EXAMEN INF301 2018
Algorithmique & Prog. Impérative
11 janvier 2019

Durée : 2h
Tous documents interdits
Une feuille A4 R/V manuscrite autorisée

Les questions faisant apparaître le symbole ♣ peuvent présenter zéro, une ou plusieurs bonnes réponses. Les autres ont une **unique bonne réponse**. ***Les réponses aux Questions à Choix sont à donner exclusivement sur la feuille de réponse (page 5 à détacher). Les réponses aux questions ouvertes sont à donner sur votre copie d'examen.***
Cet examen comporte 22 points mais sera noté sur 20. Le barème est indicatif et pourra être modifié.

Exercice 1 (Suite de Conway)

(Texte adapté de Wikipédia)
La suite de Conway est une suite mathématique inventée en 1986 par le mathématicien John Horton Conway (...) Le premier terme de la suite de Conway est posé comme égal à 1. Chaque terme de la suite se construit en énonçant le terme précédent, c'est-à-dire en indiquant combien de fois chacun de ses chiffres se répète.

1
11
21
1211
111221
312211
13112221
1113213211

Concrètement :

$X_0 = 1$: Ce terme comporte simplement un « 1 ». Par conséquent, le terme suivant est « un 1 », soit "11".

$X_1 = 11$: Celui-ci est composé de deux « 1 » : Le terme suivant sera donc « deux 1 », soit "21".

$X_2 = 21$: On a maintenant un « 2 » et un « 1 », le terme suivant sera alors « un 2 un 1 », soit "1211".

$X_3 = 1211$. En poursuivant le procédé, $X_4 = 111221$, $X_5 = 312211$, $X_6 = 13112221$, $X_7 = 1113213211\ldots$

3113 12 11 13 1221

Partie 1

Question 1 Quel est le terme suivant, X_8 ? [0.5 pt]
[A] 31231211131221 [B] 3113123111212 [C] 3131211131221 [D] 31131211131221
[E] Aucune réponse correcte. [F] Toutes les réponses sont correctes. [G] Manque de données dans l'énoncé. [H] La question est absurde.

Soit une séquence S représentée par tableau, initialement de longueur 1, avec S.tab[0] = 1, représentant ainsi X_0. Pour X_1, on aurait la longueur à 2, et en plus S.tab[1] = 1.

Question 2 On rajoute à la structure S un champ "indice" pour se souvenir de l'indice i du terme X_i. Donnez sous forme de dessin la représentation mémoire correspondant à X_4. Dessinez **au dos de la feuille de réponses** (page 6)

[0.5 pt]

Pour passer d'un terme au suivant, nous allons avoir besoin de compter, à partir d'une position, le nombre d'éléments identiques successifs, et d'effectuer des décalages dans la séquence soit pour faire de l'espace, soit pour le réduire.

Question 3 ♣ [2 pts]

Tout d'abord, cherchons une fonction `compte`(S, i), qui prend en argument une séquence et une position i (indice dans le tableau). Par exemple, sur X_4, `compte(S, 0)` $= 3$ (car il y a trois « 1 » à partir de la position 0), `compte(S, 3)` $= 2$, et `compte(S, 5)` $= 1$. Parmi les algorithmes suivants, cochez tous ceux qui proposent une implémentation correcte de `compte`. (On suppose que i est toujours un indice valide, et que S est non vide.)

A

compte (s, i)
- $j \leftarrow i + 2$
- **tant que** $j \leqslant s$.long et s.tab$[j-1] = s$.tab$[j-2]$ **faire**
  - $j \leftarrow j + 1$
- **retourner** $j - i - 1$

B

compte (s, i)
- $j \leftarrow i$
- **tant que** s.tab$[i] = s$.tab$[j]$ et $j < s$.long **faire**
  - $j \leftarrow j + 1$
- **retourner** $j - i$

faux, mauvais ordre, vérifier d'abord que j est valide

C

compte (s, i)
- **pour** j de $i+1$ à s.longueur-1 **faire**
  - **si** s.tab$[i] \neq s$.tab$[j]$ **alors**
    - **retourner** $j - i$

faux pour la dernière série de nombres identiques
↳ on ne retournera rien

Question 4 Intéressons nous maintenant au décalage. On veut une fonction `decale`$(S$, pos, $d)$, qui décale tous les éléments d'indices entre pos et S.longueur-1 (inclus) de d positions : vers la « fin » de S si d est positif, vers le « début » si d est négatif. Donner le pseudo-code de la fonction `decale`. [2 pts]

Nous pouvons maintenant écrire la fonction `Conway`(S), qui prend en argument une séquence S représentant un terme X_n de la suite de Conway, et transforme S pour qu'elle représente le terme suivant, X_{n+1}. Dans les questions suivantes, vous devrez compléter les « trous » de l'algorithme puis corriger les erreurs éventuelles.

$= O(k^2)$

Conway(S)
- **pour** i de 0 à S.longueur-1 **faire** $\times k$
  - val $\leftarrow S$.tab$[i]$ $O(1)$
  - n $\leftarrow$ compte (S, i) $O(k)$
  - decale (S, A: , B:) $O(k)$
  - s.tab$[i] \leftarrow$ C:
  - s.tab$[i+1] \leftarrow$ D: $\}O(1)$
- S.indice $\leftarrow S$.indice$+1$

Question 5 Que doit contenir la boîte A ? [1 pt]

A $i+n$ B $i+2$ C $i+1$ D $i+n+1$ E *Aucune réponse correcte.* F *Toutes les réponses sont correctes.* G *Manque de données dans l'énoncé.* H *La question est absurde.*

Question 6 Que doit contenir la boîte B ? [1 pt]

A $n-1$ B $n+1$ C $n-2$ D $2-n$ E *Aucune réponse correcte.* F *Toutes les réponses sont correctes.* G *Manque de données dans l'énoncé.* H *La question est absurde.*

Question 7 Que doit contenir la boîte C ? [1 pt]

A i B $i+1$ C n D val E *Aucune réponse correcte.* F *Toutes les réponses sont correctes.* G *Manque de données dans l'énoncé.* H *La question est absurde.*

il faudrait mettre val

Question 8 Que doit contenir la boîte D ? [1 pt]

[A] n [B] i [C] $n+1$ [D] $i+1$ [E] *Aucune réponse correcte.* [F] *Toutes les réponses sont correctes.* [G] *Manque de données dans l'énoncé.* [H] *La question est absurde.*

Question 9

La fonction `Conway` est-elle maintenant correcte ? Si oui, justifiez sur un exemple, si non, expliquez la/les erreurs et comment la/les corriger. [1 pt]

Question 10 [1 pt]

Quelle est la complexité de la fonction `Conway` ? (avec k le nombre de chiffres du terme X_n)

[A] $O(k^2)$ [B] $O(\log k)$ [C] $O(k)$ [D] $O(2^k)$ [E] *Aucune réponse correcte.* [F] *Toutes les réponses sont correctes.* [G] *Manque de données dans l'énoncé.* [H] *La question est absurde.*

compte et decale sont en $O(k)$, dans le corp d'une boucle qui fait $O(k)$ itérations

Partie 2

Dans cette partie, on cherche à améliorer la complexité de la fonction `Conway`. Voici à droite un tableau mesurant en secondes le temps de calcul du terme X_n de la suite de Conway à partir du terme X_{n-1}.

terme	secondes	terme	secondes
X_{20}	0.004	X_{29}	0.528
X_{21}	0.007	X_{30}	0.862
X_{22}	0.012	X_{31}	1.471
X_{23}	0.021	X_{32}	2.517
X_{24}	0.035	X_{33}	4.231
X_{25}	0.062	X_{34}	7.211
X_{26}	0.104	X_{35}	12.29
X_{27}	0.174	X_{36}	20.86
X_{28}	0.300	X_{37}	35.50

Question 11 Utilisez les informations de la question précédente et de ce tableau pour en déduire une propriété intéressante des termes de la suite de Conway. [2 pts]

Nous vous proposons d'utiliser une représentation de la séquence S avec une liste chaînée plutôt qu'avec un tableau. Comme dans la partie précédente, on veut modifier la séquence « en place », c'est-à-dire en modifiant S au fur et à mesure.

Proposez une solution pour le calcul des termes de la suite de Conway à base de listes chaînées, en justifiant leur utilisation (par rapport à la représentation par tableau et longueur). **Attention : au moins un des algorithmes utilisés devra être récursif !**

Consignes / indications :

- Limitez la création de cellules au strict nécessaire.
- Commencez par montrer sur un ou plusieurs exemples comment la structure de donnée sera modifiée en mémoire pour passer d'un terme au suivant. (Dessinez **au dos de la feuille de réponses**, page 6.)
- Expliquez en quelques phrases le principe de votre algorithme.
- Donnez le pseudo-code de votre algorithme. (**Attention :** séparez en plusieurs fonctions pour faciliter la compréhension et éviter les erreurs !)
- Faites une analyse de complexité.

	Dessin	[1.5 pts]
Barème indicatif	Explications	[1.5 pts]
	Pseudo-Code	[4 pts]
	Complexite	[2 pts]

Q4 : il ne faut pas écraser de valeurs par erreur.

décale(S, pos, d)
si S.longueur + d > LMAX alors erreur "plus de place"
si d > 0
  pour i de S.longueur − 1 à pos
    S.tab[i + d] ← S.tab[i]
si d < 0
  pour i de pos à S.longueur − 1
    S.tab[i + d] ← S.tab[i]
S.longueur ← S.longueur + d

attention au sens pour ne pas écraser les valeurs

Note : le décalage est nécessairement inférieur à la position (|d| < pos), donc inutile de vérifier si i+d < 0.

Q9 : même avec D = val, elle n'est pas correcte. Problèmes :
* chaque groupe de nombres est transformé en 2 nombres
  → il faut passer de i à i+2 et non i+1
* l'algorithme utilise une boucle "pour" alors qu'une des bornes change (la longueur) dans la boucle
  → il faut utiliser un "tant que"

Q11 Avec t_n le temps pour calculer X_{n+1} à partir de X_n,

on remarque $t_{n+1} \simeq 1{,}7\, t_n$

Or on sait que $t_n = O(K_n^2)$ avec K_n le nombre de chiffres de X_n

On en déduit $K_{n+1}^2 \simeq 1{,}7\, K_n^2$

Soit $K_{n+1} \simeq \sqrt{1{,}7}\, K_n$

Conclusion : le nombre de chiffres des termes de Conway augmente exponentiellement (suite géométrique de raison ≃ 1,3)

Feuille de réponses

Noircissez entièrement les cases.

Les réponses aux QCM sont à donner exclusivement sur cette feuille : les réponses données sur les feuilles précédentes ne seront pas prises en compte.

☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0
☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1
☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2
☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3
☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4
☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5
☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6
☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7
☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8
☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9

⟵ codez votre **numéro d'anonymat** ci-contre, et **ré-inscrivez le** ci-dessous.

↓

Numéro d'**anonymat** :

....

Question 1 : ☐A ☐B ☐C ☐D ☐E ☐F ☐G ☐H

Question 2 : .... dessin X_4 ☐W ☐I ☐P ☐C *Réservé*

Question 3 : ☐A ☐B ☐C

Question 4 : .... code decale ☐W ☐II ☐I ☐P ☐PP ☐C *Réservé*

Question 5 : ☐A ☐B ☐C ☐D ☐E ☐F ☐G ☐H

Question 6 : ☐A ☐B ☐C ☐D ☐E ☐F ☐G ☐H

Question 7 : ☐A ☐B ☐C ☐D ☐E ☐F ☐G ☐H

Question 8 : ☐A ☐B ☐C ☐D ☐E ☐F ☐G ☐H

Question 9 : .... Conway correction ☐W ☐I ☐P ☐C *Réservé*

Question 10 : ☐A ☐B ☐C ☐D ☐E ☐F ☐G ☐H

Question 11 : .... temps analyse ☐W ☐II ☐I ☐P ☐PP ☐C *Réservé*

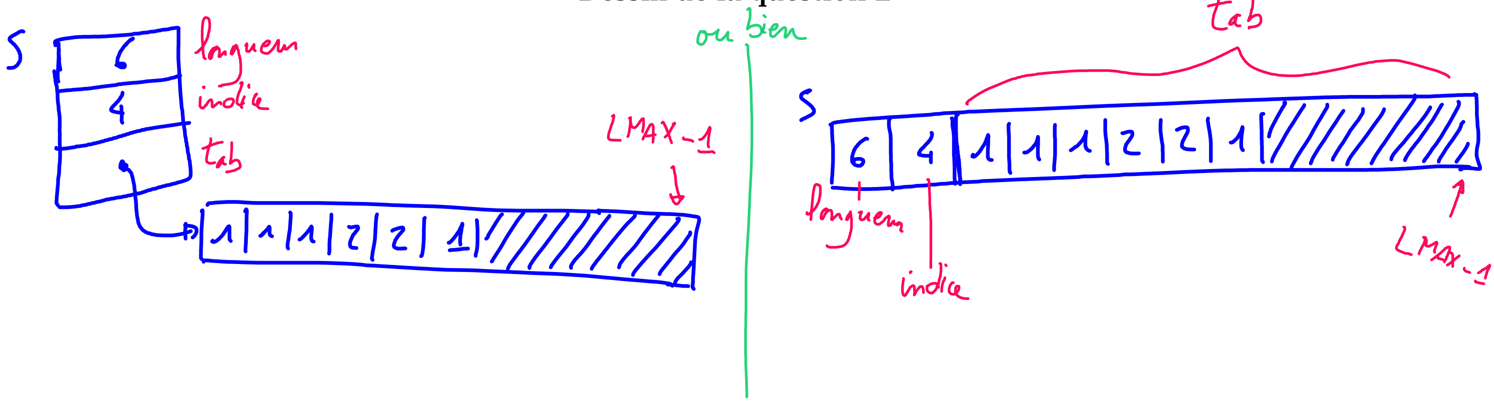
Question 12 : .... dessin ☐W ☐II ☐I ☐P ☐PP ☐C *Réservé*

Question 13 : .... explications ☐W ☐II ☐I ☐P ☐PP ☐C *Réservé*

Question 14 : .... pseudo-Code ☐W ☐II ☐I ☐P ☐PP ☐C *Réservé*

Question 15 : .... complexité ☐W ☐II ☐I ☐P ☐PP ☐C *Réservé*

Dessin de la question 2



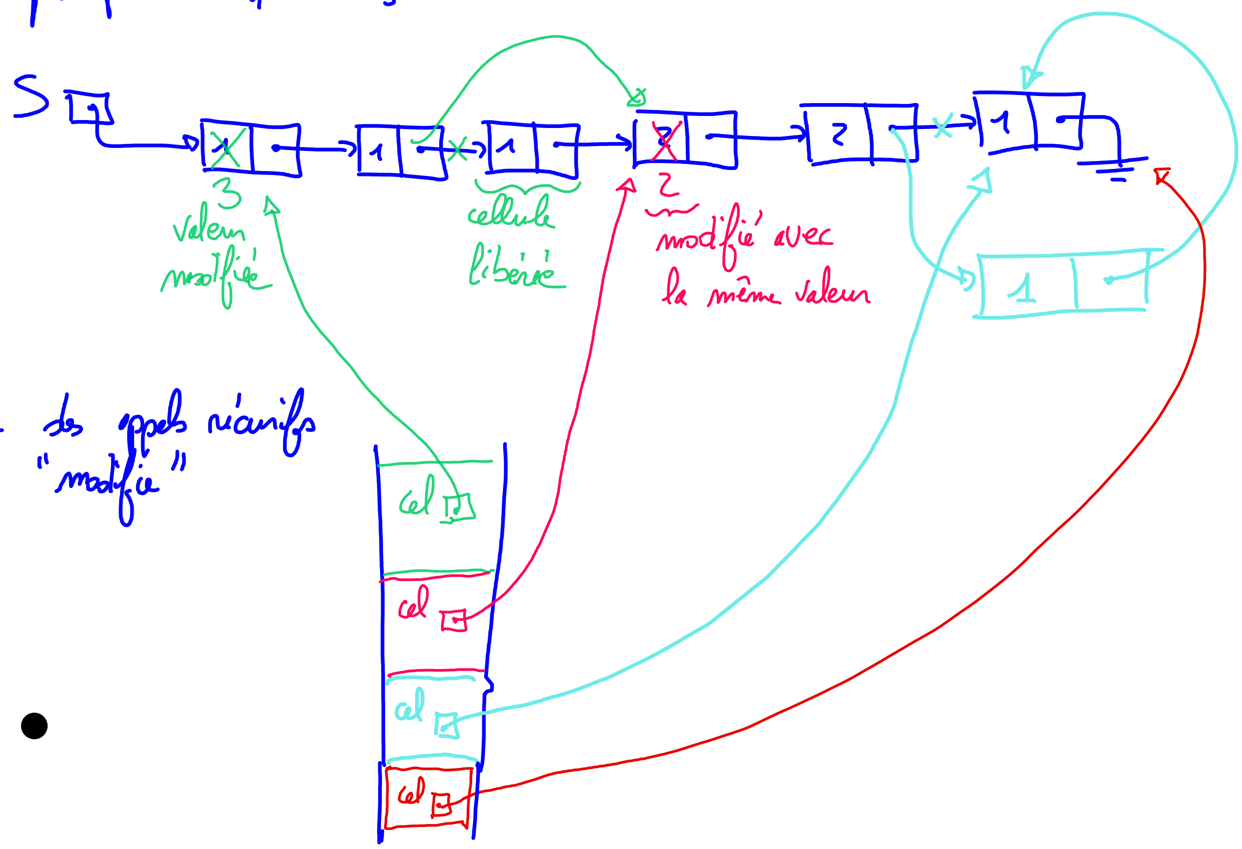
Dessin de la partie 2

Principe : on parcourt la liste chaînée. Pour chaque "bloc" de chiffres identiques :

— soit n le nombre de chiffres du bloc
— si $n = 1$ on crée une cellule avant
— si $n = 2$ on modifie la valeur de la première
— si $n \geqslant 3$ idem que $n = 2$, et on supprime $n-2$ cellules
(note : en fait $n \geqslant 3$ est impossible)

puis on continue récursivement sur le bloc suivant
On renvoie à chaque appel la première cellule de la liste courante.

Exemple pour X_4 à X_5



pile des appels récursifs de "modifie"

Note : il y a de nombreuses manières de résoudre le problème. En voici deux sur cette page et la suivante

```
Conway(S)
    S.Tete ← modifie(S.Tete)
    S.indice ← S.indice + 1

modifie(cel)
    si cel = Nil retourner Nil
    n ← compte(cel, cel.val)
    si n = 1
        ncel ← nouvelle cellule  ✗
        ncel.val ← 1
        ncel.suiv ← cel
        cel.suiv ← modifie(cel.suiv)
        retourner ncel
    sinon
        cel.val ← n
        next ← cel.suiv
        suite ← supprime(next.suiv, n-2)
        next.suiv ← modifie(suite)
        retourner cel
```

complexité : on passe au plus un nombre constant de fois par la même cellule : quand on traite un bloc, on va compter toutes les cellules, puis éventuellement en supprimer

↳ coût d'un appel de "modifie" $= O(\text{taille du bloc})$

$\Rightarrow$ on le fait sur tous les blocs

↳ $\sum_{b \in \text{blocs}} O(\text{taille de } b) = O(k)$

(avec k longueur de la liste chaînée)

max $O(t)$ appels récursifs, t = taille du bloc

```
compte(cel, v)
    si cel = Nil retourner 0        } O(1)
    si cel.val ≠ v retourner 0      }
    retourner 1 + compte(cel.suiv, v)
```

max $O(\text{Taille de bloc max})$ appels récursifs

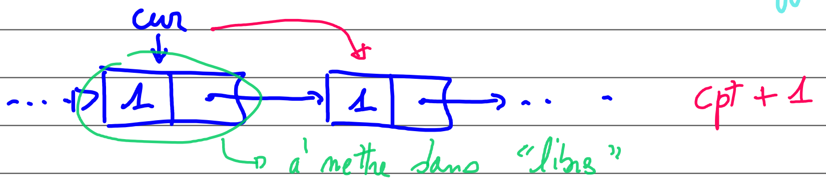
```
supprime(cel, x)
    si x = 0 retourner cel
    suite ← cel.suiv
    libérer(cel)  ✗
    retourner supprime(suite, x-1)
```

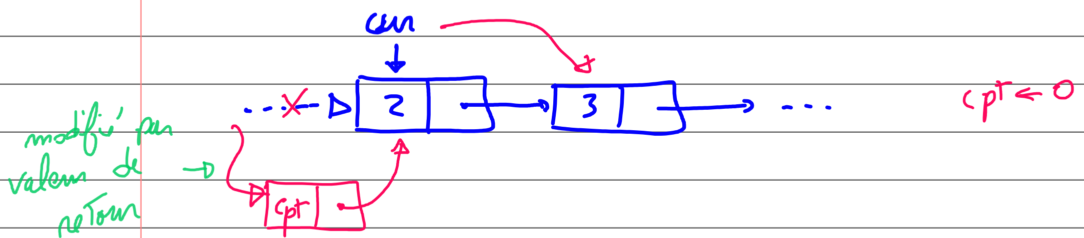
✗ note : au lieu de libérer, on pourrait l'ajouter en tête d'une liste chaînée globale « cellules_libres » dans laquelle on pioche en priorité quand on a besoin d'une nouvelle cellule.

Version alternative "en ligne" (ne fait qu'avancer en gardant des informations en mémoire)

Principe : un compteur du nombre d'occurrences du chiffre en cours, on regarde récursivement les deux premières cellules de la liste :
* valeurs égales ⇒ incrémenter et appel réc
* sinon (1 seule cellule ou valeurs différentes) ⇒ ajout cellule avec compteur et appel réc

Exemple :





Conway(S)
 libres ← nouvelle_séquence
 S.tete ← conway_rec(S.tete, 0, libres)
 S.indice ← S.indice + 1
 /* Note : libres est vide à la fin */

liste des cellules libres

Conway_rec(cur, cpt, libre)
 si cur = Nil retourner Nil
 next ← cur.suivant
 si next ≠ Nil et cur.val = next.val
  push(libres, cur)
  retourner conway_rec(next, cpt+1, libre)
 sinon
  num ← pop(libres)
  num.val ← cpt + 1
  num.suiv ← conway_rec(next, 0, libres)
  retourner num

push(P, cel)
 cel.suiv ← P.tete
 P.tete ← cel

pop(P)
 si P.tete = Nil
  retourner nouvelle cellule
 sinon
  cel ← P.tete
  P.tete ← cel.suiv
  retourner cel