ÉCOLE POLYTECHNIQUE – ÉCOLE NORMALE SUPÉRIEURE DE CACHAN

ÉCOLE SUPÉRIEURE DE PHYSIQUE ET DE CHIMIE INDUSTRIELLES

CONCOURS D'ADMISSION 2013 — FILIÈRE MP HORS SPÉCIALITÉ INFO

FILIÈRE PC

COMPOSITION D'INFORMATIQUE – B – (XEC)

(Durée : 2 heures)

L'utilisation des calculatrices **n'est pas autorisée** pour cette épreuve.
Le langage de programmation choisi par le candidat doit être spécifié en tête de la copie.

⋆ ⋆ ⋆

Points fixes de fonctions à domaine fini

Dans ce problème, on s'intéresse aux points fixes des fonctions $f\colon E \to E$, où E est un ensemble fini. Le calcul effectif et efficace des points fixes de telles fonctions est un problème récurrent en informatique (transformation d'automates, vérification automatique de programmes, algorithmique des graphes, etc), et admet différentes approches selon la structure de E et les propriétés de f.

On suppose par la suite un entier strictement positif $n > 0$ fixé et rangé dans une constante globale de même nom, et on pose $\mathsf{E}_n = \{0, \ldots, n-1\}$. On représente une fonction $f\colon \mathsf{E}_n \to \mathsf{E}_n$ par un tableau `t` de taille n, autrement dit $f(x) =$`t[x]` pour tout $x = 0, \ldots, n-1$. Ainsi la fonction f_0 qui à $x \in \mathsf{E}_{10}$ associe $2x + 1$ modulo 10 est-elle représentée par le tableau

1	3	5	7	9	1	3	5	7	9
t[0]	t[1]	t[2]	t[3]	t[4]	t[5]	t[6]	t[7]	t[8]	t[9]

.

Les tableaux sont indexés à partir de 0 et la notation `t`$[i]$ est utilisée dans les questions pour désigner l'élément d'indice i du tableau `t`, indépendamment du langage de programmation choisi. Quel que soit le langage utilisé, on suppose qu'il existe une primitive `allouer(n)` pour créer un tableau d'entiers de taille n (le contenu des cases du nouveau tableau est à priori quelconque). On suppose les entiers machines signés, et on suppose que les entiers $-n, -n+1, \ldots, n-1, n$ ne débordent pas de la capacité des entiers machines – en d'autres termes, les entiers machines représentent fidèlement ces entiers. On suppose que les tableaux peuvent être passés en argument – le type de passage de paramètre, par valeur ou par adresse, devra être précisé par le candidat si le comportement du code écrit venait à en dépendre. On note dans l'énoncé `vrai` et `faux` les deux valeurs possibles d'un booléen. Le candidat reste libre d'utiliser d'autres notations ou d'autres primitives, pourvu qu'elles existent dans le langage de son choix et qu'elles soient clairement

spécifiées. Enfin, le code écrit devra être sûr (pas d'accès invalide à un tableau, pas de division par zéro, et le programme termine, notamment) pour toutes valeurs des paramètres vérifiant les conditions données dans l'énoncé.

Le temps de calcul d'une procédure `proc` de paramètres $p_1, \ldots, p_k$ est défini comme le nombre d'opérations (accès en lecture ou écriture à une case d'un tableau ou à une variable, appel à une des primitives données dans l'énoncé) exécutées par `proc` pour ces paramètres ; on note $T(\texttt{proc}, n)$ le temps de calcul maximal pris sur tous les paramètres possibles pour n fixé. On dit que `proc` s'exécute en temps linéaire si il existe des réels $\alpha, \beta > 0$ et un entier $N \geq 0$ tels que $\alpha.n \leq T(\texttt{proc}, n) \leq \beta.n$ pour tout $n \geq N$. De même, on dit que `proc` s'exécute en temps logarithmique si il existe des réels $\alpha, \beta > 0$ et un entier $N \geq 0$ tels que $\alpha \log n \leq T(\texttt{proc}, n) \leq \beta \log n$ pour tout $n \geq N$.

Partie I. Recherche de point fixe : cas général

On rappelle que x est un *point fixe* de la fonction f si et seulement si $f(x) = x$.

Question 1 Écrire une procédure `admet_point_fixe(t)` qui prend en argument un tableau `t` de taille n et renvoie `vrai` si la fonction $f : \mathsf{E}_n \to \mathsf{E}_n$ représentée par `t` admet un point fixe, `faux` sinon. Par exemple, `admet_point_fixe` devra renvoyer `vrai` pour le tableau donné en introduction, puisque 9 est un point fixe de la fonction f_0 qui à x associe $2x + 1$ modulo 10.

Question 2 Écrire une procédure `nb_points_fixes(t)` qui prend en argument un tableau `t` de taille n et renvoie le nombre de points fixes de la fonction $f : \mathsf{E}_n \to \mathsf{E}_n$ représentée par `t`. Par exemple, `nb_points_fixes` devra renvoyer 1 pour le tableau donné en introduction, puisque 9 est le seul point fixe de f_0.

On note f^k l'itérée k-ième de f, autrement dit

$$\begin{array}{rccl} f^k : & \mathsf{E}_n & \to & \mathsf{E}_n \\ & x & \mapsto & \underbrace{f(f(\ldots f}_{k \text{ fois}}(x))\ldots) \ . \end{array}$$

Question 3 Écrire une procédure `itere(t,x,k)` qui prend en premier argument un tableau `t` de taille n représentant une fonction $f : \mathsf{E}_n \to \mathsf{E}_n$, en deuxième et troisième arguments des entiers x, k de E_n, et renvoie $f^k(x)$.

Question 4 Écrire une procédure `nb_points_fixes_iteres(t,k)` qui prend en premier argument un tableau `t` de taille n représentant une fonction $f : \mathsf{E}_n \to \mathsf{E}_n$, en deuxième argument un entier $k \geq 0$, et renvoie le nombre de points fixes de f^k.

Un élément $z \in \mathsf{E}_n$ est dit *attracteur principal* de $f : \mathsf{E}_n \to \mathsf{E}_n$ si et seulement si z est un point fixe de f, et pour tout $x \in \mathsf{E}_n$, il existe un entier $k \geq 0$ tel que $f^k(x) = z$.

Afin d'illustrer cette notion, on pourra vérifier que la fonction f_1 représentée par le tableau

ci-dessous admet 2 comme attracteur principal.

5	5	2	2	0	2	2
t[0]	t[1]	t[2]	t[3]	t[4]	t[5]	t[6]

En revanche, on notera que la fonction f_0 donnée en introduction n'admet pas d'attracteur principal, puisque $f_0^k(0) \neq 9$ quel que soit l'entier $k \geq 0$.

Question 5 Écrire une procédure `admet_attracteur_principal(t)` qui prend en argument un tableau `t` de taille n et renvoie `vrai` si et seulement si la fonction $f\colon \mathsf{E}_n \to \mathsf{E}_n$ représentée par `t` admet un attracteur principal, `faux` sinon. On ne requiert pas ici une solution efficace.

On suppose aux questions 6 et 7 que f admet un attracteur principal. Le *temps de convergence* de f en $x \in \mathsf{E}_n$ est le plus petit entier $k \geq 0$ tel que $f^k(x)$ soit un point fixe de f. Pour la fonction f_1 ci-dessus, le temps de convergence en 4 est égal à 3. En effet, $f_1(4) = 0$, $f_1^2(4) = 5$, $f_1^3(4) = 2$, et 2 est un point fixe de f_1. On note $\mathsf{tc}(f, x)$ le temps de convergence de f en x.

Question 6 Écrire une procédure `temps_de_convergence(t,x)` qui prend en premier argument un tableau `t` de taille n représentant une fonction $f\colon \mathsf{E}_n \to \mathsf{E}_n$ qui admet un attracteur principal, en deuxième argument un entier x de E_n, et renvoie le temps de convergence de f en x. On pourra admettre que $\mathsf{tc}(f, x)$ vaut 0 si x est un point fixe de f, et $1 + \mathsf{tc}(f, f(x))$ si x n'est pas un point fixe de f.

Question 7 Écrire une procédure `temps_de_convergence_max(t)` qui prend en argument un tableau `t` de taille n représentant une fonction $f\colon \mathsf{E}_n \to \mathsf{E}_n$ qui admet un attracteur principal, et renvoie $\max_{i=0\ldots n-1} \mathsf{tc}(f, i)$. On impose un temps de calcul <u>linéaire</u> en la taille n du tableau. À titre d'indication, on pourra au besoin créer un deuxième tableau, qui servira d'intermédiaire au cours du calcul. On ne demande pas de démonstration du fait que le temps de calcul de la solution proposée est linéaire.

Partie II. Recherche efficace de points fixes

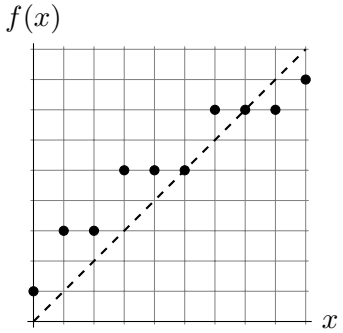
Toute procédure `point_fixe(t)` retournant un point fixe d'une fonction arbitraire est de complexité au mieux linéaire en n. On s'intéresse maintenant à des améliorations possibles de cette complexité lorsque la fonction considérée possède certaines propriétés spécifiques. Nous examinons deux cas.

Premier cas.

Le premier cas que nous considérons est celui d'une fonction croissante de E_n dans E_n. On rappelle qu'une fonction $f\colon \mathsf{E}_n \to \mathsf{E}_n$ est croissante si et seulement si pour tous $x, y \in \mathsf{E}_n$ tels que $x \leq y$, $f(x) \leq f(y)$.

On admet qu'une fonction croissante de E_n dans E_n admet toujours un point fixe.

À titre d'exemple, la fonction dont le tableau et le graphe sont donnés ci-dessous est croissante. Elle a deux points fixes, à savoir les entiers 5 et 7.



1	3	3	5	5	5	7	7	7	8
t[0]	t[1]	t[2]	t[3]	t[4]	t[5]	t[6]	t[7]	t[8]	t[9]

Question 8 Écrire une procédure `est_croissante(t)` qui prend en argument un tableau `t` de taille n et renvoie `vrai` si la fonction représentée par `t` est croissante, `faux` sinon. On impose un temps de calcul linéaire en la taille n du tableau. On ne demande pas de démonstration du fait que le temps de calcul de la solution proposée est linéaire.

Question 9 Écrire une procédure `point_fixe(t)` qui prend en argument un tableau `t` de taille n représentant une fonction croissante $f : \mathsf{E}_n \to \mathsf{E}_n$, et retourne un entier $x \in \mathsf{E}_n$ tel que $f(x) = x$. On impose un temps de calcul logarithmique en la taille n du tableau. On ne demande pas ici de démonstration du fait que le temps de calcul de la solution proposée est logarithmique, ceci étant le sujet de la question suivante.

Question 10 Démontrer que la procédure de la question 9 termine. On rappelle que pour prouver qu'une boucle termine, il suffit d'exhiber un entier positif i, fonction des variables du programme, qui décroît strictement à chaque itération de boucle. Justifier que le temps de calcul est logarithmique en la taille n du tableau.

Deuxième cas.

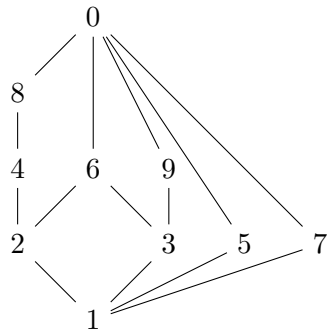
On peut généraliser la notion de fonction croissante comme suit. On rappelle qu'une relation binaire $\preceq$ sur un ensemble E est une relation d'ordre si et seulement si elle est réflexive ($x \preceq x$ pour tout $x \in E$), anti-symétrique (pour tous $x, y \in E$, si $x \preceq y$ et $y \preceq x$, alors $x = y$), et transitive (pour tous $x, y, z \in E$, si $x \preceq y$ et $y \preceq z$, alors $x \preceq z$). Soit $\preceq$ une relation d'ordre sur E. Une fonction $f : E \to E$ est *croissante au sens de* $\preceq$ si et seulement si pour tous $x, y \in E$, $x \preceq y$ implique $f(x) \preceq f(y)$.

Ceci généralise la notion de fonction croissante de E_n dans E_n, que l'on retrouve en prenant $E = \mathsf{E}_n$ et $\preceq$ la relation d'ordre $\leq$. On s'intéresse dorénavant à d'autres relations d'ordre sur E_n.

On dit qu'un élément m de E est un *plus petit élément* de E au sens de $\preceq$ si et seulement si, pour tout $x \in E$, $m \preceq x$. On admet que pour tout ensemble fini E, muni d'une relation d'ordre $\preceq$ et qui admet un plus petit élément m au sens de $\preceq$, pour toute fonction croissante $f : E \to E$ au sens de $\preceq$, il existe un entier $k \geq 0$ tel que $f^k(m)$ est un point fixe de f dans E.

Question 11 Soit E un ensemble fini quelconque muni d'une relation d'ordre $\preceq$ et admettant un plus petit élément m au sens de $\preceq$. Soit $f\colon E \to E$ une fonction croissante au sens de $\preceq$, et soit $k \geq 0$ un entier tel que $f^k(m)$ soit un point fixe de f dans E. Démontrer que $f^k(m)$ est en fait le plus petit point fixe de f au sens de $\preceq$, autrement dit que pour tout autre point fixe x de f dans E, on a $f^k(m) \preceq x$.

Nous nous intéressons maintenant à un choix particulier d'ordre $\preceq$, appelé *ordre de divisibilité* et noté $|$. Précisément, on note $a \mid b$ la relation d'ordre "a divise b" sur les entiers positifs, vraie si et seulement s'il existe un entier $c \geq 0$ tel que $ca = b$. Ainsi, l'ensemble E_{10} ordonné par divisibilité peut se représenter graphiquement comme suit.



D'après la définition donnée précédemment, une fonction $f\colon \mathsf{E}_n \to \mathsf{E}_n$ *croissante au sens de l'ordre de divisibilité* est une fonction telle que pour tous x, y dans E_n, si $x \mid y$, alors $f(x) \mid f(y)$. Par exemple, la fonction représentée par le tableau ci-dessous est croissante au sens de l'ordre de divisibilité.

0	2	4	6	4	8	0	2	0	6
`t[0]`	`t[1]`	`t[2]`	`t[3]`	`t[4]`	`t[5]`	`t[6]`	`t[7]`	`t[8]`	`t[9]`

.

On remarque que, par la question 11, toute fonction de E_n dans E_n croissante au sens de l'ordre de divisibilité a un plus petit point fixe au sens de l'ordre de divisibilité.

On rappelle que le pgcd de deux entiers $x \geq 1$ et $y \geq 1$ est le plus grand entier non nul qui divise x et y. On étend cette définition à des entiers naturels quelconques, en convenant de définir le pgcd d'un entier $x \geq 0$ et de 0 comme valant x.

Question 12 Soit f une fonction de E_n dans E_n, croissante au sens de l'ordre de divisibilité, et notons $x_1, \ldots, x_m$ les points fixes de f dans E_n. Montrer que le plus petit point fixe de f au sens de l'ordre de divisibilité est exactement le pgcd de $x_1, \ldots, x_m$.

Question 13 Écrire une procédure `pgcd_points_fixes(t)` qui prend en argument un tableau `t` de taille n représentant une fonction de E_n dans E_n, croissante au sens de la divisibilité, et renvoie le pgcd de ses points fixes. On impose un temps de calcul logarithmique en la taille n du tableau. On ne demande pas ici de démonstration du fait que le temps de calcul de la solution proposée est logarithmique, ceci étant le sujet de la question qui suit.

Question 14 Justifier que la procédure de la question 13 a un temps de calcul logarithmique en la taille n du tableau.

* *
*