- La calculatrice est interdite.
- Seules sont autorisées l'utilisation des types de bases (entiers, flottants, chaînes de caractères, booléens), la structure de données liste (avec méthodes `append` et `pop`), les structures algorithmiques alternatives et itératives, et l'utilisation d'algorithmes récursifs.
- Si vous introduisez de nouvelles fonctions ou de nouvelles variables, on s'efforcera de leur donner un nom explicite et d'indiquer en commentaire leur rôle.
- Enfin, dans un exercice, il est possible (et même recommandé) d'utiliser les fonctions des questions précédentes, que ces questions aient été traitées ou non.

Exercice n° 1 — Quelques questions indépendantes

1. Donner la valeur des expressions suivantes :

a. `2**5`
b. `38//7`
c. `10==2*5 and (not(3**2!=9))`
d. `38%5`

2. L'instruction suivante a été validée dans la console : `s="Programmation"`
Déterminer la valeur des expressions suivantes :

a. `s[6]` **b.** `s[17]` **c.** `s[1:7]` **d.** `s[:5]`

3. On a désormais validé dans la console : `L=[1,2,3,4,5]` et `M=[6,7,8,9,0]`
Donner la valeur des expressions suivantes :

a. `L+M`
b. `L*2`

Exercice n° 2 — Recherche de maximum

Pour les différentes questions de cet exercice, il est évidemment interdit d'utiliser la fonction `max` de Python.

1. Écrire une fonction non récursive `maximum` qui prend en argument une liste non vide `L` d'entiers, et qui renvoie son maximum.
Par exemple, si `L`$= [2,7,5,8,3]$, `maximum(L)` renvoie 8.

2. Écrire une fonction récursive `maximum_rec` qui prend en argument une liste non vide `L` d'entiers, et qui renvoie son maximum.

3. Écrire une fonction `posmaximum` qui prend en argument une liste non vide `L` d'entiers, et qui renvoie la position de son maximum. S'il y a plusieurs indices possibles, la fonction renverra l'indice le plus petit.

   Par exemple, si `L`= [2,7,5,8,3], `posmaximum(L)` renvoie 3.

4. Écrire une fonction `tteposmaximum` qui prend en argument une liste non vide `L` d'entiers, et qui renvoie une liste des positions de son maximum. Il faudra parcourir le tableau une seule fois et ne pas utiliser les fonctions précédentes.

   Par exemple, si `L`= [8,7,5,8,3], `tteposmaximum(L)` renvoie `[0,3]`.

Exercice n° 3 — Nombres premiers

1. Écrire une fonction récursive `divise` qui prend en arguments deux entiers naturels a et b supérieurs ou égaux à 2, et qui renvoie un booléen de façon à ce qu'il renvoie `True` s'il existe un entier de $[\![2,a]\!]$ qui divise b.
2. En déduire une fonction `premier` qui prend en argument un entier naturel n, et qui renvoie un booléen indiquant si `n` est premier.
3. Écrire une fonction `liste_premiers` qui prend en argument un entier `n` et qui renvoie la liste des nombres premiers inférieurs ou égaux à `n`

Exercice n° 4 — Chaînes de caractères

Pour les questions ci-dessous, on utilisera la variable suivante pour illustrer :

```
texte = "L'objectif de ce texte est de servir d'exemple pour les
 différentes questions de cet exercice."
```

1. Écrire une fonction `compte_caractere` qui prend en argument une chaîne de caractères `t` et qui renvoie le nombre de caractères présents dans `t`, espaces non compris.

   Par exemple, `compte_caractere(texte)` renvoie le nombre 80, nombre de lettres et de caractères différents de l'espace.

2. Écrire une fonction `compte_mot` qui prend en argument une chaîne de caractères `t` et qui renvoie le nombre de mots présents dans `t`. Pour simplifier, on appelle mot toute chaîne de caractères encadrés par deux espaces (sauf si elle est en début ou fin de la chaîne `t`). Ainsi, `L'objectif` ou `d'exemple` comptent chacune pour un seul mot.

   Par exemple, `compte_mot(texte)` renvoie le nombre 15.

3. Écrire une fonction `recherche_index_lettre` qui prend en argument une chaîne de caractères `t` et un caractère `l`, et qui renvoie l'indice de la première occurrence du caractère `l` dans la chaîne `t` et -1 sinon.

   Par exemple, `recherche_index_lettre(texte,' ')` renvoie le nombre 10.

4. Écrire une fonction `recherche_mot_plus_long` qui prend en argument une chaîne de caractères `t` et qui renvoie le mot le plus long. Pour simplifier, on considère que la ponctuation fait partie des mots : `exercice.` est donc un mot de 9 lettres.
   Par exemple, `recherche_mot_plus_long(texte)` renvoie la chaîne `'différentes'`.

Exercice nº 5 — La multiplication égyptienne

On considère le programme suivant, qui implémente un algorithme égyptien de multiplication. Les arguments a et b sont des entiers naturels.

```
def Egyptienne(a,b) :
  t = 0
  while a > 0 :
    if a % 2 == 1 :
      t = t + b
    b = 2 * b
    a = a // 2
  return t
```

1. Que renvoie `Egyptienne(57,4)` ? On détaillera dans un tableau donnant les valeurs successives de a, b et t l'exécution de cette instruction.
2. Montrer que la fonction `Egyptienne` termine toujours.
3. En s'appuyant sur un invariant de boucle, montrer que la fonction renvoie le produit entre les deux arguments.
4. Quelle est la complexité de cette fonction ?

Exercice nº 6 — Carrés magiques

Soit $n \in \mathbb{N}^*$. Un carré magique d'ordre n est une matrice carrée d'ordre n qui contient des nombres entiers strictement positifs. Ces nombres sont disposés de sorte que les sommes sur chaque ligne, les sommes sur chaque colonne et les sommes sur chaque diagonale soient égales. La valeur de ces sommes est appelée constante magique.
Voici un carré magique d'ordre 3 :

21	7	17
11	15	19
13	23	9

Si on effectue les sommes sur les lignes, les colonnes et les diagonales, on trouve systématiquement 45.
Pour représenter une matrice carrée d'ordre n, on utilisera une liste qui contient n listes toutes de même longueur n. Pour l'exemple précédent, le tableau est représenté par la liste `M=[[21,7,17],[11,15,19],[13,23,9]]`.
On appelera diagonale de la matrice `M` la diagonale qui part du cœfficient en haut à gauche pour finir en bas à droite, et antidiagonale l'autre diagonale. Ainsi, pour l'exemple

ci-dessus, la diagonale est constituée des cœfficients 21, 15 et 9 et l'antidiagonale est constituée des cœfficients 17, 15 et 13.
Dans tout l'exercice, on considèrera que les matrices carrées sont représentées par de telles listes et on ne vérifiera ni que la matrice est bien carrée, ni que les cœfficients sont bien des entiers strictement positifs. On rappelle également que les fonctions à coder doivent être adaptées à toute taille de matrice, et non pas uniquement aux matrices de taille 3.

Partie I - Opérations sur une matrice carrée

1. Écrire une fonction `somme_ligne` qui prend comme argument une matrice carrée `M` et un entier `i` et qui renvoie la somme des termes de la ligne d'indice `i` de `M`.
   Par exemple, si `M=[[1,2,3],[4,5,6],[7,8,9]]`, `somme_ligne(M,1)` renvoie la somme $4+5+6=15$.
2. Écrire une fonction `somme_colonne` qui prend comme argument une matrice carrée `M` et un entier `j` et qui renvoie la somme des termes de la colonne d'indice `j` de `M`.
   Par exemple, si `M=[[1,2,3],[4,5,6],[7,8,9]]`, `somme_colonne(M,0)` renvoie la somme $1+4+7=12$.
3. Écrire une fonction `somme_diag1` qui prend comme argument une matrice carrée `M` qui renvoie la somme de la diagonale de `M`
   Par exemple, si `M=[[1,2,3],[4,5,6],[7,8,9]]`, `somme_diag1(M)` renvoie la somme $1+5+9=15$.
4. Écrire une fonction `somme_diag2` qui prend comme argument une matrice carrée `M` qui renvoie la somme de l'antidiagonale de `M`
   Par exemple, si `M=[[1,2,3],[4,5,6],[7,8,9]]`, `somme_diag2(M)` renvoie la somme $3+5+7=15$.

Partie II - Carré magique

5. Écrire une fonction `carré_magique` qui prend comme argument une matrice carrée `M` et qui renvoie `True` si `M` est un carré magique et `False` sinon.
   Par exemple, si `M=[[1,2,3],[4,5,6],[7,8,9]]`, `carré_magique(M)` renvoie `False` et si `M=[[21,7,17],[11,15,19],[13,23,9]]`, `carré_magique(M)` renvoie `True`

Partie III - Carré magique normal

Un carré magique normal d'ordre n est un carré magique d'ordre n constitué de tous les entiers positifs compris entre 1 et n^2. Voici par exemple un carré magique normal d'ordre 3 :

2	7	6
9	5	1
4	3	8

6. Écrire une fonction `estPresent` qui prend comme arguments une matrice carrée `M` et un entier `e` et qui renvoie `True` si l'entier `e` est l'un des éléments présents dans la matrice `M` et `False` dans le cas contraire.

Par exemple, si `M=[[21,7,17],[11,15,19],[13,23,9]]`, `estPresent(M,21)` renvoie `True` et si `M=[[2,7,6],[9,5,1],[4,3,8]]`, `estPresent(M,21)` renvoie `False`

7. Écrire une fonction `magique_normal` qui prend comme argument une matrice carrée `M` et qui renvoie `True` si `M` est un carré magique normal et `False` sinon.

Par exemple, si `M=[[21,7,17],[11,15,19],[13,23,9]]`, `magique_normal(M)` renvoie `False` et si `M=[[2,7,6],[9,5,1],[4,3,8]]`, `magique_normal(M)` renvoie `True`