Séance 6b: Exercices sur les tableaux d'entiers

Université Paris-Diderot

Objectifs:

- Manipuler les tableaux d'entiers
- Concevoir et programmer des algorithmes sur les tableaux

Dans cette séance, vous résoudrez des exercices et des problèmes sur des tableaux d'entiers. Vous écrirez des boucles et définirez des fonctions intermédiaires pour rendre votre code plus lisible et concis.

Exercice 1 (Maximum, ⋆)
Écrire une fonction `maximum` *qui prend en argument un tableau d'entiers* `tab` *et qui renvoie l'indice* `i` *du maximum de ce tableau.*

Contrat:
```
int[] tab = {1000,7,1,1,1}
maximum(tab) doit renvoyer 0.
```

□

Exercice 2 (Signes, ⋆)
Écrire une fonction `samesign` *qui prend en argument deux tableaux d'entiers* `t1` *et* `t2` *et qui renvoie un tableau dont la valeur en la coordonnée* `i` *vaut* `1` *si les deux tableaux* `t1` *et* `t2` *ont des valeurs de même signe en la coordonnée* `i`, `-1` *sinon. Si les tailles des tableaux sont différentes la fonction renvoie le tableau vide* `{}`.

Contrat:
```
int[] t1 = {1000,1,-1,1}, int[] t2 = {-1,1,-1,-1000}
samesign(t1,t2) doit renvoyer le tableau {-1,1,1,-1}.
```

□

Exercice 3 (Variance, ⋆⋆)
Écrire une fonction `variance` *qui prend en argument un tableau d'entiers* `tab` *et qui renvoie la variance de ce tableau, c'est-à-dire la moyenne des valeurs absolues des écarts à la moyenne.* $var(x_1, x2, \dots x_k) = \frac{1}{k}\sum_{i=1}^{k} |x_i - m|$, *où* $m = \frac{1}{k}\sum_{i=1}^{k} x_i$ *est la moyenne*

Contrat:
```
 int[] t = {1000,1,1,1}
variance(t) doit renvoyer 374.
```

□

Exercice 4 (Décalage, ⋆⋆)

Écrire une fonction `shift` *qui prend en argument un tableau d'entiers* `tab` *et qui renvoie une copie de ce tableau où toutes les valeurs sont décalées d'une case vers la droite (et la dernière valeur se retrouve en position 0).*

> ***Contrat:***
> `int t[] = {1000,1,2,3}`
> `shift(t)` *doit renvoyer le tableau* `{3,1000,1,2}`.

Même chose mais avec un décalage de `n` *cases (où* `n` *est un nouveau paramètre de la fonction). Que faire quand* `n` *est négatif ?* □

Exercice 5 (Pairs et Impairs, $\star\star$ – $\star\star\star$)

Écrire une fonction `paritysort` *qui prend en argument un tableau d'entiers* `tab` *et qui renvoie un tableau contenant les valeurs de* `tab` *et tel que tous les nombres pairs se trouve à gauche des nombres impairs.*

> ***Contrat:***
> `int[] tab = {1,2,3,4,5,6,8,10}`
> `paritysort(tab)` *doit renvoyer (par exemple)* `{2,4,6,8,10,1,3,5}`.

(bonus) Modifier la fonction pour que les valeurs ayant la même parité soient triées par ordre croissant.

□

Exercice 6 (Fibonacci (bis), $\star\star$)
La suite de Fibonacci $(F_n)_{n\geq 1}$ est definie par $F_0 = 0$, $F_1 = 1$ et $F_{n+2} = F_{n+1} + F_n$. Écrire une fonction `fibonacci` *qui renvoie un tableau de taille n contenant les premiers termes de la suite.*

> ***Contrat:***
> `fibonacci(5)` *doit renvoyer* `{0,1,1,2,3}`.

□

Exercice 7 (Crible d'Ératosthène, $\star\star\star$)
Le crible d'Eratosthène est un moyen de trouver tous les nombres premiers plus petits qu'un entier donné `n`. *Pour cela on construit le tableau des nombres de 2 à n, puis, en partant de 2, on supprime (en les remplacant par 0 par exemple) tous les multiples (propres) des éléments du tableau. Ainsi, on commence par retirer du tableau tous les nombres multiples de 2 strictement plus grand que 2. Le nombre suivant non nul dans le tableau est alors 3, on supprime donc tous les multiples de 3 strictement plus grand que 3, et ainsi de suite avec le prochain élément du tableau, 5. À la fin de la procédure les seuls nombres non nuls dans le tableau sont les nombres premiers plus petits que n.*

Écrire une fonction `crible` *qui prend un paramètre un entier* `n` *et qui renvoie le tableau* `{2,3,..., n}` *auquel on a appliqué le crible d'Ératosthène.*

> ***Contrat:***
>
> `crible(10)` *doit renvoyer* `{2,3,0,5,0,7,0,0,0}`.

Écrire ensuite une fonction `prime`, *qui renvoit les n premiers nombres premiers.* □