Programmation 2
Feuille de TD-TP n°4
(Pointeurs et tableaux)

Exercice 1 (TD). Tableaux et arithmétique des pointeurs

Cet exercice est une variante de l'exercice 1 de la feuille de TD n°1. Le but est de calculer, pour un tableau *tab* contenant au moins n valeurs entières, la valeur de

$$\max\{tab[j] - tab[i] \mid 0 \leqslant i \leqslant j \leqslant n-1\}.$$

On rappelle qu'il existe un algorithme récursif pour faire ce calcul :

1. on « casse » le tableau en un sous-tableau « gauche » et un sous-tableau « droit », de tailles égales à une unité près ;
2. on rappelle l'algorithme sur chacun des sous-tableaux ;
3. on renvoie le minimum des résultats des deux rappels récursifs et de la différence entre la valeur maximale du sous-tableau « droit » et la valeur minimale du sous-tableau « gauche ».

Consigne à suivre tout au long de l'exercice. Pour l'écriture des fonctions, vous ne devez pas utiliser le symbole `[` ni le symbole `]`.

Questions. [**N.B.** Le type `item` est une redéfinition du type `int`. Il est utilisé ici pour distinguer les valeurs du tableau des indices.]

1. Écrivez la définition d'une fonction `int pos_min_tab(item *tab, int n)` qui renvoie **un indice** de la valeur minimale du segment initial de taille n du tableau tab.
2. Écrivez la définition d'une fonction `int pos_max_tab(item *tab, int n)` qui renvoie **un indice** de la valeur maximale du segment initial de taille n du tableau tab.
3. Écrivez la définition d'une fonction **récursive** `item max_diff_tab(item *tab, int n)` qui renvoie $\max\{tab[j] - tab[i] \mid 0 \leqslant i \leqslant j \leqslant n-1\}$.

Exercice 2 (TD). Tableaux bidimensionnels, structures et pointeurs

Dans cet exercice, on utilise des structures à trois champs pour représenter des petites matrices (tableaux à deux dimensions) d'entiers.

Le troisième champ est un tableau d'entiers de 10 lignes et 10 colonnes ; la matrice occupe les n premières lignes et les m premières colonnes de ce tableau, si n et m sont les valeurs des premier et deuxième champs, respectivement :

```
typedef int item;
struct matrice {
      int nb_lignes;
      nb_colonnes ;
      item coeff [10][10];
};
```

Question 1. Soit la déclaration `struct matrice m = {2, 3, {{-7, 0, 4}, {1, 8, -3}}};`

1. Que donnerait l'évaluation de `m.coeff` ? Celle de `m.coeff[1]` ? Celle de `m.coeff[0][2]` ? Celle de `m.coeff[2][2]` ? Justifiez vos réponses.
2. L'expression `m.coeff[5]` peut-elle apparaître en partie gauche d'une affectation ? Justifiez votre réponse.

Question 2. Dans cette question comme dans l'exercice 1, la consigne est de n'utiliser ni le symbole `[` ni le symbole `]` pour l'écriture des fonctions.

1. Écrivez la définition d'une fonction
   `void transposer_mat(struct matrice *am_t, const struct matrice *am)`
   qui reçoit les adresses de deux matrices en arguments et modifie la première matrice, d'adresse `am_t`, de sorte qu'elle représente la transposée de la deuxième matrice, d'adresse `am`. On rappelle que la transposée d'une matrice s'obtient en échangeant le rôle des lignes et des colonnes. [**N.B.** Le « qualifieur » `const` qui apparaît dans la déclaration du deuxième paramètre d'entrée précise que les valeurs des champs de la matrice d'adresse am ne devront pas changer au cours de l'exécution de la fonction.]
2. Écrivez la définition d'une fonction
   `void reduire_mat (struct matrice *am)`
   qui reçoit en argument l'adresse d'une matrice et modifie la matrice en supprimant la première ligne et la première colonne. Les lignes et les colonnes sont donc décalées, respectivement vers le « haut » et la « gauche », et écourtées.

Exercice 3 (TP). Écart maximal dans un tableau

Écrivez un programme qui

1. lit un entier positif n compris entre 0 et 100 entrés au clavier par l'utilisateur ;
2. initialise les coefficients d'une tableau *tab* d'entiers de taille n à des valeurs entières « aléatoires » puis affiche le tableau (s'il est de taille inférieure à 20) ;
3. calcule et affiche le minimum de *tab*, son maximum et $\max\{tab[j] - tab[i] \mid 0 \leqslant i \leqslant j \leqslant n-1\}$.

Consignes pour l'écriture du programme.

- Pour le calcul du minimum, du maximum et de $\max\{tab[j] - tab[i] \mid 0 \leqslant i \leqslant j \leqslant n-1\}$ (étape 3), votre programme doit appeler les fonctions de l'exercice 1 ci-dessus.
- Pour la lecture de l'entier n (étape 1), écrivez une fonction
  `int saisir_entier(int vmin, int vmax)`
  qui demande à l'utilisateur d'entrer au clavier un entier compris entre *vmin* et *vmax* (*vmin* $\leqslant$ *vmax*), vérifie que la valeur saisie est dans l'intervalle spécifié (tant que ce n'est pas le cas, la fonction demande à l'utilisateur de recommencer) puis renvoie cette valeur.
- Pour l'initialisation « aléatoire » des valeur d'un tableau (étape 2), écrivez une fonction
  `void initialiser_alea_tab(item *tab, int taille)`
  qui reçoit un tableau comme premier paramètre d'entrée et la taille d'un segment initial comme deuxième paramètre d'entrée, et initialise les valeurs de ce segment à des entiers compris entre 0 et 99 ; pour l'utilisation des fonctions `rand` et `srand`, vous pouvez consulter les diapos du cours n°3 (31 janvier 2023), postées sur Moodle.
- Pour l'affichage d'un tableau (étape 2), écrivez une fonction
  `void afficher_tab((item *tab, int taille)`
  qui reçoit un tableau comme premier paramètre d'entrée et la taille d'un segment initial comme deuxième paramètre d'entrée, et affiche à l'écran les valeurs de ce segment.