Nom ______________________________

Prénom ______________________________

Code permanent ______________________________

Solution de l'examen 2

Date : 29 mars 2017

Titre du cours : Mathématiques algorithmiques

Sigle : MAT1060

Enseignant : Alexandre Blondin Massé

Instructions

1) Vous avez trois heures pour répondre à l'examen ;
2) Vous avez droit à une feuille de notes recto-verso ;
3) Il est interdit d'utiliser un ordinateur, peu importe sa taille et sa forme (téléphone portable, agenda électronique, etc.) ;
4) Il est interdit de parler et de prêter de la documentation à un autre étudiant ;
5) Au besoin, utilisez le verso comme brouillon ;
6) À moins d'avis contraire, justifiez toutes vos réponses et donnez le détail de vos calculs ;
7) Indiquez clairement vos réponses finales ;

Question	1	2	3	4	5	6	7	Total
Sur	10	15	10	15	10	20	20	100
Note								

Question 1. .... (10 points)

Dans cette question, aucune justification n'est requise.

(a) (2 points) Soit A un ensemble défini récursivement par

(i) Si p est un nombre naturel premier, alors $p \in A$;

(ii) Si $m, n \in A$, alors $mn \in A$.

Énumérez tous les éléments de l'ensemble $\mathbb{N} - A$.

(a) $0, 1$**.**

(b) (2 points) Donnez une définition récursive par rapport à n de la fonction

$$f : \mathbb{R} \times \mathbb{N} : (x, n) \mapsto xn.$$

(b) $f(x,n) = \begin{cases} 0, & \textbf{si } n = 0\,\textbf{;} \\ x + f(x, n-1), & \textbf{sinon.} \end{cases}$.

(c) (2 points) Combien existe-t-il de sous-ensembles de $\{1, 2, 3, 4, 5, 6, 7\}$ dont la somme ne dépasse pas 7 ?

(c) 19**.**

(d) (2 points) Soit n un nombre naturel. Combien y a-t-il de mots sur l'alphabet $\{0, 1\}$ de longueur $2n$ qui contiennent autant de 0 que de 1 ?

(d) $C(n, 2n)$

(e) (2 points) Soit L une liste d'entiers. Que retourne la fonction Mystère(L) suivante ?

fonction Mystère(L : liste d'entiers) : entier ou $-\infty$
  si L est vide **alors**
    retourner $-\infty$
  sinon
    Soit x la tête de L (le premier élément de L)
    Soit L' la queue de L (la liste obtenue de L en supprimant x)
    $m \leftarrow$ Mystère(L')
    si $x > m$ **alors retourner** x
    sinon retourner m
  fin si
fin fonction

(e) **Le maximum de L.**

Question 2. .... (15 points)

Soit $A = \{0, 1\}$ l'alphabet binaire habituel. Rappelons que si w est un mot sur A, alors $\widetilde{w}$ est son image-miroir (le mot écrit à l'envers) et $\overline{w}$ est son complément (le mot obtenu en inversant les lettres). On définit un ensemble de mots E de façon récursive à l'aide des règles suivantes :

(i) $0 \in E$;

(ii) Si $w \in E$, alors $\overline{w} \in E$.

(iii) Si $w \in E$, alors $\widetilde{w} \in E$.

(iv) Si $w \in E$, alors $w\overline{w} \in E$;

De plus, soit E_n l'ensemble des mots de E obtenus en appliquant exactement n fois n'importe quelle suite de règles choisies parmi (ii), (iii) et (iv) (la même règle peut être appliquée plusieurs fois).

(a) (5 points) Énumérez tous les éléments de E de longueur au plus 8.

Solution: Voici les mots énumérés par ordre croissant de longueur :

Longueur	Mots
0	aucun
1	0, 1
2	01, 10
3	aucun
3	0110, 1001
4	aucun
5	aucun
6	aucun
7	aucun
8	01101001, 10010110

(b) (2 points) Vrai ou faux (justifiez brièvement) ? Tout mot $w \in E$ est un palindrome ou un antipalindrome. *Rappel :* w est un palindrome si et seulement si $w = \widetilde{w}$ et w est un antipalindrome si et seulement si $w = \overline{\widetilde{w}}$.

Solution: C'est vrai. Ça se montre par induction. Le cas de base est $w = 0$, qui est bien un palindrome. Ensuite, il suffit de remarquer que les règles (ii) et (iii) préservent les palindromes de même que les antipalindromes. Finalement, la règle (iv) transforme un palindrome en antipalindrome et un antipalindrome en palindrome.

(c) (8 points) Soit $n \geq 0$ un naturel. Montrez par induction simple sur n que pour tout mot $w \in E_n$, il existe un nombre naturel k tel que $|w| = 2^k$ (c'est-à-dire que la longueur de w est une puissance de 2). *Remarque :* Vous pouvez prendre pour acquis que pour tous mots u et v, on a $|uv| = |u| + |v|$, $|\widetilde{u}| = |u|$ et $|\overline{u}| = |u|$.

Solution: CAS DE BASE. Si $w \in E_0$, alors w est obtenu par la règle (i), de sorte que $w = 0$, qui est de longueur $1 = 2^0$.

HYPOTHÈSE D'INDUCTION. On suppose que pour tout mot $w' \in E_n$, on ait que $|w'| = 2^{k'}$ pour un certain naturel k'.

INDUCTION. Soit $w \in E_{n+1}$. Il y a trois cas possibles.

- w est obtenu par la règle (ii), c'est-à-dire que $w = \overline{w'}$ pour un certain $w' \in E_n$. Alors $|w| = |\overline{w'}| = |w'| = 2^{k'}$ pour un certain naturel k', par hypothèse d'induction.
- w est obtenu par la règle (iii), c'est-à-dire que $w = \widetilde{w'}$ pour un certain $w' \in E_n$. Ici aussi, $|w| = |\widetilde{w'}| = |w'| = 2^{k'}$ pour un certain naturel k', par hypothèse d'induction.
- w est obtenu par la règle (iv), c'est-à-dire que $w = w'\overline{w'}$ pour un certain $w' \in E_n$. Dans ce cas, $|w| = |w'\overline{w'}| = 2|w'| = 2 \cdot 2^{k'} = 2^{k'+1}$ pour un certain naturel k'.

Dans les trois cas, $|w|$ est bien une puissance de 2, ce qui termine la démonstration.

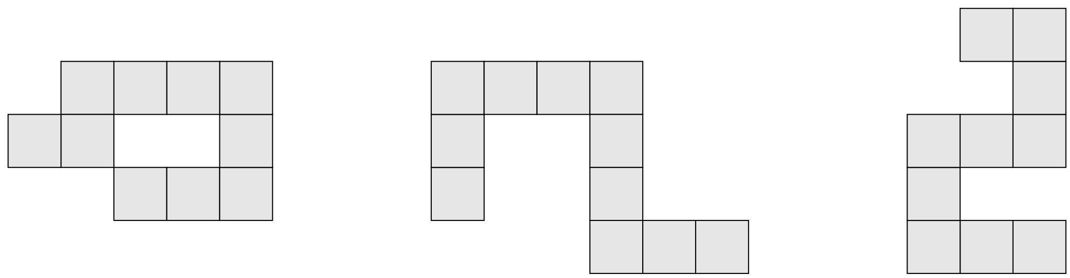
Question 3. .... (10 points)

Soit $n \geq 2$ un nombre naturel. Un *serpentmino de longueur n* est un ensemble de n carrés unités adjacents par **côté** (donc deux carrés unités qui partagent seulement un point ne sont pas adjacents) qui vérifie les conditions suivantes :

(i) Il existe exactement deux carrés unités qui sont adjacents à exactement un seul autre carré unité (on appelle ces deux carrés unités les *extrémités* du serpent) ;

(ii) Tous les autres carrés sont adjacents à exactement deux autres carrés unités.

Le *périmètre* d'un serpentmino est la longueur de son contour. Clairement, comme un serpentmino est formé de carrés unités, son périmètre est toujours un nombre naturel. Voici des exemples de serpentminos de périmètre 22, 24 et 22 respectivement :



Démontrez par induction (simple) sur n que le périmètre d'un serpentmino est $2n + 2$.

Solution: CAS DE BASE. Si $n = 2$, alors on a nécessairement un des deux serpentminos suivants :

qui ont tous les deux un périmètre de $6 = 2 \cdot 2 + 2$.

HYPOTHÈSE D'INDUCTION. On suppose que tout serpentmino de longueur n a un périmètre de $2n + 2$.

INDUCTION. Soit S un serpentmino de longueur $n+1$. Alors il existe un serpentmino S' de longueur n tel que S est obtenu en ajoutant un carré unité adjacent à une des deux extrémités de S'. Par hypothèse d'induction, le périmètre de S' est $2n + 2$. Comme le carré unité qu'on a ajouté augmente le nombre d'arêtes de 2 (on ajoute 3 arêtes au périmètre, mais une des arêtes devient intérieure), on en conclut que le périmètre de S est $2n + 2 + 2 = 2(n + 1) + 2$.

Question 4. .... (15 points)

Soit $n \geq 8$ un nombre naturel. L'objectif de cette question est de montrer qu'il existe deux nombre naturels a et b tels que $n = 3a + 5b$.

(a) (5 points) Montrez que la proposition est vraie pour $n = 8, 9, 10, 11, 12$.

Solution: On observe que

$$\begin{aligned} 8 &= 3 \cdot 1 + 5 \cdot 1, \\ 9 &= 3 \cdot 3 + 5 \cdot 0, \\ 10 &= 3 \cdot 0 + 5 \cdot 2, \\ 11 &= 3 \cdot 2 + 5 \cdot 1, \\ 12 &= 3 \cdot 4 + 5 \cdot 0. \end{aligned}$$

(b) (10 points) Montrez par induction généralisée sur n que la proposition est vraie.

Solution: Cas de base. Nous avons montré que la proposition est vraie pour $n = 8, 9, 10$ dans la partie (a).

Hypothèse d'induction. Supposons que pour tout nombre naturel n' tel que $8 \leq n' < n$, il existe des naturels a' et b' tels que $n' = 3a' + 5b'$.

Induction. Soit $n \geq 11$. Notons que $8 \leq n - 3 < n$ et donc on peut appliquer l'hypothèse d'induction à $n - 3$, c'est-à-dire qu'il existe des naturels a' et b' tels que $n - 3 = 3a' + 5b'$. Ceci entraîne $n = 3a' + 5b' + 3 = 3(a' + 1) + 5b'$. On a donc bien qu'il existe des naturels a et b tels que $n = 3a + 5b$, puisqu'il suffit de prendre $a = a' + 1$ et $b = b'$.

Question 5. .... (10 points)
Parmi les nombres de 1 à 300 (inclusivement), combien sont divisibles par 3, par 5 ou par 7 ?

Solution: Pour tout naturel positif n, $A(n)$ l'ensemble des nombres entre 1 et 300 qui sont divisibles par n. On cherche

$$|A(3) \cup A(5) \cup A(7)|.$$

Il ne reste qu'à calculer les cardinalités des intersections simples, doubles et triples des ensembles $A(3)$, $A(5)$ et $A(7)$. On trouve

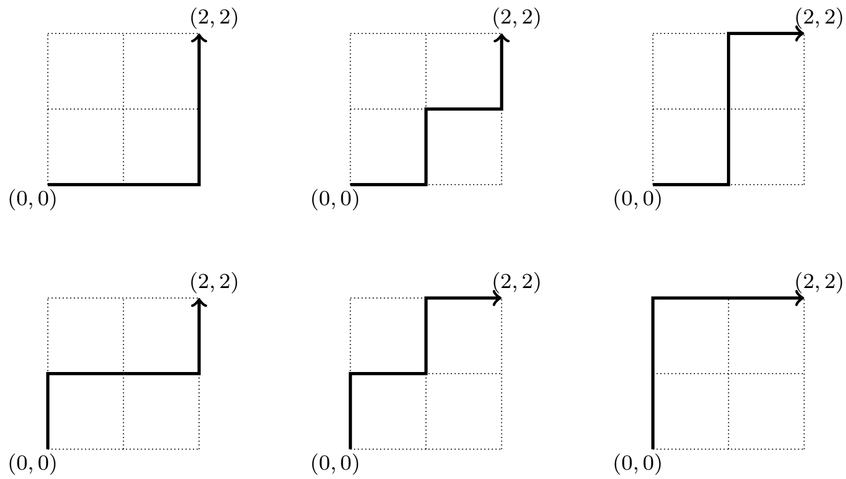
$$\begin{aligned}
|A(3)| &= |\{3, 6, 9, \ldots, 297, 300\}| = 100 \\
|A(5)| &= |\{5, 10, 15, \ldots, 295, 300\}| = 60 \\
|A(7)| &= |\{7, 14, 21, \ldots, 287, 294\}| = 42 \\
|A(3) \cap A(5)| &= |\{15, 30, 45, \ldots, 285, 300\}| = 20 \\
|A(3) \cap A(7)| &= |\{21, 42, 63, \ldots, 294\}| = 14 \\
|A(5) \cap A(7)| &= |\{35, 70, 105, 140, 175, 210, 245, 280\}| = 8 \\
|A(3) \cap A(5) \cap A(7)| &= |\{105, 210\}| = 2,
\end{aligned}$$

de sorte que

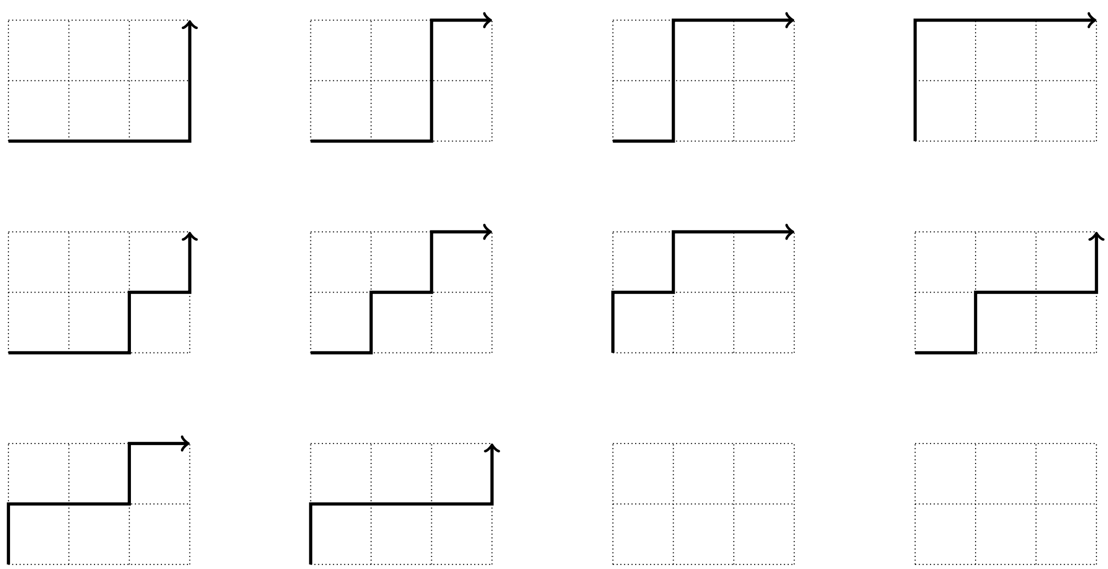
$$\begin{aligned}
|A(3) \cup A(5) \cup A(7)| &= |A(3)| + |A(5)| + |A(7)| \\
&\quad - (|A(3) \cap A(5)| + |A(3) \cap A(7)| + |A(5) \cap A(7)|) \\
&\quad + |A(3) \cap A(5) \cap A(7)| \\
&= 100 + 60 + 42 - (20 + 14 + 8) + 2 \\
&= 162.
\end{aligned}$$

Question 6. .... (20 points)

Soient $m, n \geq 0$ deux entiers. Dans cette question, on s'intéresse au nombre de chemins qu'on peut dessiner sur une grille de dimensions $m \times n$ du point en bas à gauche $(0, 0)$ jusqu'au point en haut à droite (m, n) en utilisant seulement des déplacements vers la droite et vers le haut. Par exemple, si $m = n = 2$, alors il existe 6 chemins :



(a) (6 points) Dessinez tous les chemins pour le cas $m = 3$ et $n = 2$. Utilisez les grilles ci-bas. Notez qu'il peut y en avoir moins que 12, même s'il y a 12 grilles qui sont dessinées.



(*suite à la page suivante*)

(*suite de la question 6*)

(b) (12 points) Soit $G(m, n)$ le nombre de chemins qui existent sur une grille de dimension $m \times n$. Donnez des arguments combinatoires qui justifient les égalités suivantes :

(i) $G(m, 0) = 1$ pour tout $m \geq 0$;

(ii) $G(0, n) = 1$ pour tout $n \geq 0$;

(iii) $G(m, n) = G(n, m)$ pour tout $m, n \geq 0$.

(iv) $G(m, n) = G(m - 1, n) + G(m, n - 1)$ pour tout $m, n \geq 1$.

Solution: Il y a un unique chemin de $(0, 0)$ vers $(m, 0)$, qui est parfaitement horizontal, donc $G(m, 0) = 1$.
Il y a un unique chemin de $(0, 0)$ vers $(0, n)$, qui est parfaitement vertical, donc $G(0, n) = 1$.
Le nombre de chemins sur une grille $m \times n$ est le même que sur une grille $n \times n$ puisque les déplacements vers la droite deviennent des déplacement vers le haut, et ceux vers le haut sont remplacés par des déplacements vers le haut, d'où $G(m, n) = G(n, m)$.
Tout chemin sur la grille $m \times n$ termine par le point $(m - 1, n)$ suivi du point (m, n) ou bien par le point $(m, n - 1)$ suivi du point (m, n). Noter que ces deux situations sont mutuellement exclusive, c'est-à-dire qu'exactement une de ces deux possibilités survient. Par conséquent, par le principe de la somme, nous avons $G(m, n) = G(m - 1, n) + G(m, n - 1)$.

(c) (2 points) Donnez une formule non récursive pour $G(m, n)$ (aucune justification requise).

Solution: On constate que

$$G(m, n) = \frac{(m + n)!}{m!n!}.$$

En effet, chaque chemin peut être représenté par un mot sur l'alphabet $\{d, h\}$ qui identifie les déplacements *droite* et *haut* respectivement. Il suffit donc de compter le nombre de mots qui contiennenet m lettres d et n lettres h. Comme il y a $C(m, m + n)$ façons de placer les m lettres d (et ensuite les lettres h ont un positionnement forcé), on obtient le résultat.

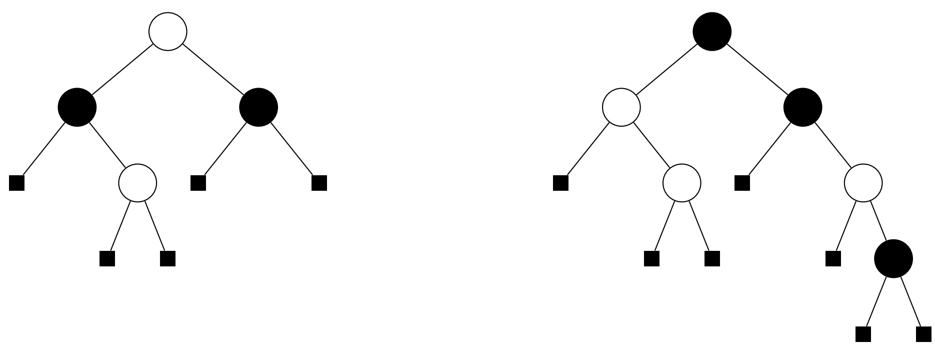
Question 7. .... (20 points)

Soit $C = \{blanc, noir\}$ l'ensemble des deux couleurs *noir* et *blanc*. On définit un arbre *blanc-noir* T récursivement comme suit :

(i) Soit $T = (\emptyset, noir)$, où $\emptyset$ dénote l'arbre *vide* (autrement dit, l'arbre vide est toujours noir) ;

(ii) Soit $T = (G, D, c)$, où G et D sont des arbres blanc-noir, appelés *sous-arbre gauche* et *sous-arbre droit* de T, et $c \in C$ est une couleur.

Voici des exemples d'arbres blanc-noir (les arbres vides sont identifiés par des petits carrés noirs) :



Vous pouvez supposer que pour tout arbre T, la notation suivante est disponible :

- $T.\textsc{estVide}()$ retourne vrai si et seulement si $T = (\emptyset, noir)$;
- Si $T.\textsc{estVide}()$, alors $T.\textsc{couleur}()$ retourne toujours *noir* ;
- Si $T = (G, D, c)$, alors $T.\textsc{gauche}()$, $T.\textsc{droit}()$ et $T.\textsc{couleur}()$ retournent G, D et c respectivement.

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

fonction $\textsc{NbNoeuds}(T$: arbre blanc-noir, c : couleur) : naturel

qui retourne le nombre de noeuds de couleur c dans l'arbre T.

Solution: Il suffit de compter récursivement le nombre de noeuds de couleur c dans les sous-arbres, puis d'ajouter 1 si le noeud courant est de couleur c. On trouve donc

fonction $\textsc{NbNoeuds}(T$: arbre blanc-noir, c : couleur) : naturel
    si $T.\textsc{estVide}()$ **alors**
        retourner 0
    sinon
        $n_c \leftarrow 1$ **si** $T.\textsc{couleur}() = c$ **sinon** $n_c \leftarrow 0$
        retourner $n_c + \textsc{NbNoeuds}(T.\textsc{gauche}()) + \textsc{NbNoeuds}(T.\textsc{droit}())$
    fin si
fin fonction

(b) (10 points) Soit T un arbre blanc-noir. On dit que T est *fortement noir* s'il n'existe pas de chemin de la racine vers une feuille contenant deux noeuds blancs consécutifs. À la page précédente, l'exemple de gauche est un arbre blanc-noir fortement noir, alors que le deuxième ne l'est pas, puisqu'on trouve deux noeuds blanc consécutifs en effectuant les déplacements *gauche*, *droit* à partir de la racine.

Donnez le pseudocode d'une fonction récursive

fonction EstFortementNoir(T : arbre blanc-noir) : booléen

qui retourne *vrai* si et seulement si T est fortement noir.

Solution: Noter qu'un arbre vide est par définition fortement noir. Il suffit donc de vérifier que pour tout noeud blanc, ses deux sous-arbres ont comme racine un noeud noir. On obtient donc ceci :

```
fonction EstFortementNoir(T : arbre blanc-noir) : booléen
    si T.estVide() alors
        retourner vrai
    sinon
        G ← T.gauche()
        D ← T.droit()
        si T.couleur() = blanc et G.couleur() = blanc alors
            retourner faux
        sinon si T.couleur() = blanc et D.couleur() = blanc alors
            retourner faux
        sinon
            retourner EstFortementNoir(G) et
                    EstFortementNoir(D)
        fin si
    fin si
fin fonction
```