Tableaux

1. Ecrire un algorithme **MonTableau** qui permet de définir ce tableau T = [17, 38, 10, 25, 72]. Ajouter ensuite les instructions permettant de :
   (a) afficher le sous-tableau du 2ème au dernier élément.
   (b) afficher le sous-tableau du début au 3ème élément.
   (c) renverser et afficher le tableau T.

2. Ecrire un algorithme **max** qui permet de définir un tableau T et qui calcul et affiche le plus grand élément présent dans T.

3. Ecrire un algorithme **min** qui permet de définir un tableau T et qui calcul et affiche le plus petit élément présent dans T.

4. Ecrire un algorithme **sum** qui permet de définir un tableau T et qui calcul et affiche la somme des éléments du T.

5. Ecrire un algorithme **index** qui permet de définir un tableau des entiers T et un entier x, cet algorithme permet d'afficher le premier indice de l'élément x dans T.

6. Ecrire un algorithme **count** qui permet de définir un tableau des entiers T et un entier x. Cet algorithme permet de calculer et d'afficher le nombre des éléments ayant x comme valeur dans T.

7. Ecrire un algorithme **PairImpair** qui permet de définir le tableau suivant : T = [32, 5, 12, 8, 3, 75, 2, 15]. Ajouter ensuite les instructions permettant de :
   (a) afficher les nombres pairs.
   (b) afficher les nombres impairs.

8. **Recherche dans un tableau** : Ecrire un algorithme **dichotomique** qui permet de définir un tableau trié des entiers T et un entier k. Cet algorithme permet de chercher et d'afficher la position de k dans T en utilisant la méthode de **recherche dichotomique**.
   Principe : Soient **T** un tableau déjà trié de **n** entier et **k** le nombre qu'on recherche. On compare le nombre **k** au nombre qui se trouve au milieu du **T**. Si c'est le même, on a trouvé, sinon on recommence sur la première moitié (ou la seconde) selon que **k** est plus petit (ou plus grand) que le nombre du milieu du tableau.

9. **Tri par sélection** : Ecrire un algorithme **TriSelection** qui tri un tableau d'entier T avec la méthode de tri par sélection.
   Principe : consiste à chercher le plus petit élément pour le placer en 1er, puis de chercher le plus petit élement dans le reste et de le mettre en second, etc...
   Méthode :
   (a) Chercher le minimum du tableau à partir de la case j
   (b) Le permuter avec la case j

10. **Tri à bull** : Ecrire un algorithme **TriBull** qui tri un tableau d'entier T avec la méthode de tri à bull.
Principe : On commence par i = 0, on compare l'élément T[0] et l'élément T[1], s'il ne sont pas dans le bon ordre, on les permute, on passe ensuite à l'élément T[1] et l'élément T[2], puis l'élément T[2] et T[3] et ainsi de suite jusqu'au (n-1)^{ième} T[n-1] et n^{ième} éléments T[n].
Méthode :
    (a) à partir du début du tableau, s'assurer que chaque paire de case adjacentes vérifient T[i] $\leq$ T[i+1], sinon effectuer les permutations nécessaire.
    (b) Répéter l'étape a jusqu'ce qu'il n'y a plus de changements

11. **Tri par insertion** : Ecrire un algorithme **TriInsertion** qui tri un tableau d'entier T avec la méthode de tri par insertion.
Principe : L'algorithme du est un algorithme qui insère un élément dans un tableau d'éléments déjà triés (par exemple, par ordre croissant).
C'est le tri du joueur de cartes. On fait comme si les éléments à trier étaient donnés un par un, le premier élément constituant, à lui tout seul, un tableau triée de longueur 1. On range ensuite le 2^{ième} élément pour que les 2 premiers éléments deviennent triés, puis on range le 3^{ième} élément pour que les 3 premiers éléments deviennent triés, et ainsi de suite...

12. **Les matrices** : (Tableaux à deux dimensions)
    (a) Ecrire un algorithme **matrice** qui remplit une matrice d'entiers $M_{5,4}$ (5 : nombre de lignes, 4 : nombre de colonnes).
    Afficher ensuite les éléments de la matrice élément par élément.
    (b) Ecrire un algorithme **matrice_somme** qui calcul et affiche, si c'est possible, la matrice somme S de deux matrices M et N lus au clavier.
    (c) Ecrire un algorithme **matrice_produit** qui calcul et affiche, si c'est possible, la matrice produit P de deux matrices M et N.
    (d) Ecrire un algorithme **matrice_transpose** qui calcul et retourne la matrice transposé T de la matrice M.
    (e) Ecrire un algorithme **matrice_pascale** qui calcul et retourne la matrice qui représente le triangle de pascale.