e3a

CONCOURS ARTS ET MÉTIERS ParisTech - ESTP - ARCHIMEDE

Épreuve d'Informatique MP

Durée 3 h

Si, au cours de l'épreuve, un candidat repère ce qui lui semble être une erreur d'énoncé, d'une part il le signale au chef de salle, d'autre part il le signale sur sa copie et poursuit sa composition en indiquant les raisons des initiatives qu'il est amené à prendre.

L'usage de calculatrices est interdit.

AVERTISSEMENT

Indiquer en tête de copie ou de chaque exercice le langage utilisé

Quel que soit le langage utilisé, le candidat pourra supposer qu'il dispose d'une fonction test_liste_vide qui prend en entrée une liste et renvoie le booléen 1 si la liste est vide et 0 sinon.

Le candidat pourra, s'il le juge utile, supposer que chaque liste se termine par un élément de marquage de fin de liste appelé NIL.

La **présentation**, la lisibilité, l'orthographe, la qualité de la **rédaction, la clarté et la précision** des raisonnements entreront pour une **part importante** dans **l'appréciation des copies**. En particulier, les résultats non justifiés ne seront pas pris en compte. Les candidats sont invités à encadrer les résultats de leurs calculs.

Tournez la page S.V.P.

Exercice 1

Soit L un tableau non vide rempli avec des entiers naturels. Une *pente positive* (respectivement une *pente négative*) de longueur l est une suite de l indices consécutifs $i, i+1, \ldots, i+l-1$ sur lesquels L prend des valeurs croissantes $L[i] \leq L[i+1] \leq \cdots \leq L[i+l-1]$, (respectivement des valeurs décroissantes: $L[i] \geq L[i+1] \geq \cdots \geq L[i+l-1]$).

1. Ecrire la fonction :

`MAXPENTEPOS`	`données`	L : `tableau d'entiers naturels`
	`résultat`	l : `entier naturel`

qui prend en entrée un tableau d'entiers naturels et renvoie la plus grande longueur de ses pentes positives.

2. Ecrire la fonction :

`MAXPENTE`	`données`	L : `tableau d'entiers naturels`
	`résultat`	l : `entier naturel`

qui prend en entrée un tableau d'entiers naturels et renvoie la plus grande longueur de ses pentes (qu'elles soient positives ou négatives).

Exercice 2

Soit T un tableau de nombres entiers relatifs dont les cases sont numérotées de 1 à $N \geqslant 1$. Si k et l sont deux entiers naturels tels que $1 \leqslant k \leqslant l \leqslant N$, on note $T[k \cdots l]$ le sous-tableau formé par les cases de T de k à l ; on appelle sous-somme de T de la case l à la case m, et on note $\sigma_{k \to l}(T)$, la somme des valeurs consécutives du sous-tableau $T[k \cdots l]$, c'est-à-dire $\sigma_{k \to l}(T) = T[k] + T[k+1] + \cdots + T[l]$. On cherche la valeur maximale de toutes les sous-sommes de T. On se propose d'étudier différents algorithmes de résolution de ce problème.

1. Ecrire la fonction :

`CUMUL`	`données`	N : `entier`
		T : `tableau d'entiers de longueur` N
	`résultat`	U : `tableau d'entiers de longueur` N

qui prend en entrée un tableau T d'entiers de longueur N et renvoie un tableau d'entiers de longueur N dont la k-ième case contient la sous-somme $\sigma_{1 \to k}(T)$, pour k compris entre 1 et N. On précisera la complexité de cet algorithme.

2. Ecrire la fonction :

`MAXSOMME`	`données`	N : `entier`
		T : `tableau d'entiers de longueur` N
	`résultat`	k : `entier`

qui prend en entrée un tableau T de longueur N et renvoie une sous-somme maximale parmi les sous-sommes de ce tableau. On précisera la complexité de cet algorithme et on cherchera à minimiser cette complexité.

3. Une seconde méthode est basée sur le principe "diviser pour régner" : on divise le tableau en deux sous-tableaux de tailles presque égales. Proposer un algorithme récursif qui prend en entrée un tableau T de nombres entiers relatifs et calcule le quadruplet (S, MS_G, MS, MS_D) tel que : Si T est un tableau à N cases numérotées de 1 à $N \geqslant 1$,

   - S est la somme $\sigma_{1\to N}(T)$,
   - MS_G est le maximum des sous-sommes $\sigma_{1\to l}(T)$, pour l compris entre 1 et N,
   - MS est le maximum des sous-sommes $\sigma_{k\to l}(T)$, pour k et l entiers tels que $1 \leqslant k \leqslant l \leqslant N$,
   - MS_D est le maximum des sous-sommes $\sigma_{k\to N}(T)$, pour k compris entre 1 et N.

   Quelle est la complexité de l'algorithme proposé?

Exercice 3

Les deux parties de cet exercice sont indépendantes.

1. On dispose d'une fonction `ECHANGER` qui prend en entrée un tableau T d'entiers naturels et deux indices I, J, compris entre 1 et la longueur du tableau T, et échange les contenus des cases I et J dans le tableau.

   On considère l'algorithme écrit en pseudo-langage:

```
PROC1  données          T : tableau[1..N] de 0,1
       variables        I, J : entiers
       I ← 1
       J ← N
       tant que I < J   faire
                        si T[I] = 0 alors I ← I + 1
                        sinon ECHANGER(T, I, J) ; J ← J − 1
                        fin si
                        fin faire
```

   (a) Que fait ce programme sur le tableau $[1, 0, 1, 0, 1, 0]$? Détailler son exécution.

   (b) Démontrer que ce programme termine.

   (c) Quel est le but de ce programme ? Le démontrer précisément.

   (d) Proposer une implémentation de l'algorithme ci-dessus en remplaçant la boucle par une procédure récursive.

   (e) Proposer un programme pour la fonction `ECHANGER` qui n'utilise pas une variable supplémentaire.

Tournez la page S.V.P.

2. On dispose de deux fonctions :
   - La fonction **par** qui prend en entrée un entier naturel et renvoie la valeur 0 si cet entier est pair et 1 si cet entier est impair,
   - la fonction **ent** qui prend en entrée un nombre rationnel et renvoie sa partie entière.

   On considère l'algorithme écrit en pseudo-langage:

```
PROC2  données        I, J : entiers
       variables      t : entier
                      t ← 0
       tant que I ≥ 1  faire
                      si par(I) = 1 alors t := t + J
                      fin si
                      J ← 2 * J
                      I ← ent(I/2)
                      fin faire
       Retourner t
```

   (a) Ecrire l'exécution de ce programme sur les entiers $I = 8$ et $J = 9$.
   (b) Ecrire l'exécution de ce programme sur les entiers $I = 9$ et $J = 8$.
   (c) Que calcule ce programme ? Le justifier précisément.

Exercice 4

Soit Σ un alphabet fini. On note Σ^* l'ensemble des mots finis écrits avec des lettres de Σ.

On rappelle la définition : Soient u et v deux mots dans Σ^*. Soient p, q dans $\mathbb{N}$ et $a_1, ..., a_p, b_1, ..., b_q$ dans Σ tels que $u = a_1...a_p$ et $v = b_1....b_q$. On dit que u est un *facteur* de v si $q \geq p$ et s'il existe un entier naturel r compris entre 0 et $q - p$ tel que, pour tout i compris entre 1 et p, $a_i = b_{i+r}$.

Soit $\mathcal{F}_\Sigma$ l'ensemble des automates finis $\mathcal{A} = (Q, \Sigma, F, \epsilon, T)$ tels que :

- L'ensemble des états Q est un ensemble fini, réunion d'un singleton $\{\epsilon\}$ et d'une partie de l'ensemble Σ.
- L'état ϵ est l'unique état initial.
- L'ensemble F est une partie de Q désignant l'ensemble des états finals.
- T désigne l'ensemble des transitions. Il vérifie la propriété : Pour toute lettre a dans l'alphabet Σ, les transitions étiquetées par la lettre a ne peuvent aboutir que dans l'état a.

En raison de cette dernière propriété, dans un automate de l'ensemble $\mathcal{F}_\Sigma$, l'étiquette d'une transition est déterminée par l'état d'arrivée de la transition. Pour cette raison, si $\mathcal{A} = (Q, \Sigma, F, \epsilon, T)$ est un automate de l'ensemble $\mathcal{F}_\Sigma$, si a_1, a_2 sont deux lettres de Σ telle qu'il existe une transition de l'état a_1 vers l'état a_2 dans T, on note (a_1, a_2) cette transition et son étiquette est alors a_2.

Un exemple d'un tel automate sur l'alphabet à trois lettres $\{a, b, c\}$ est donné par la figure 1. Pour cet automate $\mathcal{A}_{ex} = (Q_{ex}, \{a, b, c\}, F_{ex}, \epsilon, T_{ex})$, $Q_{ex} = \{\epsilon, a, b, c\}$, $F_{ex} = \{a, c\}$ et $T_{ex} = \{(\epsilon, a), (a, a), (b, a), (\epsilon, b), (b, b), (c, b), (a, c), (c, c)\}$.

On note $\mathcal{L}(\mathcal{F}_\Sigma)$ l'ensemble des langages reconnus par les automates de l'ensemble $\mathcal{F}_\Sigma$.

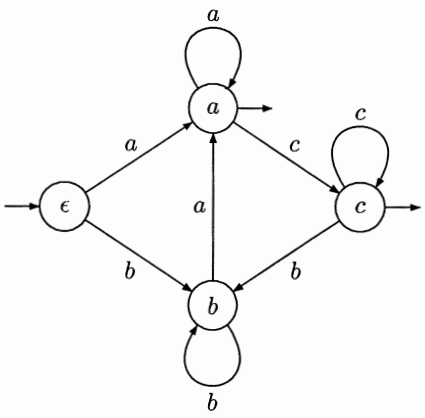

Figure 1: L'automate $\mathcal{A}_{ex}$ de l'ensemble $\mathcal{F}_{\{a,b,c\}}$.

1. Dans le cas particulier où Σ est l'alphabet à une lettre $\{a\}$, dresser la liste des (six) langages de l'ensemble $\mathcal{L}(\mathcal{F}_{\{a\}})$. Pour chacun d'eux, on dessinera un automate de l'ensemble $\mathcal{F}_{\{a\}}$ le reconnaissant.

2. On considère le cas particulier où Σ est l'alphabet à trois lettres $\{a, b, c\}$.

   (a) Démontrer que le langage réduit au singleton $\{abc\}$ appartient à $\mathcal{L}(\mathcal{F}_{\{a,b,c\}})$. On dessinera explicitement un automate dans $\mathcal{F}_{\{a,b,c\}}$ qui reconnait le langage (abc).

   (b) Démontrer que le langage $(abc)^*$ appartient à $\mathcal{L}(\mathcal{F}_{\{a,b,c\}})$. On dessinera explicitement un automate dans $\mathcal{F}_{\{a,b,c\}}$ qui reconnait le langage $(abc)^*$.

3. On revient au cas général. Soit p un entier naturel ≥ 2. Soit $u = a_1 a_2 \cdots a_p$ un mot de longueur p dans Σ. On note L_u le langage réduit au singleton $\{u\}$.

   (a) On suppose les p lettres $a_1, \cdots, a_p$ distinctes deux à deux.

      i. Démontrer que L_u est dans $\mathcal{L}(\mathcal{F}_\Sigma)$.

      ii. Démontrer que $(L_u)^*$ est dans $\mathcal{L}(\mathcal{F}_\Sigma)$.

   (b) Réciproquement, on suppose que le langage L_u est dans $\mathcal{L}(\mathcal{F}_\Sigma)$. Démontrer que les p lettres $a_1, \cdots, a_p$ sont distinctes deux à deux.

4. Démontrer qu'un automate de $\mathcal{F}_\Sigma$ est un automate déterministe.

5. Soit L un langage dans $\mathcal{L}(\mathcal{F}_\Sigma)$ reconnu par un automate $\mathcal{A} = (Q, \Sigma, F, \epsilon, T)$ dans $\mathcal{F}_\Sigma$. Démontrer que le langage L^* est dans $\mathcal{L}(\mathcal{F}_\Sigma)$. On pourra utiliser l'automate $\mathcal{A}$ pour construire un automate dans $\mathcal{F}_\Sigma$ qui reconnaît L^*.

6. Soient P et D deux parties de Σ et soit ϕ une partie formée de mots de longueur 2. On note $L(P, D, \phi)$ le langage formé par les mots dans Σ^* dont la première lettre est dans P, la dernière lettre est dans D et ne contenant aucun facteur de longueur 2 dans ϕ.

   (a) Dans le cas particulier où Σ est l'alphabet à deux lettres $\{a, b\}$, $P_0 = \{a\}$, $D_0 = \{b\}$ et $\phi_0 = \{a^2, b^2\}$, décrire précisément le langage $L(P_0, D_0, \phi_0)$.

   (b) Dans le cas particulier où Σ est l'alphabet à trois lettres $\{a, b, c\}$, déterminer P_1, D_1 et ϕ_1 tels que $(abc)^+ = L(P_1, D_1, \phi_1)$.

   (c) Toujours dans le cas particulier où Σ est l'alphabet à trois lettres $\{a, b, c\}$, déterminer P_{ex}, D_{ex} et ϕ_{ex} tels que $L_{ex} = L(P_{ex}, D_{ex}, \phi_{ex})$, L_{ex} désignant le langage reconnu par l'automate $\mathcal{A}_{ex}$ de la figure 1.

   (d) On revient au cas général. Démontrer que le langage $L(P, D, \phi)$ est dans $\mathcal{L}(\mathcal{F}_\Sigma)$.

Tournez la page S.V.P.

(e) Soit L un langage dans Σ^*. On note $P(L)$ l'ensemble des lettres a dans Σ telle qu'il existe un mot de L dont la première lettre est a. On note $D(L)$ l'ensemble des lettres a dans Σ telle qu'il existe un mot de L dont la dernière lettre est a. On note $\phi(L)$ l'ensemble complémentaire des facteurs de longueur 2 de mots de L. On suppose que L est un langage dans $\mathcal{L}(\mathcal{F}_\Sigma)$ ne contenant pas le mot vide. Démontrer que $L = L(P(L), D(L), \phi(L))$.

7. On se donne des lettres $a_1, \cdots, a_p, \cdots$, distinctes deux à deux. On considère les langages définis récursivement par : $L_0 = \emptyset$ et, $L_i = (L_{i-1})^* a_i$, pour i un entier ≥ 1. Soit p un entier ≥ 1.

   (a) Démontrer que L_p est un langage de $\mathcal{L}(\mathcal{F}_{\{a_1,\ldots,a_p\}})$.

   (b) Proposer un algorithme qui prend en entrée l'entier naturel p et retourne un automate de $\mathcal{F}_{\{a_1,\ldots,a_p\}}$ qui reconnait L_p.