A 2011 INFO. MP

ÉCOLE NATIONALE DES PONTS ET CHAUSSÉES,
ÉCOLES NATIONALES SUPÉRIEURES DE L'AÉRONAUTIQUE ET DE L'ESPACE,
DE TECHNIQUES AVANCÉES, DES TÉLÉCOMMUNICATIONS,
DES MINES DE PARIS, DES MINES DE SAINT-ÉTIENNE, DES MINES DE NANCY,
DES TÉLÉCOMMUNICATIONS DE BRETAGNE,
ÉCOLE POLYTECHNIQUE (FILIÈRE TSI)

CONCOURS D'ADMISSION 2011

ÉPREUVE D'INFORMATIQUE

Filière MP
Durée de l'épreuve : 3 heures.
L'utilisation d'une calculatrice est autorisée.

Sujet mis à disposition des concours : ENSAE ParisTech, TELECOM SudParis (ex-INT), TPE-EIVP

L'énoncé de cette épreuve comporte 8 pages.

Les candidats sont priés de mentionner de façon apparente sur la première page de la copie :

INFORMATIQUE - MP

Recommandations aux candidats

- **Si, au cours de l'épreuve, un candidat repère ce qui lui semble être une erreur d'énoncé, il le signale sur sa copie et poursuit sa composition en expliquant les raisons des initiatives qu'il est amené à prendre.**
- **Tout résultat fourni dans l'énoncé peut être utilisé pour les questions ultérieures même s'il n'a pas été démontré.**
- **Il ne faut pas hésiter à formuler les commentaires qui semblent pertinents même lorsque l'énoncé ne le demande pas explicitement.**

Composition de l'épreuve

L'épreuve comporte :

Exercice sur les automates

On considère dans cet exercice des mots définis sur l'alphabet $\Sigma = \{a, b\}$.
Le *transposé* (ou *miroir*) d'un mot $u = u_1 \ldots u_n$, où les u_i $(1 \leq i \leq n)$ sont des éléments de Σ, est le mot, noté $\overline{u}$, qui s'écrit $u_n \ldots u_1$. Ainsi, le transposé de *abbbbaba* est *ababbbba*. Un mot u est un *palindrome* s'il est identique à son transposé : $u = \overline{u}$. Pour un entier k donné, le *préfixe* (respectivement, *suffixe*) de longueur k d'un mot u de longueur au moins k est le sous-mot formé des k premiers (respectivement, derniers) symboles de u. Ainsi, le préfixe de longueur 3 de *abbbbaba* est *abb* tandis que son suffixe de longueur 4 est *baba*.

Pour tout entier $n \geq 1$, on définit le langage L_n sur l'alphabet Σ de la manière suivante : L_n est l'ensemble des mots de longueur supérieure ou égale à $2n$ dont le suffixe de longueur n est le transposé du préfixe de longueur n. Ainsi, *abbbbaba* appartient à L_1 (car a est le transposé de a) et à L_2 (car ba est le transposé de ab) mais *abbbbaba* n'est pas dans L_3.

❒ 1 – Donner une expression rationnelle décrivant le langage L_1.

❒ 2 – Construire un automate A non déterministe reconnaissant le langage L_2. On impose que A ait un seul état initial et un seul état final ; par ailleurs, les transitions de A seront étiquetées par des éléments de Σ.

❒ 3 – Déterminiser l'automate obtenu à la question précédente. Il n'est pas nécessaire de détailler le processus de déterminisation.

❒ 4 – En s'inspirant des questions précédentes, montrer que L_n est un langage rationnel pour tout $n \geq 1$.

❒ 5 – Soit $n \geq 1$. On considère maintenant le langage L'_n formé de tous les mots de longueur strictement inférieure à $2n$, ainsi que des mots de longueur supérieure ou égale à $2n$ dont le suffixe de longueur n est le transposé du préfixe de longueur n. Indiquer si L'_n est un langage rationnel et prouver la réponse.

❒ 6 – Montrer que le langage des palindromes sur l'alphabet $\{a, b\}$ n'est pas rationnel.

❒ 7 – Montrer que l'intersection d'une suite infinie de langages rationnels n'est pas nécessairement un langage rationnel.

Problème d'algorithmique et programmation : ordres pour un tournoi

Préliminaire concernant la programmation. Il faudra écrire des fonctions ou des procédures à l'aide d'un langage de programmation qui pourra être soit **Caml**, soit **Pascal**, tout autre langage étant exclu. **Indiquer en début de problème le langage de programmation choisi ; il est interdit de modifier ce choix au cours de l'épreuve**. Certaines questions du problème sont formulées différemment selon le langage de programmation ; cela est indiqué chaque fois que nécessaire. Par ailleurs, pour écrire une fonction ou une procédure en langage de programmation, le candidat pourra définir des fonctions ou des procédures auxiliaires qu'il explicitera, ou faire appel à d'autres fonctions ou procédures définies dans les questions précédentes.
Dans l'énoncé du problème, un même identificateur écrit dans deux polices de caractères différentes désigne la même entité, mais du point de vue mathématique pour la police écrite en italique (par exemple : T) et du point de vue informatique pour celle écrite en romain (par exemple : `T`).

Dans ce problème, on note faux et vrai les deux valeurs possibles d'une variable booléenne. On considérera des matrices carrées ; les colonnes et les lignes d'une matrice carrée de dimension $n \times n$ seront toujours numérotées de 0 à $n-1$. Si T est une matrice carrée de dimension $n \times n$, pour $0 \leq i \leq n-1$ et $0 \leq j \leq n-1$, $t_{i,j}$ représentera le coefficient de T situé sur la ligne i et la colonne j. On appelle *tournoi* une matrice carrée T à coefficients booléens qui, si la matrice est de dimension $n \times n$, vérifie

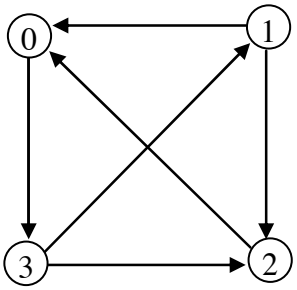
- pour $0 \leq i \leq n-1$ et $0 \leq j \leq n-1$ avec $i \neq j$: $t_{i,j}$ = vrai $\Leftrightarrow$ $t_{j,i}$ = faux ;
- pour $0 \leq i \leq n-1$, $t_{i,i}$ = faux.

Si la matrice est de dimension $n \times n$, le tournoi est dit d'*ordre n*. L'ordre n des tournois considérés dans ce problème sera toujours au moins égal à 1.

On représentera un tournoi T par un dessin de la façon suivante : à chaque entier i vérifiant $0 \leq i \leq n-1$, on fait correspondre un cercle contenant l'entier i ; pour tout couple d'entiers i et j vérifiant $0 \leq i \leq n-1$, $0 \leq j \leq n-1$ et $i \neq j$, si $t_{i,j}$ vaut vrai on trace une flèche du cercle contenant i au cercle contenant j ; on dira que ce dessin est un *graphe G* qui représente T. Le tournoi T_4 défini à gauche ci-dessous est représenté par le graphe G_4 qui se trouve à sa droite.

$$\begin{pmatrix} \text{faux} & \text{faux} & \text{faux} & \text{vrai} \\ \text{vrai} & \text{faux} & \text{vrai} & \text{faux} \\ \text{vrai} & \text{faux} & \text{faux} & \text{faux} \\ \text{faux} & \text{vrai} & \text{vrai} & \text{faux} \end{pmatrix}$$

Le tournoi T_4

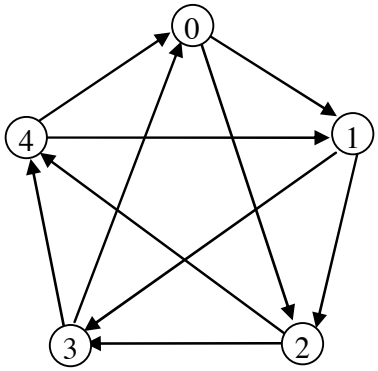


Le graphe G_4

On utilisera aussi le tournoi T_5 défini à gauche ci-dessous et représenté par le graphe G_5 qui se trouve à sa droite.

$$\begin{pmatrix} \text{faux} & \text{vrai} & \text{vrai} & \text{faux} & \text{faux} \\ \text{faux} & \text{faux} & \text{vrai} & \text{vrai} & \text{faux} \\ \text{faux} & \text{faux} & \text{faux} & \text{vrai} & \text{vrai} \\ \text{vrai} & \text{faux} & \text{faux} & \text{faux} & \text{vrai} \\ \text{vrai} & \text{vrai} & \text{faux} & \text{faux} & \text{faux} \end{pmatrix}$$

Le tournoi T_5

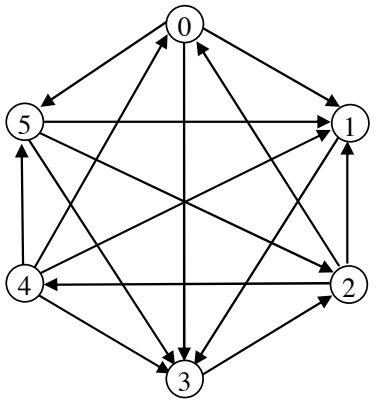


Le graphe G_5

On utilisera enfin le tournoi T_6 défini à gauche ci-dessous et représenté par le graphe G_6 qui se trouve à sa droite.

$$\begin{pmatrix} \text{faux} & \text{vrai} & \text{faux} & \text{vrai} & \text{faux} & \text{vrai} \\ \text{faux} & \text{faux} & \text{faux} & \text{vrai} & \text{faux} & \text{faux} \\ \text{vrai} & \text{vrai} & \text{faux} & \text{faux} & \text{vrai} & \text{faux} \\ \text{faux} & \text{faux} & \text{vrai} & \text{faux} & \text{faux} & \text{faux} \\ \text{vrai} & \text{vrai} & \text{faux} & \text{vrai} & \text{faux} & \text{vrai} \\ \text{faux} & \text{vrai} & \text{vrai} & \text{vrai} & \text{faux} & \text{faux} \end{pmatrix}$$

Le tournoi T_6



Le graphe G_6

On s'intéresse à un jeu nommé *J* qui se joue à deux joueurs ; pour chaque partie du jeu *J*, il y a un gagnant et un perdant, il n'y a pas de match nul. Soit n un entier strictement positif. On considère un ensemble de n joueurs. Une *compétition C* du jeu *J* effectuée par les n joueurs consiste à ce que chaque joueur joue une et une seule fois au jeu *J* contre chaque autre joueur. Les joueurs sont identifiés par des numéros allant de 0 à $n - 1$. Le résultat de cet ensemble de $n(n-1)/2$ parties est représenté par un tournoi *T* d'ordre n : pour i et j distincts vérifiant les inégalités $0 \leq i \leq n-1$ et $0 \leq j \leq n-1$, le coefficient $t_{i,j}$ de *T* vaut vrai si le joueur i a gagné contre le joueur j et faux sinon ; pour $0 \leq i \leq n-1$, le coefficient $t_{i,i}$ vaut faux. On dira que *le tournoi T représente le résultat de la compétition C*.
Par exemple, s'il y a quatre joueurs et que la compétition est représentée par le tournoi T_4 ci-dessus :

- le joueur 0 a gagné contre le joueur 3 et perdu contre les joueurs 1 et 2,
- le joueur 1 a gagné contre les joueurs 0 et 2 et perdu contre le joueur 3,
- le joueur 2 a gagné contre le joueur 0 et perdu contre les joueurs 1 et 3,
- le joueur 3 a gagné contre les joueurs 1 et 2 et perdu contre le joueur 0.

On appelle *classement d'ordre n* toute permutation des entiers 0, 1, 2, …, $n - 1$. Un classement σ d'ordre n sera noté $(\sigma(0), \sigma(1), \ldots, \sigma(n-1))$. Après une compétition entre n joueurs, un classement σ d'ordre n est interprété comme un classement des joueurs par résultats décroissants ; le joueur $\sigma(0)$ est considéré comme étant le meilleur tandis que le joueur $\sigma(n-1)$ est considéré comme étant le moins bon ; un joueur a est mieux placé qu'un joueur b si le joueur a apparaît avant le joueur b dans le classement ; par exemple, pour quatre joueurs, le classement (1, 3, 2, 0) est interprété comme : 1 est meilleur que 3 qui est meilleur que 2 qui est meilleur que 0.
Après une compétition, on peut chercher à déterminer le classement qui représente le mieux le résultat de la compétition ; il y a différentes méthodes permettant de faire cela, ces méthodes ne donnent pas toutes le même classement ; on étudie deux d'entre elles dans ce problème : *la méthode de Copeland* et *la méthode de Slater*.

Indications pour Caml

Soit n un entier. Un vecteur de n vecteurs de longueur n est appelé *matrice de dimension* $n \times n$. Un tournoi d'ordre n sera codé en Caml par une matrice de dimension $n \times n$ de booléens. Par exemple, le tournoi T_4 défini ci-dessus sera codé de la façon suivante :

```
let T = [|[|false; false; false; true|];
          [|true; false; true; false|];
          [|true; false; false; false|];
          [|false; true; true; false|];|];;
```

Un classement d'ordre n sera codé par un vecteur d'entiers de dimension n. Par exemple, le classement $(1, 3, 2, 0)$ sera codé par :

```
let classement = [|1; 3; 2; 0|];;
```

Fin des indications pour Caml

Indications pour Pascal

On définit la constante et les types ci-dessous :

```
const MAX = 100;
type Matrice = array[0..MAX - 1, 0..MAX - 1] of Boolean;
type Vecteur = array[0..MAX - 1] of Integer;
```

Un tournoi d'ordre n sera codé par une variable de type `Matrice`. On supposera qu'on ne traite que des tournois d'ordre au plus `MAX`. Par exemple, le tournoi T_4 sera codé par une variable de type `Matrice` nommée `T` de la façon suivante :

```
T[0, 0] := false;   T[0, 1] := false;
T[0, 2] := false;   T[0, 3] := true;
T[1, 0] := true;    T[1, 1] := false;
T[1, 2] := true;    T[1, 3] := false;
T[2, 0] := true;    T[2, 1] := false;
T[2, 2] := false;   T[2, 3] := false;
T[3, 0] := false;   T[3, 1] := true;
T[3, 2] := true;    T[3, 3] := false;
```

Un classement d'ordre n sera codé par une variable de type `Vecteur`. Par exemple, le classement $(1, 3, 2, 0)$ sera codé par une variable de type `Vecteur` nommée `classement` de la façon suivante :

```
classement[0] := 1; classement[1] := 3;
classement[2] := 2; classement[3] := 0;
```

Fin des indications pour Pascal

Première partie : méthode de Copeland

Dans un tournoi T, on appelle *score* de i (pour $0 \leq i \leq n-1$) le nombre de termes égaux à vrai sur la ligne i de T ; par rapport à la compétition du jeu J décrite ci-dessus, le score de i est le nombre de parties gagnées par le joueur i. Par exemple, si la compétition est décrite par T_4, les scores des joueurs 0, 1, 2 et 3 sont respectivement égaux à 1, 2, 1 et 2.

Soit T un tournoi représentant une compétition C ; un classement est appelé *classement de Copeland pour le tournoi T* si les scores des joueurs dans ce classement sont décroissants au sens large. Le classement (1, 3, 0, 2) est un classement de Copeland pour le tournoi T_4 puisque, dans l'ordre de ce classement, les scores forment la suite décroissante 2, 2, 1 et 1.

❒ 8 – Indiquer un classement de Copeland pour le tournoi T_6.

❒ 9 – La fonction `calculer_scores`.

Caml : Écrire en Caml une fonction `calculer_scores` telle que, si `T` est une matrice de booléens codant un tournoi d'ordre n, alors `calculer_scores T` renvoie un vecteur d'entiers de longueur n contenant, pour i qui varie de 0 à $n-1$, le score du joueur i à l'indice i.

Indiquer la complexité de cette fonction.

Pascal : Écrire en Pascal une fonction `calculer_scores` telle que, si `T` est de type `Matrice` et code un tournoi T d'ordre n, alors `calculer_scores(T, n)`

renvoie un tableau de type `Vecteur` contenant, pour i qui varie de 0 à $n - 1$, le score du joueur i dans la case d'indice i.
Indiquer la complexité de cette fonction.

❒ 10 – La fonction `classement_Copeland`.

Caml : Écrire en Caml une fonction `classement_Copeland` telle que, si `T` est une matrice de booléens codant un tournoi T d'ordre n, alors `classement_Copeland T` renvoie un vecteur d'entiers de longueur n contenant un classement de Copeland pour T.
Indiquer la complexité de cette fonction.

Pascal : Écrire en Pascal une fonction `classement_Copeland` telle que, si `T` est de type `Matrice` et code un tournoi T d'ordre n, alors `classement_Copeland(T, n)` renvoie un tableau de type `Vecteur` contenant un classement de Copeland pour T.
Indiquer la complexité de cette fonction.

Deuxième partie : valeur de Slater d'un classement

Soient $T = \left(t_{i,j}\right)_{0 \le i \le n-1,\, 0 \le j \le n-1}$ un tournoi d'ordre n représentant le résultat d'une compétition C et σ un classement d'ordre n ; on dit qu'une partie jouée entre deux joueurs i et j pendant la compétition C est *contredite* par le classement σ si i est avant j dans σ alors que i a perdu la partie contre j, ou bien si j est avant i dans σ alors que j a perdu la partie contre i. On appelle *valeur de Slater du classement σ pour T* et on note $Slater(T, \sigma)$ le nombre de parties de C contredites par σ. Autrement dit, $Slater(T, \sigma)$ est le nombre de couples (i, j) vérifiant simultanément :

- $0 \le i < j \le n - 1$,
- $t_{\sigma(i),\, \sigma(j)}$ = faux.

À titre d'exemple, considérons le classement $\sigma_4 = (1, 3, 2, 0)$; pour calculer $Slater(T_4, \sigma_4)$, on peut examiner successivement les valeurs de $t_{1,3}$ (qui vaut faux et contribue donc pour 1), de $t_{1,2}$ (qui vaut vrai), de $t_{1,0}$ (qui vaut vrai), de $t_{3,2}$ (qui vaut vrai), de $t_{3,0}$ (qui vaut faux et contribue donc pour 1), de $t_{2,0}$ (qui vaut vrai) : $Slater(T_4, \sigma_4)$ vaut donc 2, les parties contredites par le classement étant la partie entre 1 et 3 et la partie entre 0 et 3.

❒ 11 – En posant $\sigma_5 = (2, 4, 0, 1, 3)$, calculer $Slater(T_5, \sigma_5)$.

❒ 12 – En notant σ_6 le classement de Copeland pour T_6 déterminé à la question ❒ 8, calculer $Slater(T_6, \sigma_6)$.

❒ 13 – La fonction `valeur_Slater`.

Caml : Écrire en Caml une fonction `valeur_Slater` telle que, si `T` est une matrice de booléens codant un tournoi T d'ordre n et si `sigma` est un vecteur codant un classement σ d'ordre n, alors `valeur_Slater T sigma` renvoie $Slater(T, \sigma)$.
Indiquer la complexité de la fonction `valeur_Slater`.

Pascal : Écrire en Pascal une fonction `valeur_Slater` telle que, si `T`, de type `Matrice`, code un tournoi T d'ordre n, si `sigma`, de type `Vecteur`, code un classement σ d'ordre n, alors `valeur_Slater(T, sigma, n)` renvoie $Slater(T, \sigma)$.
Indiquer la complexité de la fonction `valeur_Slater`.

Troisième partie : indice de Slater d'un tournoi

On appelle *indice de Slater d'un tournoi T d'ordre n* et on note $s(T)$ le minimum de $Slater(T, \sigma)$ sur l'ensemble des classements σ d'ordre n.

Un *classement de Slater* pour un tournoi T est un classement σ d'ordre n vérifiant $Slater(T, \sigma) = s(T)$. Ainsi, un classement de Slater pour T contredit un minimum de parties de la compétition représentée par le tournoi T.

❒ 14 – Calculer $s(T_4)$ et indiquer un classement de Slater pour T_4.

Soit T un tournoi d'ordre n. Soient i et j deux entiers distincts compris entre 0 et $n - 1$; on dit que (i, j) *est un arc de* T si $t_{i,j}$ vaut vrai ; cela signifie aussi que le joueur i a gagné contre le joueur j dans la compétition représentée par T, ou bien encore qu'il y a une flèche de i vers j dans le graphe illustrant T.

Soit $p \geq 3$; on considère p entiers distincts, $i_1, i_2, ..., i_p$, compris entre 0 et $n - 1$; on dit que $(i_1, i_2, ..., i_p)$ est un *circuit* de T si (i_1, i_2), (i_2, i_3), …, (i_{p-1}, i_p), (i_p, i_1) sont des arcs de T (autrement dit, i_1 a gagné contre i_2, i_2 a gagné contre i_3, ..., i_{p-1} a gagné contre i_p et enfin i_p a gagné contre i_1) ; on dira que (i_1, i_2), (i_2, i_3), ..., (i_{p-1}, i_p), (i_p, i_1) sont les arcs de ce circuit.

Si p vaut 3, on dit qu'il s'agit d'un *triangle* de T. On dit que deux circuits de T sont *arc-disjoints* s'ils n'ont pas d'arc en commun. Par exemple, dans T_5, les circuits (0, 1, 3) et (0, 2, 3, 4) sont arc-disjoints.

❒ 15 – On suppose qu'il existe m circuits deux à deux arc-disjoints dans le tournoi T. Indiquer en fonction de m un minorant de l'indice de Slater de T. Justifier la réponse.

❒ 16 – On considère un tournoi T d'ordre n, un classement σ d'ordre n et un ensemble de m circuits deux à deux arc-disjoints dans le tournoi T. On suppose que l'on a $Slater(T, \sigma) = m$. Indiquer alors ce qu'on peut dire de l'indice de Slater de T et du classement σ.

❒ 17 – Calculer $s(T_6)$ et indiquer un classement de Slater pour T_6 ; déduire de ce résultat qu'un classement de Copeland pour un tournoi T n'est pas toujours un classement de Slater pour T.

On considère un ensemble E de triangles deux à deux arc-disjoints d'un tournoi T. On dit que cet ensemble est un ensemble *maximal de triangles deux à deux arc-disjoints* de T si tout triangle de T possède au moins un arc en commun avec au moins un triangle de E. On remarquera que l'ensemble E n'est pas nécessairement de cardinal maximum parmi les ensembles E de triangles deux à deux arc-disjoints du tournoi T.

❒ 18 – Indiquer un ensemble maximal de triangles deux à deux arc-disjoints de T_5.

❒ 19 – La fonction `compter_triangles`.

Il s'agit de calculer le cardinal d'un ensemble maximal de triangles deux à deux arc-disjoints d'un tournoi T d'ordre n. Pour cela, la fonction `compter_triangles` construit une suite de triangles deux à deux arc-disjoints (suite dont il n'est pas nécessaire de mémoriser les éléments) jusqu'à avoir un ensemble maximal. La complexité de cette fonction devra être de l'ordre de n^3, on ne justifiera pas cette complexité.

Caml : Écrire en Caml une fonction `compter_triangles` telle que, si `T` est une matrice de booléens codant un tournoi T d'ordre n, alors `compter_triangles T` renvoie le cardinal d'un ensemble maximal de triangles deux à deux arc-disjoints de T.

Pascal : Écrire en Pascal une fonction `compter_triangles` telle que si `T`, de type `Matrice`, code un tournoi T d'ordre n, alors `compter_triangles(T, n)` renvoie le cardinal d'un ensemble maximal de triangles deux à deux arc-disjoints de T.

Quatrième partie : méthode de Slater

On se propose d'écrire en langage de programmation une fonction qui, étant donné un tournoi T d'ordre n, détermine un classement de Slater pour T. Pour cela, on énumère tous les classements possibles, c'est-à-dire toutes les permutations de $\{0, 1, \ldots, n-1\}$, on calcule pour chaque classement σ la valeur de $Slater(T, \sigma)$ et on retient un classement qui donne la plus petite valeur de cette fonction.

Pour préparer l'écriture de cette fonction, on s'intéresse, pour n fixé, à la liste des permutations de $\{0, 1, 2, \ldots, n-1\}$ triées par ordre lexicographique croissant ; par exemple, pour $n = 3$, cette liste est $(0, 1, 2), (0, 2, 1), (1, 0, 2), (1, 2, 0), (2, 0, 1), (2, 1, 0)$.

❒ 20 – On considère la liste des permutations des entiers $\{0, 1, 2, 3, 4, 5\}$ triées par ordre lexicographique croissant ; indiquer les huit premières permutations qui suivent la permutation $(1, 2, 5, 4, 0, 3)$. On ne justifiera pas la réponse.

❒ 21 – La fonction `permutation_suivante`.

La fonction `permutation_suivante` reçoit en paramètre une permutation σ des entiers $\{0, 1, 2, \ldots, n-1\}$. Si cette permutation n'est pas la permutation $(n-1, n-2, \ldots, 1, 0)$, c'est-à-dire si ce n'est pas la dernière permutation dans le tri considéré, la fonction transforme le contenu de `sigma` pour que `sigma` contienne, après l'appel de la fonction, la permutation suivant la permutation σ dans l'ordre lexicographique et renvoie la valeur `true` ; dans le cas contraire, elle renvoie la valeur `false` et ne transforme pas `sigma`.

Caml : Écrire en Caml une fonction `permutation_suivante` telle que, si n est un entier au moins égal à 1 et si `sigma` est un vecteur de longueur n contenant une permutation σ des entiers $\{0, 1, 2, \ldots, n-1\}$, alors `permutation_suivante sigma` effectue les opérations décrites ci-dessus.

Pour faciliter l'écriture de la fonction `permutation_suivante`, on pourra écrire auparavant une fonction qui échange deux valeurs d'un vecteur.

Pascal : Écrire en Pascal une fonction `permutation_suivante` telle que, si n est un entier au moins égal à 1 et si `sigma`, de type `Vecteur`, contient entre les indices 0 et $n-1$ une permutation σ des entiers $\{0, 1, 2, \ldots, n-1\}$, alors `permutation_suivante(sigma, n)` effectue les opérations décrites ci-dessus.

Pour faciliter l'écriture de la fonction `permutation_suivante`, on pourra écrire auparavant une procédure qui échange deux valeurs d'un tableau.

❒ 22 – La fonction `classement_Slater`.

Caml : Écrire en Caml une fonction non récursive `classement_Slater` telle que, si `T` est une matrice de booléens codant un tournoi T d'ordre n, alors `classement_Slater T` renvoie un vecteur de longueur n contenant un classement de Slater pour T.

Pascal : Écrire en Pascal une fonction non récursive `classement_Slater` telle que, si `T`, de type `Matrice`, code un tournoi T d'ordre n, alors `classement_Slater(T, n)` renvoie un tableau de type `Vecteur` contenant un classement de Slater pour T.