REPUBLIQUE TUNISIENNE

Ministère de l'Enseignement Supérieur,
de la Recherche Scientifique, des Technologies de
l'Information et de la Communication

Concours Nationaux d'Entrée
aux Cycles de Formation d'Ingénieurs
Session 2014

الجمهورية التونسية

وزارة التعليم العالي والبحث العلمي
وتكنولوجيا المعلومات و الاتصال

المناظرات الوطنية للدخول
إلى مراحل تكوين المهندسين
دورة 2014

Concours Toutes Options
Epreuve d'Informatique

Date : Mardi 03 Juin 2014 **Heure : 14 H** **Durée : 2 H** **Nombre de pages : 6**

Barème : PROBLEME 1 (MAPLE) : 10 points
PROBLEME 2 (ALGORITHMIQUE) : 10 points

DOCUMENTS NON AUTORISES
L'USAGE DES CALCULATRICES EST INTERDIT

PROBLEME 1 (MAPLE)

Soit f une fonction numérique à n variables réelles $(n \in \mathbb{N})$ toute fonction de $\mathbb{R}^n$ dans $\mathbb{R}$ telle que : $f:(x_1,x_2,\ldots,x_n) \mapsto f(x_1,x_2,\ldots,x_n)$. On suppose que toutes les dérivées partielles premières et secondes de f existent et continues.

L'objectif est de déterminer la liste des points critiques de f ainsi que leur nature (minimum local, maximum local, point col ou point dégénéré) s'il y en a.

Un point $U=(u_1,u_2,\ldots,u_n)$ est dit critique si le vecteur gradient de f noté ∇f (avec $\nabla f = \left[\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \ldots, \frac{\partial f}{\partial x_n}\right]$) s'annule en ce point. Ce qui signifie :

$$\nabla f(U)=0 \Leftrightarrow \left[\frac{\partial f}{\partial x_1}(U)=0, \frac{\partial f}{\partial x_2}(U)=0, \ldots, \frac{\partial f}{\partial x_n}(U)=0\right]$$

Pour déterminer la nature d'un point critique, il faut analyser la matrice Hessienne H de f en ce point. On définit, dans le cas général une matrice Hessienne carrée H d'ordre n par :

$$H_{i,j} = \frac{\partial^2 f}{\partial x_i \partial x_j}(x_1,x_2,\ldots,x_n) \quad \text{avec } 1 \le i \le n \text{ et } 1 \le j \le n$$

D'où

$$H = \begin{pmatrix} \frac{\partial^2 f}{\partial x_1 \partial x_1}(x_1,x_2,\ldots,x_n) & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n}(x_1,x_2,\ldots,x_n) \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1}(x_1,x_2,\ldots,x_n) & \cdots & \frac{\partial^2 f}{\partial x_n \partial x_n}(x_1,x_2,\ldots,x_n) \end{pmatrix}$$

Théorèmes :

1. Toute matrice Hessienne est une matrice symétrique à valeurs réelles.
2. Toute matrice Hessienne admet des valeurs propres réelles.
3. Toute matrice Hessienne est définie positive si toutes ses valeurs propres sont strictement positives.
4. Toute matrice Hessienne est définie négative si toutes ses valeurs propres sont strictement négatives.

On détermine la nature d'un point critique comme suit :

- Un point critique de f est dit dégénéré lorsque le déterminant de H est nul en ce point (au moins une des valeurs propres de H est nulle).
- Un point critique de f est non dégénéré, si le déterminant de H est non nul en ce point. Le signe des valeurs propres (toutes non nulles) détermine la nature de ce point (extremum local ou point col) :
  - ✓ si la matrice Hessienne est définie positive, la fonction atteint un minimum local au point critique.
  - ✓ si la matrice Hessienne est définie négative, la fonction atteint un maximum local au point critique.
  - ✓ si les valeurs propres sont de signes différents le point critique est alors un point col.

Travail demandé

Compléter la procédure MAPLE **EXTREMUM** en suivant la démarche donnée ci-dessous :

```
> EXTREMUM:=proc(E :: algebraic , X :: list)
   local n, m, f, V, P, g, H, VP, LVP, LS, SPC ;
   # au besoin utiliser d'autres variables locales
   ...
   ...
   RETURN(SPC) ;
   end proc ;
```

où **E** est une expression dépendante des variables $[x_1, x_2, \cdots, x_n]$ et **X** une liste des variables $[x_1, x_2, \cdots, x_n]$. **EXTREMUM** retourne la séquence **SPC** formée par le nombre de points critiques et la liste de ces points s'il y en a. Chaque élément de cette liste est une liste formée par le point critique et sa nature (*"minimum local"*, *"maximum local"*, *"point col"* ou *"point dégénéré"*).

Exemple:

L'appel à `EXTREMUM(x^4+y^4-(x-y)^2,[x,y])` donne comme résultat :

$$3, [[\{x=-1, y=1\}, \textit{minimum local}], [\{x=0, y=0\}, \textit{point dégénéré}], [\{x=1, y=-1\}, \textit{minimum local}]]$$

et les valeurs des variables locales `n, V, P, m, H, VP, LVP, LS, SPC`, suite à l'exécution de cet appel, sont :

$$n := 2$$

$$V := 4x^3 - 2x + 2y, 4y^3 + 2x - 2y$$

$$P := \{\{x = -1, y = 1\}, \{x = 0, y = 0\}, \{x = 1, y = -1\}\}$$

$$m := 3$$

$$H := \begin{bmatrix} 12x^2 - 2 & 2 \\ 2 & 12y^2 - 2 \end{bmatrix}$$

$$VP := [6x^2 - 2 + 6y^2 + 2\sqrt{9x^4 - 18x^2y^2 + 1 + 9y^4},$$
$$6x^2 - 2 + 6y^2 - 2\sqrt{9x^4 - 18x^2y^2 + 1 + 9y^4}]$$

$$LVP := [[12, 8], [0, -4], [12, 8]]$$

$$LS := [[1, 1], [0, -1], [1, 1]]$$

$$SPC := 3, [[\{x = -1, y = 1\}, \textit{minimum local}], [\{x = 0, y = 0\}, \textit{point dégénéré}],$$
$$[\{x = 1, y = -1\}, \textit{minimum local}]]$$

Démarche à suivre dans la procédure :

1. Affecter à **n** le nombre d'éléments de la liste des variables $[x_1, x_2, \cdots, x_n]$.
2. Transformer **E** en une fonction **f** dépendante des variables $x_1, x_2, \cdots, x_n$.
3. Calculer dans la séquence **V**, le vecteur gradient de **f**.
4. Calculer dans **P**, l'ensemble des points critiques. On suppose que tous les points critiques sont de type `numeric` en MAPLE.
5. Affecter à **m** le nombre de points critiques.

Dans le cas où **m** > 0 :

6. Définir une fonction **g** tel que $g(i, j) = \dfrac{\partial^2 f}{\partial x_i \partial x_j}(x_1, x_2, \ldots, x_n)$.
7. Déterminer la matrice Hessienne **H** en utilisant **g**.
8. Calculer dans **VP**, la liste des valeurs propres de **H**.
9. Déterminer dans **LVP**, la liste des valeurs propres des points critiques.
10. Définir la fonction **signe** qui, pour une valeur x, retourne 1 si x est positive, -1 si x est négative et 0 si x est nulle.
11. En utilisant la fonction **signe**, déterminer dans **LS**, la liste des listes des signes correspondant à **LVP**.
12. Déterminer dans **SPC**, la séquence formée par le nombre de points critiques et la liste des points critiques. Chaque élément de cette liste est une liste formée par le point critique et sa nature (minimum local, maximum local, col ou dégénéré).

PROBLEME 2 (ALGORITHMIQUE)

La quantité d'informations disponible en ligne s'est considérablement accrue. Notons par exemple la numérisation de nombreux ouvrages, le développement exponentiel du web, etc. Un problème clé lié à ce volume de données est l'efficacité de la recherche d'informations.
Dans un moteur de recherche internet par exemple, un facteur d'importance d'une page pour une requête donnée est l'appartenance des mots recherchés à la page ainsi que leur nombre d'apparitions (**occurrences**). Ces informations peuvent être obtenues en mettant en œuvre des algorithmes de recherche comme nous le verrons dans la suite.

Un texte donné est une suite de **n** lettres minuscules stockées dans un tableau d'entiers à une dimension où chaque entier (compris entre 1 et 26) représente une lettre minuscule suivant l'ordre lexicographique (voir tableau suivant).

a	b	c	d	e	f	g	h	i	j	K	l	m	n	o	p	q	r	s	t	u	v	w	x	Y	z
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26

Il est à noter que les espaces, les fins de lignes et les caractères de ponctuation présents dans le texte ne sont pas représentés dans le tableau.
Un mot, composé d'une ou de plusieurs lettres, est un texte.

Exemple :

Le texte "`quel bon bonbon`" est représenté par le tableau **tab** où la case d'indice 0 stocke le nombre de lettres dans le texte.

Texte : q u e l b o n b o n b o n

tab :	**13**	17	21	5	12	2	15	14	2	15	14	2	15	14		
indice :	0	1	2	3	4	5	6	7	8	9	10	11	12	**13**	...	*NMAX*

Travail demandé :
Dans la suite on suppose avoir déjà effectué les déclarations suivantes :

CONSTANTE **NMAX** = 10000

TYPE **TEXTE** = **tableau**$[0..NMAX]$**de entier**

FREQUENCE = **tableau**$[1..26, 0..NMAX]$**de entier**

FREQMAX = **tableau**$[0..26]$**de entier**

On désigne également dans les entêtes par **E** paramètre passé par valeur et par **S** paramètre passé par variable.

1. Ecrire une fonction nommée **Taille** qui saisit et retourne un entier compris entre 1 et **NMAX**. Cette fonction a pour entête :

   FONCTION **Taille**( ) : entier

2. Ecrire une procédure nommée **SaisieTxt** qui saisit dans un tableau **tab** les **n** entiers représentants les lettres d'un texte donné. Cette procédure a pour entête :

   PROCEDURE **SaisieTxt**(**E** **n** : entier , **S** **tab** : TEXTE)

3. Ecrire une fonction nommée **VerifMot** qui retourne *VRAI* si un mot identifié par **mot** apparaît à une position **k** d'un texte **tab** et *FAUX* sinon. Cette fonction a pour entête :

   FONCTION **VerifMot**(**E** **tab** , **mot** : TEXTE , **E** **k** : entier) : booléen

4. En utilisant impérativement **VerifMot**, écrire une fonction nommée **RechercheMot** qui retourne l'indice de la première apparition (occurrence) d'un mot identifié par **mot** si ce mot apparaît dans un texte **tab**, et -1 sinon. Cette fonction a pour entête :

FONCTION **RechercheMot(E tab , mot** : TEXTE) : entier

Exemple :

La première apparition du mot "bonbon" dans le texte "quel bon bonbon" est à l'indice 5.

Texte :		q	u	e	l	b	o	n	b	o	n	b	o	n
tab :	**13**	17	21	5	12	2	15	14	2	15	14	2	15	14
indice :	0	1	2	3	4	5	6	7	8	9	10	11	12	**13**

mot :				b	o	n	b	o	n
1ère occurrence du mot:	**6**	2	15	14	2	15	14
indice :	0	1	2	3	4	5	**6**

5. Ecrire une procédure nommée **TabOccur** qui détermine dans un tableau **occ** les occurrences d'un mot identifié par **mot** apparaissant dans un texte **tab**. La case d'indice 0 de **occ** stocke le nombre d'occurrences. La suite des cases stocke les indices des occurrences si le mot apparaît dans **tab**. Cette procédure a pour entête :

PROCEDURE **TabOccur(E tab , mot** : TEXTE , **S occ** : TEXTE)

Exemple :

Dans le texte "quel bon bonbon" le mot "bonbon" a pour nombre d'occurrences 2 avec 5 indice de la première occurrence et 8 comme indice de la deuxième occurrence.

Texte :		q	u	e	l	b	o	n	b	o	n	b	o	n
tab :	**13**	17	21	5	12	2	15	14	2	15	14	2	15	14
indice :	0	1	2	3	4	5	6	7	8	9	10	11	12	**13**

mot :				b	o	n	b	o	n
1ère occurrence du mot:	**6**	2	15	14	2	15	14
indice :	0	1	2	3	4	5	**6**

mot :						b	o	n	b	o	n
2éme occurrence du mot:						**6**	2	15	14	2	15	14
indice :						0	1	2	3	4	5	**6**

occ :	**2**	5	8
indice :	0	1	**2**

6. Ecrire une fonction nommée **MotRecouv** qui à partir, d'un mot identifié par **mot** et du tableau d'occurrence **occ** correspondant, retourne *VRAI* si certaines occurrences du mot identifié par **mot** se recouvrent dans le texte (comme le montre l'exemple précédant), et *FAUX* sinon. Cette fonction a pour entête :

FONCTION **MotRecouv**(**E** **mot** , **occ** : TEXTE) : booléen

7. En utilisant impérativement la procédure **TabOccur**, écrire une procédure nommée **FrequenceLettres** qui détermine dans un tableau **Freq** de type **FREQUENCE** la fréquence de chaque lettre de l'alphabet (nombre d'occurrences) ainsi que ses positions dans le texte si la lettre existe. Dans ce cas, chaque ligne *i* de **Freq** contient dans l'ordre la fréquence de la lettre ayant l'ordre lexicographique *i* à la colonne 0 et les indices de ses occurrences dans la suite des colonnes. Dans le cas contraire la fréquence de la lettre est nulle. Cette procédure a pour entête :

PROCEDURE **FrequenceLettres** (**E** **tab** : TEXTE , **S** **Freq** : FREQUENCE)

8. Ecrire une procédure nommée **LettresPlusFreq** qui détermine dans un tableau **tabF** de type **FREQMAX** les lettres ayant la fréquence maximale dans le texte représenté par **tab**. La case d'indice 0 de **tabF** stocke la fréquence maximale et la suite des cases contient l'ordre lexicographique des lettres ayant toutes cette même fréquence maximale.