Correction

EIDGENÖSSISCHE TECHNISCHE HOCHSCHULE – LAUSANNE
POLITECNICO FEDERALE – LOSANNA
SWISS FEDERAL INSTITUTE OF TECHNOLOGY – LAUSANNE

Faculté Informatique et Communications
Cours ICC aux sections MA et PH
Chappelier J.-C.

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

INFORMATIQUE, CALCUL & COMMUNICATIONS

Sections MA & PH

Correction Examen intermédiaire I

28 octobre 2016

SUJET 1

Instructions :

- Vous disposez d'une heure quinze minutes pour faire cet examen (15h15 - 16h30).
- L'examen est composé de 2 parties : un questionnaire à choix multiples, à 12 points, prévu sur 45 minutes, et une partie à questions ouvertes, à 8 points, prévue sur 30 minutes.
  Mais vous êtes libres de gérer votre temps comme bon vous semble.
- **AUCUN DOCUMENT N'EST AUTORISÉ, NI AUCUN MATÉRIEL ÉLECTRONIQUE**.
- Pour la première partie (questions à choix multiples), chaque question n'a qu'une seule réponse correcte parmi les quatre propositions.
  Indiquez vos réponses en bas de **cette** page en écrivant *clairement* pour chaque question une lettre majuscule parmi A, B, C et D. (Vous êtes autorisés à dégrafer cette page)
  Aucune autre réponse ne sera considérée, et en cas de rature, ou de toute ambiguïté de réponse, nous compterons la réponse comme fausse.
- Pour la seconde partie, répondez directement sur la donnée, à la place libre prévue à cet effet.
- Toutes les questions comptent pour la note finale.

Notations : dans cet examen, le premier élément d'une liste L est noté $L(1)$. Par ailleurs, la notation $\lfloor x \rfloor$ représente la « partie entière par défaut » de x, c.-à-d. le plus grand entier inférieur ou égal à x. Par exemple $\lfloor 1.5 \rfloor = 1$ et $\lfloor 1 \rfloor = 1$.

Réponses aux quiz :

Reportez ici *en majuscule* la lettre de la réponse choisie pour chaque question, sans aucune rature.

1	2	3	4	5	6	7	8	9	10	11	12
A	C	C	A	C	A	B	D	C	A	C	D

PARTIE QUIZ

Question 1) Quelle est la complexité respective de chacun des algorithmes suivants :

algo1.1
entrée : n, *entier naturel* sortie : *0*
Si $n = 0$ **sortir :** 0 **sortir : algo1.1**$(\lfloor n/2 \rfloor)$

algo1.2
entrée : n, *entier naturel* sortie : *0*
Si $n \leq 2$ **sortir :** 0 **sortir : algo1.2**$(\lfloor n/2 \rfloor)$ + **algo1.2**$(\lfloor n/2 \rfloor + 1)$

algo1.3
entrée : n, *entier naturel* sortie : *0*
Si $n \leq 1$ **sortir :** 0 **sortir : algo1.3**$(n-1)$ + **algo1.3**$(n-2)$

✔**A]** **algo1.1** : $\mathcal{O}(\log(n))$; **algo1.2** : $\mathcal{O}(n)$; **algo1.3** : $\mathcal{O}(2^n)$

B] **algo1.1** : $\mathcal{O}(n/2)$; **algo1.2** : $\mathcal{O}(n)$; **algo1.3** : $\mathcal{O}(2n)$

C] **algo1.1** : $\mathcal{O}(n/2)$; **algo1.2** : $\mathcal{O}(n^2)$; **algo1.3** : $\mathcal{O}(n^2)$

D] **algo1.1** : $\mathcal{O}(\log(n))$; **algo1.2** : $\mathcal{O}(n^2)$; **algo1.3** : $\mathcal{O}(n/2)$

Question 2) Quelle est la complexité de l'algorithme suivant :

algo2
entrée : n, m, *deux entiers naturels* sortie : *? ? ?*

Si $m < n$
  sortir : 21
Si $m \leq n^2$
  sortir : algo2$(m^2+1,\, n^2)$
sortir : algo2$(n+m,\, n)$ + **algo2**$(n+m,\, n^2+m^2)$

A] $\mathcal{O}(n^2+m^2)$ mais pas $\mathcal{O}(n+m)$

B] $\mathcal{O}(\min(n,m))$ mais pas $\mathcal{O}(1)$

✔**C]** $\mathcal{O}(1)$

D] aucune, l'algorihme pouvant ne pas terminer

Question 3) En utilisant une représentation signée sur 5 bits pour les nombres entiers, quelle est la valeur (décimale) si l'on ajoute 1 à chacun des nombres suivants : i : `01111` ; j : `10000` et k : `11111`.

A] i : -16 ; j : -17 ; k : 32.

B] i : 16 ; j : -16 ; k : aucune (erreur).

✔**C]** i : -16 ; j : -15 ; k : 0.

D] i : 16 ; j : -17 ; k : -16.

suite au dos ☞

Question 4) Quel est le résultat décimal de l'addition de -47 et -23 si l'on utilise une représentation sur 7 bits ?

✔**A]** 58 **B]** -70 **C]** 70 **D]** -6

Question 5) Que retourne l'algorithme suivant, sachant que l'on note la valeur de x en virgule flotante avec 3 bits pour l'exposant et 5 bits pour la mantisse ?

algo5

entrée : *(rien)*
sortie : *? ? ?*

```
x ⟵ 0 001 00010
Pour i de 1 à 2
    x ⟵ x × 2
sortir : x
```

A] `0 001 00010` **B]** `0 100 00010` ✔**C]** `0 011 00010` **D]** `0 100 01000`

Question 6) Les différents états successifs de L dans l'algorithme suivant :

algo6

entrée : L *liste de nombres*
sortie : *? ? ?*

```
l ⟵ taille(L)
a ⟵ ⌊l/2⌋
Tant que a ≥ 1
    Pour b de a + 1 à l
        c ⟵ b − a
        Tant que c > 0
            Si L(c) > L(c + a)
                échanger L(c) et L(c + a)
                c ⟵ c − a
            Sinon
                c ⟵ 0
    a ⟵ ⌊a/2⌋
sortir : L
```

appliqué à l'entrée $(9, 8, 3, 4, 6)$ sont :

✔**A]** $(9,8,3,4,6)$, $(3,8,9,4,6)$, $(3,4,9,8,6)$, $(3,4,6,8,9)$
B] $(9,8,3,4,6)$, $(3,4,9,8,6)$, $(3,4,6,9,8)$, $(3,4,6,8,9)$
C] $(9,8,3,4,6)$, $(6,8,3,4,9)$, $(3,8,6,4,9)$, $(3,4,6,8,9)$
D] $(9,8,3,4,6)$, $(9,3,4,6,8)$, $(3,9,4,6,8)$, $(3,4,9,6,8)$, $(3,4,6,9,8)$, $(3,4,6,8,9)$

Question 7) En notant n la taille de la liste L et en faisant une hypothèse raisonnable sur **taille**, quelle est la complexité de l'algorithme suivant :

```
algo7
entrée : liste L
sortie : ? ? ?
    M ⟵ liste de cent zéros
    t ⟵ taille(L)
    Pour i de 1 à t
        v ⟵ L(i)
        Si 1 ≤ v ≤ 100
            M(v) ⟵ M(v) + 1
    sortir : M
```

A] $\mathcal{O}(n \log(n))$ mais pas $\mathcal{O}(n)$

✔**B]** $\mathcal{O}(n)$ mais pas $\mathcal{O}(\log(n))$

C] $\mathcal{O}(\log(n))$ mais pas $\mathcal{O}(1)$

D] $\mathcal{O}(n^2)$ mais pas $\mathcal{O}(n \log(n))$

Question 8) Que calcule l'algorithme suivant :

```
algo8
entrée : E1, E2, deux ensembles non vides
sortie : ? ? ?
    t1 ⟵ taille(E1)
    t2 ⟵ taille(E2)
    Si t1 > t2
        échanger E1 et E2
        t1 ⟵ t2
    Si t1 = 1
        Si E1 ⊂ E2
            sortir : 1
        Sinon
            sortir : 0
    Partitionner E1 en deux parties (distinctes, non vides) : E11 et E12
    Partitionner E2 en deux parties (distinctes, non vides) : E21 et E22
    sortir : algo8(E11, E21) + algo8(E11, E22) + algo8(E12, E21) + algo8(E12, E22)
```

A] la cardinalité de l'union de E_1 et E_2

B] l'indice du premier élément de E_1 qui est également présent dans E_2

C] le nombre de fois que le premier élément de E_1 appartient à E_2.

✔**D]** la cardinalité de l'intersection de E_1 et E_2

Question 9) Lesquels des ensembles suivants sont dénombrables :

a. rationnels ($\mathbb{Q}$)

b. intervalle $[1, 2]$ (dans $\mathbb{R}$)

c. nombres pairs (positifs)

d. fonctions booléennes

e. nombres complexes ($\mathbb{C}$)

A] a, c et d **B]** uniquement c ✔**C]** a et c **D]** aucun des trois autres choix

suite au dos ☞

Question 10) Considérons deux problèmes de décision, « P_1 » et « P_2 ». Nous savons que « P_1 » est dans NP. Par rapport au problème de « l'arrêt de programmes » vu en cours, que peut-on dire ?

✔**A]** que le problème de « l'arrêt de programmes » est au moins aussi difficile que « P_2 »

B] que toute solution du problème de « l'arrêt de programmes » est facilement vérifiable

C] que « P_1 », « P_2 » et « l'arrêt de programmes » sont de même difficulté

D] que « P_1 » est plus difficile que le problème de « l'arrêt de programmes »

Question 11) Avec les mêmes problèmes « P_1 » et « P_2 » que dans la question précédente, si l'on arrive à montrer que « P_1 » est au moins aussi difficile que « P_2 », que peut-on en déduire ?

A] que « P_1 » est dans P

B] que « P_2 » est dans P

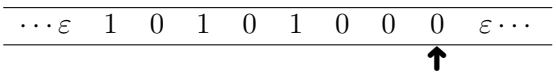
✔**C]** que « P_2 » est dans NP

D] rien du tout

Question 12) On considère la machine de Turing dont la table de transition est :

	0	1	ε
1	$(1, 0, -)$	$(2, 1, -)$	$(1, \varepsilon, -)$
2	$(2, 1, -)$	$(2, 0, -)$	$(3, \varepsilon, +)$

Quel est l'état de la bande lorsque la machine s'arrête, si elle a démarré dans l'état 1 avec sa tête de lecture positionnée comme suit :



A] $\cdots\varepsilon\ 0\ 1\ 0\ 1\ 0\ 1\ 1\ 1\ \varepsilon\cdots$

B] $\cdots\varepsilon\ 1\ 0\ 1\ 0\ 1\ 0\ 0\ 0\ \varepsilon\cdots$

C] $\cdots\varepsilon\ 0\ 1\ 0\ 1\ 0\ 0\ 0\ 0\ \varepsilon\cdots$

✔**D]** $\cdots\varepsilon\ 0\ 1\ 0\ 1\ 1\ 0\ 0\ 0\ \varepsilon\cdots$

(vous pouvez utiliser cette partie pour répondre aux exercices suivants, mais veuillez s.v.p. préciser le numéro de la question traitée)

PARTIE EXERCICES

1 – Encore des listes... [4 points]

Soient L_1 et L_2, deux listes de nombres entiers, pas forcément distincts mais ordonnés, telles que par exemple $L_1 = (-7, 0, 8, 9)$ et $L_2 = (-4, -3, 2, 4, 4, 5, 12)$.

Question 13) **[1 point]** Ecrivez un algorithme qui prend deux telles listes en entrée et retourne le plus petit écart [1] entre un nombre de L_1 et un nombre de L_2.

Pour l'exemple ci-dessus, le résultat serait 2 (écart entre $L_1(2) = 0$ et $L_2(3) = 2$).

Question 14) **[1 point]** Déterminez la complexité de votre algorithme. Justifiez votre réponse.

Question 15) **[1 point]** Si ce n'est pas déjà le cas, proposez un algorithme en $\mathcal{O}(n_1 + n_2)$, où n_i est la taille de L_i. [2]

Question 16) **[1 point]** Si l'on supprime l'hypothèse que les listes sont triées, quelle est la meilleure complexité pour résoudre le problème ? Décrivez brièvement comment atteindre cette complexité.

Réponses :

13. L'algorithme en $\mathcal{O}(n_1 \times n_2)$ devrait être facile à écrire :

écart minimal 1
entrée : L_1, L_2, *deux listes de nombres triés* sortie : *plus petit écart entre* L_1 *et* L_2

$n_1 \longleftarrow$ **taille**(L_1)
$n_2 \longleftarrow$ **taille**(L_2)
$m \longleftarrow \infty$
Pour i de 1 à n_1
  Pour j de 1 à n_2
  Si $m > \lvert L_1(i) - L_2(j) \rvert$
  $m \longleftarrow \lvert L_1(i) - L_2(j) \rvert$
sortir : m

1. i.e. la plus petite valeur absolue de différence
2. Et si c'était déjà le cas, sautez cette question, vous avez déjà ce point.

Pour l'algorithme en $\mathcal{O}(n)$: parcourir les listes en stockant l'écart minimum et en incrémentant l'index de L_1 si $L_1(i) < L_2(j)$, celui de L_2 dans le cas contraire :

écart minimal 2

entrée : L_1, L_2, *deux listes de nombres triés*
sortie : *plus petit écart entre* L_1 *et* L_2

```
n1 ⟵ taille(L1)
n2 ⟵ taille(L2)
m ⟵ ∞
i ⟵ 1
j ⟵ 1
Tant que i <= n1 et j <= n2
  Si m > |L1(i) − L2(j)|
    m ⟵ |L1(i) − L2(j)|
  Si L1(i) < L2(j)
    i ⟵ i + 1
  Sinon
    j ⟵ j + 1
sortir : m
```

16. $\mathcal{O}(n_1 \log(n_1) + n_2 \log(n_2) + n_1 + n_2) = \mathcal{O}(m \log(m))$ où $m = \max(n_1, n_2)$: trier d'abord, utiliser ensuite l'algo linéaire précédent.

2 – Toujours des listes... [4 points]

Pour une liste L, on notera $L(i:j)$ la sous-liste de L composée de $L(i), \cdots, L(j)$ (avec $i \leq j$; la liste vide sinon) . Par ailleurs, pour une liste L et un élément e, la notation $e \oplus L$ désignera la liste constituée de e (en premier) puis des éléments de L : $(e, L(1), L(2), L(3), \cdots)$.

Considérons les deux algorithmes suivants :

machin

entrée : L_1, L_2, *deux listes de nombres triés*
sortie : *? ? ?*

$n_1 \longleftarrow$ **taille**(L_1)
$n_2 \longleftarrow$ **taille**(L_2)
Si $n_1 = 0$
  sortir : L_2
Si $n_2 = 0$
  sortir : L_1
Si $L_1(1) < L_2(1)$
  sortir : $L_1(1) \oplus$ **machin**$\big(L_1(2:n_1),\, L_2\big)$
sortir : $L_2(1) \oplus$ **machin**$\big(L_1,\, L_2(2:n_2)\big)$

truc

entrée : L, *une liste de nombres*
sortie : *? ? ?*

$n \longleftarrow$ **taille**(L)
Si $n \leq 1$
  sortir : L
sortir : **machin**$\Big($**truc**$\big(L(1:\lfloor \frac{n}{2} \rfloor)\big),$
  truc$\big(L(\lfloor \frac{n}{2} \rfloor + 1 : n)\big)\Big)$

Question 17) **[0.5 point]** Que vaut **machin**$\big((23,27),(9,12,42)\big)$?

Question 18) **[0.5 point]** Que vaut **truc**$\big(27,23,9,42,12\big)$?

Question 19) **[1.5 points]** Récrire **machin** de façon *non* récursive.

Vous pouvez utilisez des instructions telles que : « *supprimer le premier(/dernier) élément de* L », « *ajouter (les éléments de)* L_1 *à la fin de* L_2 », etc.

Question 20) **[1.5 points]** En supposant **taille** en $\mathcal{O}(1)$, quelle est la complexité de **truc** ? Justifiez votre réponse.

Réponses 17 et 18 : $\big(9,12,23,27,42\big)$.

Réponse 20 : La complexité de **machin** est linéaire en la somme des tailles des deux listes.

Pour **truc**, on a donc

$$C(n) = a + 2\,C(n/2) + n$$

le n final venant de `machin` : la taille totale des deux sous-listes traitées étant bien n.

Ce qui va nous donner :

$$\begin{aligned} C(n) &= a + 2\,C(n/2) + n \\ &= 3a + 4\,C(n/4) + 2\,n \\ &= 7a + 8\,C(n/8) + 3\,n \\ &\cdots \\ &= (n-1)\cdot a + n\,C(1) + \log_2(n)\cdot n \end{aligned}$$

truc est donc en $\mathcal{O}\,(n\log(n))$. **Note :** il s'agit du *tri fusion*.

Réponse 19 : voici une version *possible*[3] :

machin
entrée : L_1, L_2, *deux listes de nombres* sortie : *fusion ordonée de* L_1 *et* L_2

$L \longleftarrow ()$ // *liste vide*
Tant que L_1 et L_2 ne sont pas (les deux) vides
  Si L_1 est vide
  ajouter L_2 à la fin de L
  sortir : L
  Si L_2 est vide
  ajouter L_1 à la fin de L
  sortir : L
  Si $L_1(1) < L_2(1)$
  ajouter $L_1(1)$ à la fin de L
  supprimer le premier élément de L_1
  Sinon
  ajouter $L_2(1)$ à la fin de L
  supprimer le premier élément de L_2
sortir : L

On pourrait aussi utiliser deux (voire trois) indices, un pour avancer dans L_1 et un pour L_2 (voire un pour L).

3. On peut aussi traiter les listes vides hors de la boucle, faire plus compliqué, ...