Devoir 3

(à remettre au plus tard le 24 avril, à 16h00)
(dans la chute du département d'informatique, située au PK-4150)

Le devoir doit être rédigé **individuellement**. Vous devez **justifier** chacune de vos réponses. **Aucun retard** ne sera accepté puisque la solution sera publiée dans la soirée.
Notes : Préférez une impression recto-verso et agrafez votre devoir. Ne le mettez pas dans une enveloppe, pour éviter de consommer du papier inutilement. Aussi, si vous avez terminé le devoir en avance, vous pouvez me le remettre en mains propres au début ou à la fin du cours.

Question	1	2	3	4	5	6	Total
Sur	20	30	20	15	15	20	120
Note							

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 $P(8, 3)$.

   (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.

   (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.

   (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.

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 $\operatorname{sign} : \mathbb{R} \to \{-1, 0, 1\}$ définie par

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

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

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

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 $\rightarrow$, définie par

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

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

3. La relation $\Leftrightarrow$ définie par

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

   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.

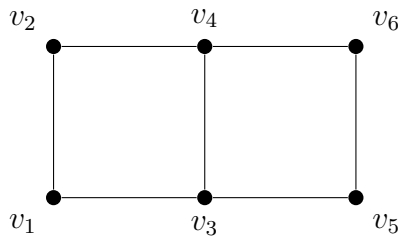
(d) (6 points) Pour chaque propriété parmi la réflexivité, l'irréflexivité, la transitivité, la symétrie, l'antisymétrie et l'asymétrie, indiquez si la relation $\rightarrow$ satisfait ou non la propriété. Dans chaque cas, justifiez.

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

(f) (5 points) Dessinez le graphe des relations $\rightarrow$ 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.

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 $n \geq 1$;

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

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

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

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

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.

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.

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*).

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

   fonction POSSÈDEPONT(G : graphe simple) : booléen

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