ÉCOLE POLYTECHNIQUE

CONCOURS D'ADMISSION 2009 **FILIÈRES PSI et PT**

ÉPREUVE D'INFORMATIQUE

(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.
On attachera une grande importance à la concision, à la clarté, et à la précision de la rédaction.
Chaque partie du problème utilise des notions introduites dans les parties précédentes.

⋆ ⋆ ⋆

À propos des nombres premiers

Le problème porte sur les entiers naturels, on utilisera les entiers fournis par le langage de programmation choisi en supposant qu'ils sont les entiers naturels. Par ailleurs, étant donnés deux entiers (naturels) a et b ($b > 0$) :

- La notation a / b *dans les programmes* désigne le quotient de la division euclidienne de a par b (parfois appelée « division entière »).
- Une fonction $\mathsf{mod}(a, b)$ (avec $a \geq 0$ et $b > 0$) donne le reste de la division euclidienne de a par b.
- Une fonction $\mathsf{sqrt}(a)$ donne la partie entière de la racine carrée de a (notée $\lfloor \sqrt{a} \rfloor$).

Un entier b *divise* un entier a si a s'écrit comme le produit bq. Dans ce cas, on dit que b est un *diviseur* (ou un facteur) de a. On observera que pour savoir si b divise a, il suffit de vérifier que $\mathsf{mod}(a, b)$ vaut 0.

I. Nombres premiers

On rappelle qu'un entier supérieur ou égal à 2 est premier si et seulement si il n'est divisible que par 1 et par lui-même.

Question 1. Écrire la fonction $\mathsf{estPremier}(n)$ qui prend un entier n ($n \geq 2$) en argument, et qui renvoie 1 si n est premier et 0 sinon. On appliquera directement la définition des nombres premiers donnée ci-dessus.

On veut maintenant calculer tous les nombres premiers inférieurs à un entier donné n. On va procéder selon trois méthodes différentes.

Question 2. Écrire la fonction petitsPremiers(n) qui prend en argument un entier $n \geq 2$, et qui :

1. Range (dans l'ordre croissant) les nombres premiers inférieurs ou égaux à n dans un tableau global (ou prédéclaré) premier, que l'on supposera assez grand.
2. Renvoie le nombre d'entiers rangés dans premier.

Par exemple pour $n = 17$, votre fonction devra renvoyer 7 et remplir le début de premier ainsi :

2	3	5	7	11	13	17	$\cdots$

La fonction petitsPremiers sera la plus simple possible, des techniques plus efficaces sont l'objet des deux questions suivantes.

Il est assez facile de voir que, pour tester si un entier *impair* m ($3 \leq m$) est premier, il suffit de vérifier que m n'est divisible par aucun nombre *premier* compris entre 3 et $\lfloor\sqrt{m}\rfloor$.

Question 3. Écrire une nouvelle fonction petitsPremiers2(n), qui agit comme petitsPremiers(n) mais est plus efficace. Cette nouvelle fonction appliquera la remarque ci-dessus, les nombres premiers entre 3 et $\lfloor\sqrt{m}\rfloor$ étant lus dans le tableau premier en cours de remplissage.

Au III$^{\text{e}}$ siècle avant Jesus-Christ, Ératosthène, mathématicien, astronome et philosophe invente une méthode efficace pour énumérer tous les nombres premiers inférieurs à un entier donné n. Cette méthode est connue sous le nom de crible d'Ératosthène, que nous allons décrire en tenant compte des moyens disponibles à l'époque.

1. Sur le sable, avec un bâton, dessiner un tableaux de $(n-1)$ cases, la première case représente 2, la suivante 3 etc. Ramasser un caillou, le poser à gauche du tableau.
2. Avancer le caillou jusqu'à trouver une case non-rayée. Cette case représente l'entier i.
3. Ramasser un second caillou et le faire avancer par sauts de i cases. Pour chacune des cases sur lesquelles le second caillou se pose :

   (a) Si la case est déjà rayée, ne rien faire. (b) Sinon, rayer la case avec le bâton.
4. Poser le second caillou dès que l'on sort du tableau. Si l'étape (b) n'a pas été effectuée, c'est fini. Sinon, recommencer en 2.

Les cases *non-rayées* représentent maintenant les nombres premiers. À titre d'exemple, voici l'état du tableau au début de l'étape 4 et pour un tableau de 10 cases, le premier caillou étant « • » :

•		X		X		X		X		($i = 2$)
	•	X		X		X	X	X		($i = 3$)
		X	•	X		X	X	X		($i = 5$)

Les nombres premiers trouvés sont donc $2, 3, 5, 7, 11$.

Question 4. Écrire la fonction petitsPremiers3(n), qui agit comme petitsPremiers(n) mais utilise le crible d'Ératosthène. On s'attachera à respecter l'esprit de cette technique, et en particulier à éviter multiplications, divisions et racines carrées, difficiles à effectuer pour les mathématiciens grecs.

On rappelle que tout entier n ($n \geq 2$) peut s'écrire de manière unique comme le produit $n = p_1 p_2 \cdots p_k$, où $p_1 \leq p_2 \leq \ldots \leq p_k$ sont des nombres premiers. On appelle cette écriture la décomposition en facteurs premiers de n. Cette définition suggère un algorithme simple

pour décomposer n en facteurs premiers. Soit $\pi_1, \pi_2, \ldots$ la suite des nombres premiers en ordre croissant.

1. Poser $m = n$, poser $i = 1$.
2. Si $m = 1$, alors terminer.
3. Si π_i divise m, alors π_i entre dans la décomposition de n. Poser $m = m/\pi_i$ et aller en 3.
4. Sinon, poser $i = i + 1$ et aller en 2.

En résumé, l'algorithme, dit *essai des divisions* revient à diviser n par les nombres premiers.

Question 5. Écrire la fonction factoriser(n) qui prend en argument un entier n ($n \geq 2$), et qui range (dans l'ordre croissant au sens large) les facteurs premiers de n dans un tableau global facteur, supposé assez grand. La fonction factoriser renvoie le nombre d'entiers rangés dans facteur. Par exemple pour $n = 90$, votre fonction devra renvoyer 4 et remplir le début de facteur ainsi :

2	3	3	5	$\cdots$

Il n'est pas trop difficile de voir que, dans l'algorithme d'essai des divisions, on peut remplacer la suite des nombres premiers par une suite plus simple à calculer, du moment que celle-ci commence par 2, soit croissante, et contienne les nombres premiers — par exemple, $q_1 = 2$, $q_{j+1} = 2j + 1$ ($j \geq 1$). En effet, un invariant de l'algorithme est que, à l'étape 2, m n'est divisible par aucun des entiers $q_1, \ldots, q_{i-1}$, et donc par aucun nombre premier strictement inférieur à q_i. Par conséquent, si q_i divise m (étape 3), alors q_i est premier.

Par ailleurs, on peut limiter la taille des facteurs essayés. La suite des q_i est strictement croissante, tandis que m décroit. On finira donc par avoir $q_i^2 > m$. Dès lors, m, s'il n'est pas égal à 1, est premier, puisque non-divisible par $q_1, q_2, \ldots, q_{i-1}$, séquence qui contient tous les nombres premiers inférieurs ou égaux à $\lfloor\sqrt{m}\rfloor$. Dans ces conditions m est le dernier facteur de la décomposition de n.

Question 6. Écrire une nouvelle fonction factoriser2(n) qui agit comme factoriser, mais ne calcule pas de table de nombres premiers et exploite la seconde remarque ci-dessus.

II. Reconnaître les puissances

Une écriture courante de la décomposition en facteurs premiers regroupe les facteurs égaux :

$$n = p_1^{\alpha_1} p_2^{\alpha_2} \cdots p_k^{\alpha_k}, \quad p_1 < p_2 < \cdots < p_k \text{ premiers}, \quad 0 < \alpha_1, 0 < \alpha_2, \ldots, 0 < \alpha_k$$

Les α_i sont les *multiplicités* de n — en fait les multiplicités des facteurs premiers de n.

Question 7. Écrire la fonction calculerAlpha(n) ($n \geq 2$) qui range les multiplicités dans un tableau global alpha et renvoie le nombre d'entiers rangés dans alpha.

On admet que l'entier n ($n \geq 2$) s'écrit comme la puissance $n = a^b$, si et seulement si b divise toutes les mutiplicités de n.

Question 8. Écrire la fonction estPuissance(n, b) ($n \geq 2, b \geq 1$) qui renvoie 1 si il existe un entier a tel que $n = a^b$ et 0 sinon.

Question 9. La décomposition en facteurs premiers est une opération coûteuse. Proposer une technique alternative pour détecter si un entier n est un carré ou pas. Aucun code n'est demandé.

III. Nombres de Carmichael

Par définition, un entier c est un *nombre de Carmichael*, si et seulement si :

- L'entier c est impair et n'est pas premier.
- La décomposition de c en facteurs premiers s'écrit $c = p_1 p_2 \dots p_k$ où $p_1 < p_2 < \dots < p_d$.
- Le nombre $p_i - 1$ divise $c - 1$ pour tous les facteurs premiers p_i de c.

On note que l'on a nécessairement $2 \leq d$ et $3 \leq p_1$.

Question 10. Écrire la fonction estCarmichael(c) ($c \geq 2$) qui renvoie 1 si c est un nombre de Carmichael, et 0 sinon. On appliquera directement la définition des nombres de Carmichael donnée ci-dessus.

Énumérer les nombres de Carmichael par la méthode évidente est trop coûteux. On s'intéresse d'abord à une classe particulière de nombres de Carmichael : ceux de la forme $c = p_1 p_2 p_3$, dits *d'ordre* 3.

Question 11. Écrire la fonction calculerCarmichael3(n) ($n \geq 2$) qui range les nombres de Carmichael d'ordre 3 inférieurs ou égaux à n dans un tableau global carmichael. La fonction calculerCarmichael3(n) renvoie le nombre d'entiers rangés dans le tableau. On utilisera la méthode suivante :

- Calculer les nombres premiers entre 3 et n.
- Tester la dernière condition de la définition des nombres de Carmichael pour tous les produits possibles de trois de ces nombres.

Pour trouver tous les nombres de Carmichael inférieurs à n, il existe une technique de criblage assez efficace. On commence par se donner un tableau d'entiers t de $n + 1$ cases, de sorte que la case d'indice i représente l'entier i. Initialement, la case d'indice i contient i. Puis ...

- Pour chaque nombre premier p, compris entre 3 (inclus) et $\lfloor \sqrt{n} \rfloor$ (exclu).
  - Pour chaque indice i valide, de la forme $i = p^2 + k \cdot p(p-1)$ (k entier naturel).
    - Poser t[i] = t[i]/p.

Maintenant, les indices plus grands que 2 qui désignent une case dont le contenu vaut 1 sont exactement les nombres de Carmichael recherchés.

Question 12. Écrire la fonction calculerCarmichael(n) ($n \geq 2$) qui range les nombres de Carmichael inférieurs ou égaux à n dans le tableau global carmichael. La fonction calculerCarmichael(n) renvoie le nombre d'entiers rangé dans le tableau. On utilisera la technique de criblage décrite ci-dessus.

* *
*