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 de l'examen intermédiaire I

27 octobre 2017

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 $L[1]$.

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	B	C	B	B	A	D	D	C	A	B	A

PARTIE QUIZ

Question 1) On considère la machine de Turing dont la table de transition est :

	0	1	ε
1	(1, 1, +)	(1, 0, +)	(2, ε, −)
2	(2, ε, −)	(3, 1, +)	(3, 0, −)

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 :

$\cdots\varepsilon$	0	1	1	1	0	1	1	1	$\varepsilon\cdots$
	↑								

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

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

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

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

Question 2) Supposons que l'on sache qu'un problème décidable nommé « CASTOR » <u>n</u>'est <u>pas</u> dans NP. Laquelle des affirmations suivantes est vraie ?

A] « CASTOR » est dans P.

✔**B]** Vérifier qu'une solution du problème « CASTOR » est effectivement une solution, prend un temps *plus que* polynomial par rapport à la taille de cette solution.

C] Vérifier qu'une solution du problème « CASTOR » est effectivement une solution, prend un temps *au plus* polynomial par rapport à la taille de cette solution.

D] Aucune des trois autres affirmations n'est vraie.

Question 3) Supposons que l'on connaisse un algorithme de complexité en $\mathcal{O}\left(4n^3 - 3n^2\right)$ permettant de trouver la solution d'un problème de décision appelé « POLLUX » portant sur des données de taille n. Que peut-on en déduire ?

A] « POLLUX » n'est pas dans P.

B] Rien du tout.

✔**C]** « POLLUX » est dans NP.

D] « POLLUX » n'est pas dans NP.

Question 4) Si l'on interprète les schémas binaires comme des nombres uniquement *positifs*, quel est le schéma résultant (sur 8 bits) de l'addition de `10010110` et `11010110` ?

A] `01101110` ✔**B]** `01101100` **C]** `101110100` **D]** `11101100`

Question 5) Même question, mais si l'on interprète les schémas binaires comme des nombres *signés* (positifs ou négatifs) :

A] `10010100` ✔**B]** `01101100` **C]** `101110100` **D]** `00010011`

Question 6) Combien de bits sont différents entre les représentations binaires de 148 et 94 (représentations de nombres entiers positifs sur au moins 8 bits) ?

✔**A]** 4 **B]** 2 **C]** 7 **D]** 6

suite au dos ☞

Question 7) On considère l'algorithme suivant :

algo7

entrée : a *nombre entier strictement positif*

sortie : *? ? ?*

Si $a = 1$
  Sortir : 1

Si a est impair
  Sortir : algo7$(a + 1) - 2 \times a - 1$

Sinon
  Sortir : $4\times$ **algo7**$(a/2)$

Laquelle des affirmations suivantes est correcte ?

- **A]** L'algorithme ne termine pas pour certaines valeurs de a.
- **B]** Sa sortie vaut $2a$ si a est pair et $-a$ si a est impair plus grand que 1.
- **C]** Sa sortie croit exponentiellement avec a.
- ✔**D]** Sa sortie n'est jamais plus grande que a^2.

Question 8) Pour une liste L et un élément e, la notation « $e \oplus L$ » désigne la liste constituée de e (en premier) puis des éléments de L : $(e, L[1], L[2], L[3], \cdots)$.

Quelle est la sortie de l'algorithme suivant sur l'entrée $L = (6, 3, 9, 6, 9, 2)$:

algo8

entrée : L *liste de valeurs*

sortie : *? ? ?*

$t \longleftarrow$ **taille**(L)

Si $t \leq 1$
  Sortir : L

Pour i de 2 à t
  Si $L[1] = L[i]$
    Sortir : algo8$(L[2], \ldots, L[t])$

Sortir : $L[1] \oplus$ **algo8**$(L[2], \ldots, L[t])$

A] $(6, 3, 9, 6, 9, 2)$ **B]** $(2, 3, 6, 9)$ **C]** $(2, 3, 6, 6, 9, 9)$ ✔**D]** $(3, 6, 9, 2)$

Question 9) Si l'on note n la taille de la liste L, quelle est la complexité de l'algorithme de la question précédente ?

- **A]** $\mathcal{O}(2^n)$ mais pas $\mathcal{O}(n^2)$.
- **B]** $\mathcal{O}(n)$ mais pas $\mathcal{O}(1)$.
- ✔**C]** $\mathcal{O}(n^2)$ mais pas $\mathcal{O}(n)$.
- **D]** Aucune, l'algorihme pouvant ne pas terminer.

Question 10) Un algorithme a besoin de 2^n instructions élémentaires pour calculer une fonction $f(n)$ donnée. Sachant qu'un humain utilisant (uniquement) cet algorithme calcule $f(30)$ en 8 minutes, combien de minutes faudrait-il à cette même personne pour calculer $f(37)$ (en repartant du début) ?

✔**A]** 1024 **B]** $\mathcal{O}(2^{37})$ **C]** 16 **D]** environ 10

Question 11) Que calcule l'algorithme suivant :

```
algo11
entrée : Une liste L non-vide de nombres entiers
sortie : ? ? ?

t ⟵ taille(L)
u ⟵ L[1]
v ⟵ L[1]
Pour i de 2 à t
    Si u > L[i]
        v ⟵ u
        u ⟵ L[i]
    Sinon
        Si v > L[i] ou i = 2
            v ⟵ L[i]
Sortir : (u, v)
```

A] La plus petite et la plus grande valeur de L.

✔**B]** Les deux plus petites valeurs de L.

C] Les deux plus grandes valeurs de L.

D] Les positions du plus petit et du plus grand élément de L.

Question 12) Laquelle des affimations suivantes est vraie pour un algorithme qui opére sur une liste L de n éléments et qui contient une boucle sur L dans une autre boucle sur L ?

✔**A]** Sa complexité est au moins en $\mathcal{O}\left(n^2\right)$.

B] Sa complexité est en $\mathcal{O}\left(n^2\right)$ mais pas en $\mathcal{O}\left(n\log(n)\right)$.

C] Sa complexité est en $\mathcal{O}\left(2^n\right)$ mais pas en $\mathcal{O}\left(n^3\right)$.

D] Il fait toujours plus d'étapes qu'un algorithme sans boucle.

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

suite au dos ☞

PARTIE EXERCICES

1 – Ecrire un algorithme [4 points]

Question 13) **[2 points]** Ecrivez un algorithme qui prend une liste de nombres en entrée et retourne la plus grande valeur de celle-ci ainsi que le nombre de fois que cette valeur maximale apparait dans la liste.

Par exemple, pour la liste $(12, 42, -4, 33, -42, 5, 42, 12, 3, 12)$, le résultat serait $(42, 2)$ car 42 est la valeur maximale et qu'elle apparait 2 fois.

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 [1], proposez un autre algorithme qui résoud le même problème mais en ne parcourant qu'*une et une seule fois* la liste (et sans en faire de copie, bien sûr !).

Réponses :

maxima

entrée : L
sortie : *le maximum de L et combien de fois il apparait*

```
m ⟵ −∞
c ⟵ 0
n ⟵ taille(L)
Pour i de 1 à n
    Si m < L(i)
        m ⟵ L(i)
        c ⟵ 1
    Sinon, si m = L(i)
        c ⟵ c + 1
Sortir : (m, c)
```

Notes :

- Attention au cas de la liste vide !
- Attention au « **Sinon** », ou alors inverser les deux branchements conditionnels ou encore réinitialiser c à 0 au lieu de 1.
- Ne pas initialiser m à 0 (ou toute autre valeur) : la liste peut contenir des nombres négatifs.

La complexité de l'algorithme ci-dessus est en $\mathcal{O}(n)$, où n est la taille de la liste (précisez vos notations !!).

Notes :

- L'hypothèse raisonnable que **taille** est en $\mathcal{O}(n)$ (ce qui inclut $\mathcal{O}(1)$!!) suffit pour conclure.
- Faire appel à un algorithme de tri introduit une complexité en $\mathcal{O}(n \log n)$, ce qui ne peut en aucun cas satisfaire les contraintes de la question 15.

suite au dos ☞

1. Si c'est déjà le cas et que votre algorithme est correct, ignorez cette question ; vous avez déjà ce point.

2 – Comprendre un algorithme [4 points]

Considérez l'algorithme suivant :

Mystère

entrée : *Une liste L de nombres, un nombre x et deux nombres entiers positifs $u (\neq 0)$ et v*

sortie : *? ? ?*

$t \longleftarrow$ **taille**(L)
Si $(v > t)$ **ou** $(v < u)$
  Sortir : 0
Si $v = u$
  Si $L[v] = x$
    Sortir : 1
  Sinon
    Sortir : 0
Sortir : Mystère$(L, x, u, u + \lfloor \frac{v-u}{2} \rfloor)$ + **Mystère**$(L, x, u + \lfloor \frac{v-u}{2} \rfloor + 1, v)$

où 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 $\lfloor 1 \rfloor = 1$.

Question 16) **[1 point]** Quelle est la sortie de cet algorithme pour l'entrée :
$L = (12, 42, -4, 33, -42, 5, 42, 12, 33, 17)$, $x = 33$, $u = 1$ et $v = 10$?

Question 17) **[1 point]** De façon générale, que calcule cet algorithme ?

Il **ne** s'agit **pas** ici de paraphraser l'algorithme mais bien de montrer *conceptuellement* ce qui est calculé par cet algorithme. *Une* seule phrase en français, bien choisie, devrait suffire.

Question 18) **[2 points]** Proposez une version non-recursive de cet algorithme.

Réponses :

16. 2
17. Le nombre de fois que x est présent dans L entre les rangs u et v.
18.

Mystère

entrée : *Une liste L de nombres, un nombre x et deux nombres entiers positifs $u (\neq 0)$ et v*

sortie : *Le nombre de fois que x est présent dans L entre les rangs u et v*

Si $v >$**taille**(L)
  Sortir : 0
$c \longleftarrow 0$
Pour i de u à v
  Si $x = L(i)$
    $c \longleftarrow c + 1$
Sortir : c