choix pour ses voisins. Quand les voisins ont été choisis, tous les autres sommets sont déterminés par ces choix. Il y en a donc $2^n n!$.

(e) (4 points) La roue W_n, pour $n \geq 3$;

> **Solution:** Pour W_3, il y a $4!$ automorphismes puisqu'il s'agit du graphe complet K_4. Pour $n \geq 4$. Le sommet central de la roue est nécessairement fixé par

l'automorphisme et les sommets autour se comportent comme pour le cycle C_n. Il y a donc $2n$ automorphismes dans ce cas.

4. (15 points) Soit $G = (V, E)$ un graphe simple et $U \subseteq V$. On appelle *sous-graphe induit par U* le graphe
$$G[U] = (U, E \cap \mathcal{P}_2(U)),$$
où $\mathcal{P}_2(V)$ est l'ensemble des paires d'éléments de V. Autrement dit, c'est l'unique sous-graphe de G obtenu en prenant les sommets qui sont dans U et toutes les arêtes dont les deux extrémités sont elles aussi dans U.

   Aussi, étant donné un graphe $G = (V, E)$, on dit que G est *acyclique* si les seuls cycles qu'il contient sont triviaux. Un *transversal de cycles* de G est un sous-ensemble de sommets $U \subseteq V$ tel que pour tout cycle $(v_1, v_2, \ldots, v_k)$ de G, il existe au moins un indice $i \in \{1, 2, \ldots, k\}$ tel que $v_i \in U$.

   Montrez que U est un transversal de cycle de G si et seulement si $G[V - U]$ est un graphe acyclique.

Solution: ($\Rightarrow$) Soit U un transversal de cycles de G et supposons, par contradiction, que $G[V - U]$ n'est pas acyclique, c'est-à-dire qu'il existe un cycle $c = (v_1, v_2, \ldots, v_k, v_1)$ dans $G[V - U]$. En particulier, c est un cycle de G, puisque $G[V - U]$ est un sous-graphe de G. Comme U est un transversal de cycles de G, il doit y avoir un indice $i \in \{1, 2, \ldots, k\}$ tel que $v_i \in U$. Or, $v_1, v_2, \ldots, v_k \in V - U$, c'est-à-dire qu'on a la fois $v_i \in U$ et $v_i \notin U$, ce qui est absurde.

($\Leftarrow$) Supposons que $G[V - U]$ est acyclique et supposons par contradiction qu'il existe un cycle $c = (v_1, v_2, \ldots, v_k, v_1)$ de G qui n'est pas couvert par U, c'est-à-dire que $v_1, v_2, \ldots, v_k \notin U$. Alors en particulier $v_1, v_2, \ldots, v_k \in V - U$, de sorte que c est un cycle de $G[V - U]$, ce qui contredit l'hypothèse que $G[V - U]$ est acyclique.

5. Étant donné un arbre non orienté quelconque $T = (V, E)$, on dénote par $E(T)$ l'arbre obtenu de T en supprimant toutes ses feuilles (une feuille est simplement un sommet de degré exactement 1, en particulier, un sommet de degré 0 n'est pas une feuille).

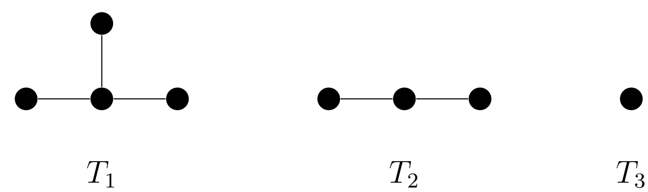
   (a) (5 points) Vrai ou faux? Pour tout arbre T, il existe un unique arbre T' tel que $T = E(T')$. Si c'est vrai, démontrez-le, si c'est faux, donnez un contre-exemple.

   (b) (5 points) Vrai ou faux? Pour tout arbre T, il existe un unique arbre T' tel que $T' = E(T)$. Si c'est vrai, démontrez-le, si c'est faux, donnez un contre-exemple.

   (c) (5 points) Je prétends que pour tout arbre T, il existe un entier $n \geq 0$ tel que $E^n(T) = E^{n+1}(T) = E^{n+2}(T) = \cdots$, de sorte que l'expression
$$\lim_{n \to \infty} E^n(T)$$

est bien définie. Montrez que mon affirmation est vraie et qu'il n'y a que deux arbres qui peuvent être obtenus comme une telle limite.

Solution: (a) C'est faux. Prenons par exemple les trois arbres ci-bas :



Clairement, on a $E(T_1) = E(T_2) = T_3$.

(b) C'est vrai. Par définition de E, on supprime toutes les feuilles de l'arbre, donc on n'a aucun choix à faire. De façon équivalente, on remarque que si L est l'ensemble des feuilles de T, alors $E(T) = T[V - L]$, qui est l'unique sous-graphe induit de T dont les sommets sont $V - L$.

(c) Tout d'abord, remarquons que si T possède une feuille, alors $E(T) \neq T$, puisqu'on supprime au moins un sommet. On cherche donc les arbres qui vérifient l'équation $E(T) = T$.

On remarque qu'il y a une solution triviale à cette équation, qui est l'arbre vide (ne l'oublions pas). De plus, observons que si T possède au moins deux sommets, alors il n'est pas possible que tous les sommets de T soient de degré 2 ou plus. Si c'était le cas, alors il serait possible de construire un cycle, contredisant le fait que T est acyclique (un arbre est acyclique).

Il ne reste que le cas où T possède un sommet de degré 0. Comme T est connexe (un arbre est toujours connexe), on en conclut que T est composé d'un seul sommet isolé.

En résumé, les deux seuls arbres possibles sont l'arbre vide T_0 et l'arbre d'un seul sommet T_1 qui peuvent être obtenus comme limite $\lim_{n\to+\infty} E^n(T)$. Clairement, cette expression est toujours bien définie puisque tout arbre ayant un nombre fini de sommets finira éventuellement par converger vers un de ces deux cas.

6. Soit $G = (V, E)$ un graphe non orienté de k composantes connexes. On dit que $\{u, v\} \in E$ est un *pont* si le nombre de composantes du graphe $G' = (V, E - \{u, v\})$ est strictement plus petit que k. Autrement dit, la suppression de l'arête $\{u, v\}$ fait augmenter le nombre de composantes connexes de G.

   Aussi, supposez que vous avez à votre disposition seulement les fonctions suivantes pour manipuler un graphe simple G :

   - G.SOMMETS() retourne l'ensemble des sommets de G;

- G.ARÊTES() retourne l'ensemble des arêtes de G;
- G.SUPPRIMER(u, v) retourne une copie du graphe G après avoir supprimé l'arête $\{u, v\}$
- G.VOISINS(u) retourne l'ensemble des sommets adjacents à u dans G;

(a) (10 points) Donnez le pseudocode d'une fonction

fonction NbComposantes(G : graphe simple) : naturel

qui retourne le nombre de composantes connexes de G. *Indice :* Une stratégie possible consiste à utiliser une procédure auxiliaire

procedure Visiter(G : graphe simple, u : sommet, *visité* : marquage)

qui explore tous les sommets dans la même composante connexe que u en les marquants comme visités (via le marquage *visité*).

Solution: Tel que suggéré dans la question, on utilise un marquage accessible depuis n'importe quelle fonction qu'on nomme *visité*. En programmation, on dit que *visité* est une variable globale. Il existe des astuces pour éviter les variables globales, qui sont généralement à éviter, mais pour simplifier la solution, nous n'en tiendrons pas compte.

```
fonction NbComposantes(G : graphe simple) : naturel
    pour u ∈ G.sommets() faire                ▷ On initialise le marquage
        visité[u] ← faux
    fin pour
    k ← 0
    pour u ∈ G.sommets() faire                ▷ On visite les sommets
        si ¬visité[u] alors
            k ← k + 1        ▷ On sait qu'on entame une nouvelle composante
            Visiter(G, u)
        fin si
    fin pour
    retourner k
fin fonction

procedure Visiter(G : graphe simple, u : sommet)
    si ¬visité[u] alors
        visité[u] ← vrai
        pour v ∈ G.voisins(u) faire   ▷ On visite récursivement les voisins
            Visiter(G, u)
        fin pour
```

```
    fin si
fin procedure
```

(b) (10 points) Donnez le pseudocode d'une fonction

$$\textbf{fonction } \textsc{PossèdePont}(G : \text{graphe simple}) : \text{booléen}$$

qui retourne vrai si et seulement si G possède un pont.

Solution: Il suffit de parcourir chaque arête et de vérifier si sa suppression change le nombre de composantes connexes. On peut en particulier réutiliser la fonction définie en (a). On trouve donc le pseudocode suivant.

```
fonction PossèdePont(G : graphe simple) : booléen
    k ← NbComposantes(G)
    pour {u, v} ∈ G.arêtes() faire
        G' ← G.supprimer(u, v)
        si NbComposantes(G') > k alors
            retourner vrai
        fin si
    fin pour
    retourner faux
fin fonction
```