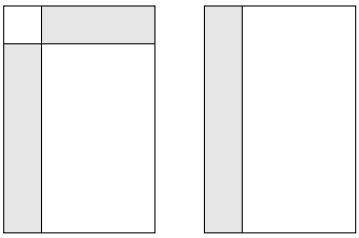
Devoir Surveillé 1 d'Informatique

- **Durée : 4 heures**
- **L'usage de tout dispositif électronique (calculatrice, téléphone portable, ...) est interdit.**
- La clarté et la lisibilité de la copie sont des qualités essentielles. Toute copie sale, peu claire ou à la rédaction approximative sera **pénalisée**. Numérotez vos feuilles doubles et indiquez votre nom sur chacune d'entre elles.



Première feuille Autres feuilles

La première page de la copie **doit laisser un cadre de 5cm**. Votre copie doit comporter une marge d'**au moins 3cm**. Toute copie ne respectant pas ces spécifications perdra des points.

- Un problème, ou un exercice à plusieurs questions, n'est pas un grand chelem, et il est approprié en l'absence de réponse à une question de supposer les résultats et de passer à la suivante. **Ne vous arrêtez pas à la première question dont vous n'avez pas la réponse !** Aucune copie ne sera pénalisée pour avoir tenté un début de réponse.
- S'il vous semble qu'une erreur s'est glissée dans un énoncé, justifiez-vous et poursuivez l'exercice *mutatis mutandis*, c'est-à-dire *en changeant ce qui doit être changé.*

Les trois parties sont **indépendantes**.

Les trois exercices visent à proposer des algorithmes efficaces pour effectuer des tâches élémentaires qui sont des briques de bases pour résoudre de nombreux problèmes. Plus précisément, nous nous attachons à extraire les éléments maximum/minimum d'un tableau (Problème 1), la plus longue chaîne d'un tableau (Problème 2) et à trier les éléments d'un tableau (Problème 3). Chacune des parties tire parti du paradigme "diviser pour régner".

Dans chacun des exercices, nous utilisons les fonctions CAML suivantes:

$min \to int \to int \to int$ et $max \to int \to int \to int$

Algorithme 1.

```
let min a b =
    if a < b then a
    else b;;
```

Algorithme 2.

```
let max a b =
    if a > b then a
    else b;;
```

1 Recherche d'extrema dans un tableau

Dans ce problème, nous considérons des tableaux dont les éléments sont des entiers naturels. C'est-à-dire, pour tout tableau T de longueur $n \in \mathbb{N}^*$, $T.(i) \in \mathbb{N}$ pour tout $0 \leqslant i < n$.

Le but de l'exercice est de proposer un algorithme efficace qui, étant donné un tableau T, renvoie le maximum et le miminum contenus dans le tableau. Plus précisément, étant donné un tableau T de longueur n, l'algorithme doit retourner la paire d'entiers $(min_T; Max_T) = (\min_{0\leqslant i<n} T.(i); \max_{0\leqslant i<n} T.(i))$.

La complexité des algorithmes de ce problème est exprimée en terme de nombre de comparaisons d'entiers.

Question 1	Ecrire en CAML un algorithme itératif (avec une boucle "for") $extr(T)$ qui prend une entrée un tableau T et renvoie la paire $(min_T; Max_T)$.

Question 2	Prouver que le nombre de comparaisons réalisées par l'algorithme $extr(T)$ proposé à la question précédente, est équivalent à $2n$, avec n la longueur de T.

Soient T un tableau de longueur $n \in \mathbb{N}^*$, $0 \leqslant a \leqslant b < n$. Considérons l'algorithme suivant:

Algorithme 3.

```
Let rec extr2 T a b =
    if a == b then (T.(a), T.(a))
    else
        if a == b − 1 then
            if T.(a) < T.(b) then (T.(a), T.(b)) else (T.(b), T.(a))
        else
            let m = (a + b)/2 in
            let (x, y) = extr2 T a m in
            let (u, v) = extr2 T m+1 b in
            (min x u , max b v);;
```

Question 3	Que calcule l'algorithme précédent ? Prouvez qu'il termine et qu'il est correct (on supposera que $0 \leqslant a \leqslant b < n$).

Question 4	Soit $x = b - a$ et posons u_x le nombre de comparaisons réalisées lors de l'appel de extr2 T a b. Exprimer la relation de récurrence satisfaite par $(u_x)_{x\geqslant 0}$.

Question 5	Donner explicitement l'expression de $(u_x)_{x\geqslant 0}$ lorsque $x = 2^p$. On posera $v_p = u_{2^p}$ pour tout $p \geqslant 0$ et on étudiera la suite $(v_p + 2)_{p\geqslant 0}$.

Question 6	Donner un algorithme, plus efficace de celui de la question 1, qui calcule (min_T, max_T).

2 Plus grande sous-séquence d'un tableau

Le but de ce problème est d'étudier différents algorithmes pour calculer le maximum des sommes d'éléments consécutifs d'un tableau donné d'entiers relatifs.

Plus précisément, soit T un tableau de n éléments à valeur dans $\mathbb{Z}$.

Pour tout $0 \leqslant a \leqslant b < n$, on note $t(a,b) = \sum_{k=a}^{b} T.(k)$. On cherche la plus grande somme de ce type. C'est-à-dire, on cherche à calculer $S(T) = \max_{0 \leqslant a \leqslant b < n} t(a,b)$.

Par exemple, pour $T = [-3,5,-4,8,-2]$, $S(T) = t(1,3) = 9$. Pour $T = [-5,-7,-1,-4]$, $S(T) = t(2,2) = -1$.

Les 5 sous-parties peuvent être traitées indépendamment.

2.1 Algorithme Naïf

On admet que l'algorithme suivant calcule effectivement $S(T)$.

Algorithme 4.

```
Let SomMax1 T =
    let n = vect_length T in
    let sMax = ref T.(0) in
    for a = 0 to n − 1 do
        for b = a to n − 1 do
            let m = ref T.(a) in
            for k = a + 1 to b do m := !m + T.(k) done;
            sMax := max m !sMax
        done;
    done;
    !sumMax;;
```

Question 7	Calculer le nombre d'itérations de l'opération "m := !m+T.(k)". En déduire l'ordre de grandeur de la complexité de l'algorithme $SomMax_1$ en fonction de la longueur n de son entrée.

2.2 Première amélioration

On se propose de supprimer l'une des boucles "for" imbriquées de la fonction $SomMax_1$. Pour cela, on peut mettre la variable $sMax$ à jour au fur et à mesure du calcul des $t(a,b) = \sum_{k=a}^{b} T.(k)$ pour a fixé et b variant de a à $n-1$.

Question 8	En suivant la suggestion ci-dessus, écrire en CAML la fonction $SomMax_2$ qui prend en entrée un tableau T et renvoie $S(T)$. Prouver que le nombre d'additions réalisées par $SomMax_2$ est $\Theta(n^2)$.

2.3 Version Récursive

Soit T un tableau de longueur n et soit $k \leqslant n$. On note $S(T,k) = \max_{0 \leqslant a \leqslant b < k} t(a,b)$.

Remarquez que $S(T,n) = S(T)$.

Question 9	Exprimer $S(T,n)$ en fonction de $S(T,n-1)$ et de $t(a,n-1)$, pour $0 \leqslant a < n$.

Question 10	Écrire en CAML une fonction récursive $SomMax_3$ qui prend en entrée un tableau T et renvoie $S(T)$. Calculer l'ordre de grandeur du nombre d'additions réalisées par $SomMax_3$.

2.4 Diviser pour régner

Soit T un tableau de longueur n et soit $0 \leqslant a \leqslant b < n$.

Posons $t_{aux}(a, b) = \max_{a \leqslant x \leqslant y \leqslant b} t(x, y)$. C'est-à-dire, $t_{aux}(a, b)$ est le maximum des sommes d'éléments consécutifs compris entre les indices a et b de T.

Soit $a \leqslant m \leqslant b$. On pose finalement $t_{join}(a, m, b) = \max_{a \leqslant x \leqslant m \leqslant y \leqslant b} t(x, y)$. C'est-à-dire, $t_{aux}(a, b)$ est le maximum des sommes d'éléments consécutifs compris entre les indices a et b de T et contenant l'élément d'indice m.

Question 11	Soit $a \leqslant m < b$. Exprimer $t_{aux}(a, b)$ en fonction de $t_{aux}(a, m)$, $t_{aux}(m + 1, b)$ et $t_{join}(a, m, b)$.

Question 12	Écrire en CAML une fonction $Junction(T, a, m, b)$ qui prend en entrée un tableau T de longueur n et trois entiers $0 \leqslant a \leqslant m \leqslant b < n$ et calcule $t_{join}(a, m, b)$ avec un nombre d'additions de l'ordre de $b - a$.

Dans la suite on pourra admettre le résultat de la question précédente.

Question 13	Écrire en CAML une fonction récursive $SumAux(T, a, b)$ qui prend en entrée un tableau T de longueur n et deux entiers $0 \leqslant a \leqslant b < n$ et calcule $t_{aux}(a, b)$.

Question 14	En déduire en CAML une fonction récursive $SomMax_4$ qui prend en entrée un tableau T et calcule $S(T)$.

Question 15	Soit u_n le nombre d'additions réalisées par la fonction $SomMax_4$ appliquée à un tableau de longueur n. Exprimer la relation de récurrence satisfaite par $(u_n)_{n \geqslant 0}$. Conclusion ?

2.5 Programmation dynamique

Dans cette section, nous étudions un algorithme encore plus efficace.

Soit T un tableau de longueur n et soit $k \leqslant n$. Rappelons que $S(T, k) = \max_{0 \leqslant a \leqslant b < k} t(a, b)$.

De plus, posons $R_k = \max_{0 \leqslant a < k} t(a, k - 1)$.

Question 16	Pour $1 < k \leqslant n$, exprimer R_k en fonction de R_{k-1}, $S(T, k - 1)$ et $T.(k - 1)$. En déduire la valeur de $S(T, k)$.

Question 17	En déduire en CAML une fonction $SomMax_5$ qui prend en entrée un tableau T et calcule $S(T)$ en temps linéaire en la taille de T.

3 Tri rapide et complexité en moyenne

Question 18	Rappeler brievement (pas plus de 5 lignes) le principe de l'algorithme de tri-fusion et donner sa complexité (sans preuve).

Dans ce problème, nous nous intéressons à un autre algorithme de tri, appelé *Tri rapide* (*quick sort* en anglais). On rappelle que, étant donné un tableau d'entiers, le but est de renvoyer un tableau avec les mêmes éléments, ordonnés dans l'ordre croissant.

Le principe du quick sort est de partitionner le tableau à trier (s'il a au moins deux éléments) en trois sous-tableaux, le premier comprenant tous les éléments inférieurs ou égaux à un élément (l'élément pivot), le deuxième ne contenant qu'un seul élément (l'élément pivot) et le troisième contenant tous les éléments supérieurs ou égaux à l'élément pivot. Puis on réapplique **récursivement** le quick sort sur les premier et troisième sous-tableau (l'élément pivot, lui, est à sa place définitive).

On considère tout d'abord la fonction pivot, décrite ci-dessous, qui prend en entrée un tableau T de longueur n et deux entiers $0 \leqslant a \leqslant b < n$.

Algorithme 5.

```
let rec pivot T a b=
    if a == b then a
    else
        let aux = T.(a + 1) in
        if T.(a) > aux then T.(a + 1) ← T.(a); T.(a) ← aux; pivot T (a + 1) b
        else T.(a + 1) ← T.(b); T.(b) ← aux; pivot T a (b − 1)
```

Question 19	Soit $T = [4, 3, 7, 9, 2]$. Que retourne pivot T 0 4 ? Quelle est la valeur de T après l'exécution de cette fonction ?

Question 20	Quel est le nombre de comparaisons d'entiers réalisées par pivot T a b ?

Algorithme 6.

```
let rec quickSort T =
    let n = vect_length T in
        let rec quickAux T a b=
            if a == b then T
            else let i = pivot T a b in quickAux (quickAux T a i) i+1 b;;
        in quickAux T 0 (n − 1);;
```

Question 21	Soit $T = [1, 2, 3, \cdots, n]$. Quel est le nombre de comparaisons d'entiers réalisées par quickSort T ? Justifiez.

D'après la question précédente, il existe des "mauvais" tableaux pour lesquels l'algorithme quickSort se comporte "mal": sa complexité est quadratique en la longueur du tableau.

Dans le pire cas, l'algorithme de tri-fusion est donc meilleur que quickSort. Cependant, nous allons voir que quickSort est efficace "en moyenne".

Dans la suite on considère des tableaux qui correspondent aux **permutations** de $[\![1, n]\!]$. C'est-à-dire, un tableau de longueur n contient des éléments, deux-à-deux distincts, dans $[\![1, n]\!]$. Soit σ_n l'ensemble de ces tableaux.

Question 22	Calculer le cardinal de σ_n.

Soit $T \in \sigma_n$ un tableau. On note $\mathcal{C}(T)$ le nombre de comparaisons d'entiers réalisées par quickSort T.

Jusqu'à présent, on s'intéressait à la complexité en pire cas, c'est-à-dire au $\max_{T \in \sigma_n} \mathcal{C}(T)$ (le tableau qui nécessite le plus de comparaisons).

Le but de cet exercice est de calculer la complexité en moyenne définie par $C_n = \frac{1}{|\sigma_n|} \sum_{T \in \sigma_n} \mathcal{C}(T)$.

Pour cela, pour tout $1 \leqslant k \leqslant n$, soit $\sigma_n^k \subseteq \sigma_n$ l'ensemble des tableaux dont le premier élément est l'entier k.

Question 23	Prouver que $\lvert\sigma_n^k\rvert = \lvert\sigma_n\rvert / n$.

Finalement, soit $C_n(k) = \frac{1}{|\sigma_n^k|} \sum_{T \in \sigma_n^k} \mathcal{C}(T)$ la complexité en moyenne de quickSort dans l'ensemble σ_n^k.

Question 24	Prouver que $C_n = \frac{1}{n} \sum_{1 \leqslant k \leqslant n} C_n(k)$.

Question 25	Prouver que $C_n(k) = n - 1 + C_n(k-1) + C_n(n-k)$.

Question 26	En déduire que $C_n = n - 1 + \frac{2}{n} \sum_{1 \leqslant k < n} C_k$

On rappelle que $H_n = \sum_{1 \leqslant k \leqslant n} \frac{1}{k} \sim \log n$

Question 27	Prouver que $\frac{C_n}{n+1} = 2H_{n+1} + O(1)$. *indice: calculer d'abord* $(n+1)C_{n+1} - nC_n$ *pour prouver que* $\frac{C_{n+1}}{n+2} = \frac{C_n}{n+1} + \frac{4}{n+2} - \frac{2}{n+1}$

Question 28	Conclusion ?