

TD n°2

1 Travaux sur les valeurs numériques

Exercice 1. Le meilleur tarif

Un imprimeur propose différents tarifs pour imprimer des documents de qualité, jusqu'à 600 pages. Voici les 4 tarifs proposés :

- tarif A : 0,15 € par page, plus un forfait de 30 € ;
- tarif B : un forfait de 70 € (quel que soit le nombre de pages) ;
- tarif C : 0.05 € par page, plus un forfait de 50 € ;
- tarif D : 0.45 € par page.

1. Ecrivez les fonctions **tarifA**, **tarifB**, **tarifC** et **tarifD** qui gèrent chacune un argument (qu'on peut noter x) et qui retournent le coût du tarif en question, pour x pages.
2. Ecrivez une fonction **meilleurTarif(x)** qui donne, selon le nombre x de pages à imprimer, le plus bas des 4 tarifs. Vous pourrez utiliser la fonction native `min` qui opère sur une liste. Par exemple, `min(3, 1, 4)` retourne 1.

Remarque : Vous pourrez, au moyen d'un tableur, vérifier vos calculs.

Exercice 2. Les années bissextiles

Les années bissextiles sont :

- les années divisibles par 4 mais pas par 100 ;
- les années divisibles par 400.

Ainsi, 1904, 1908 étaient bissextiles, mais pas 1900. 2000 l'était, et 2100 ne le sera pas.

1. Ecrivez des fonctions **divisiblePar4**, **divisiblePar100** et **divisiblePar400** qui prennent en charge un argument (l'année à tester) et retournent un booléen.
2. Généralisez ces trois fonctions en écrivant une fonction **divisiblePar(*annee*, *n*)** qui retourne un booléen qui dit si l'année en question est divisible par le nombre en question.
3. Ecrivez ensuite une fonction **estBissextile** qui prend en charge un argument (l'année à tester) et qui retourne un booléen. Vous pourrez donner deux versions (avec ou sans fonction `divisiblePar` généralisée).

2 Travaux sur l'aléatoire

Exercice 1. Numéros du loto

On va écrire un programme qui génère et affiche une liste de 7 entiers distincts entre 1 et un certain nombre (au loto ancienne version, ce sera entre 1 et 49). Il ne peut pas y avoir de répétition.

Pensez à voir les méthodes sur les list...

Exemple de tirage de loto valide : 15, 26, 7, 42, 48, 1, 37

1. Ecrivez une fonction **numerosDisponibles(*n*)** qui gère un argument entier *n* et qui construit et retourne la liste des entiers entre 1 et *n*.
Par exemple l'appel **numerosDisponibles(49)** retournera la liste [1, 2, ..., 49].
2. Ecrivez une fonction **choisirParmi(*l*)** qui prend en argument une liste *l* et qui retire de cette liste un nombre choisi au hasard, et le retourne. La liste *l* est donc modifiée par la fonction.
3. Ecrivez le programme principal qui crée la liste initiale des numéros disponibles, puis qui pioche, 7 fois de suite, dans cette liste, pour constituer une liste (qu'on pourra appeler "tirage") qui sera affichée à la fin.

Exercice 2. Le premier 6

On lance un dé équilibré à 6 faces classique jusqu'à obtenir le premier 6. On note alors le nombre de lancers qui ont été nécessaires.

On recommence ainsi plusieurs expériences, dont on va stocker les résultats, pour étudier statistiquement ce phénomène de temps d'attente du premier 6.

1. Ecrivez une fonction **lancer()** qui simule une expérience en retournant **le nombre de lancers nécessaires pour obtenir un 6**.
2. Ecrivez une fonction **stats(N)** qui appelle *N* fois la fonction **lancer()** et retourne, dans une liste, la série des *N* résultats obtenus.
3. Dans le programme principal, demandez la valeur de *N*, puis affichez la liste statistique obtenue après les *N* expériences.
4. Ecrivez une fonction **moyenne(liste)** qui retourne la moyenne des valeurs de la liste. Utilisez cette fonction pour enrichir le programme principal.

Exercice 3. Points aléatoires dans un carré

On construit deux points dans un carré de côté 1, en générant pour chaque point son abscisse et son ordonnée grâce à la méthode **random()** de la bibliothèque **random** de python.

1. Ecrivez une fonction **distance(xA, yA, xB, yB)** qui retourne le résultat de la distance euclidienne entre les points $A(x_A; y_A)$ et $B(x_B; y_B)$.
Pour mémoire, $AB = \sqrt{(x_B - x_A)^2 + (y_B - y_A)^2}$
2. Ecrivez un programme qui demande à l'utilisateur le nombre *N* de tirages à réaliser, puis qui réalise *N* fois le tirage de deux points *A* et *B*, calcule la distance *AB* et compte les points obtenus sur les *N* tirages, selon le barème suivant :
 - si la distance *AB* est inférieure à 0.25, le joueur marque 30 points ;
 - sinon, si la distance *AB* est inférieure à 0.6, le joueur marque 15 points ;
 - sinon, le joueur marque 0 point.

3 Travaux sur les chaînes de caractères

Exercice 1. écrivez un programme qui demande deux phrases, et qui affiche les lettres communes (une seule fois). Les phrases seront d'abord mises en minuscules pour effectuer le travail.

Voici quelques exemples :

```
phrase 1 ? bonjour Monsieur.  
phrase 2 ? Bonsoir madame !  
voici les lettres communes : ['b', 'o', 'n', 'r', 'm', 's', 'i', 'e']
```

```
phrase 1 ? Hello gamin !  
phrase 2 ? Salut papy !  
voici les lettres communes : ['l', 'a']
```

Exercice 2. écrivez un autre programme qui demande deux phrases, et qui affiche les lettres de la phrase 1 qui n'apparaissent pas dans la phrase 2, puis celles de la phrase 2 qui n'apparaissent pas dans la phrase 1. Les phrases seront d'abord mises en minuscules pour effectuer le travail.

Voici quelques exemples :

```
phrase 1 ? bonjour papy  
phrase 2 ? bonsoir gamin  
phrase 1 moins phrase 2 : ['j', 'u', 'p', 'y']  
phrase 2 moins phrase 1 : ['s', 'i', 'g', 'm']
```

```
phrase 1 ? Salut  
phrase 2 ? Bonsoir  
phrase 1 moins phrase 2 : ['a', 'l', 'u', 't']  
phrase 2 moins phrase 1 : ['b', 'o', 'n', 'i', 'r']
```

Exercice 3. Si ce n'est pas déjà fait, reprenez les deux programmes précédents pour définir 2 fonctions *lettres_communes(phrase1, phrase2)* et *difference(phrase1, phrase2)* qui retournent les résultats attendus : les lettres communes et les deux différences.

4 Travaux sur la suite de Syracuse

Pour un entier $N \in \mathbb{N}^*$, on définit la suite (u_n) par $u_0 = N$ et la relation de récurrence : $u_{n+1} = u_n/2$ si u_n est pair, et $u_{n+1} = 3u_n + 1$ sinon.

Cela définit une suite. Par exemple, pour $N = 17$ on trouve

$$u_0 = 17, u_1 = 52, u_2 = 26, u_3 = 13, u_4 = 40, u_5 = 20 \text{ etc}$$

L'état actuel des recherches sur cette suite **laisse penser (conjecture)** que pour tout valeur initiale de N , on finit toujours par arriver au cycle de valeurs 4, 2, 1, ...

Par exemple, la suite commençant à 17 se prolonge en

$$17, 52, 26, 13, 40, 20, 10, 5, 16, 8, 4, 2, 1, 4, 2, 1, \dots$$

On conviendra qu'on s'arrête dès que le 1 est atteint.

Exercice 1. Ecrivez une fonction *syracuse* qui prend en argument un entier n , et qui retourne la liste qui donne les éléments de la suite de Syracuse pour n .

Par exemple, *syracuse(5)* retourne la liste [5, 16, 8, 4, 2, 1].

La longueur de la suite de Syracuse (jusqu'au premier 1) n'est pas un résultat régulier.

Ecrivez une fonction *longueur_syracuse* qui prend en argument un entier n, et qui retourne la longueur de la suite de Syracuse de n.

Exercice 2. Ecrivez un programme qui permet, pour tous les entiers entre deux valeurs saisies par l'utilisateur, de donner la longueur de la suite pour chacun des entiers. Voir la capture :

```
entrez la valeur minimale de N : 120
entrez la valeur maximale de N : 140

120 ... 21
121 ... 96
122 ... 21
123 ... 47
124 ... 109
125 ... 109
126 ... 109
127 ... 47
128 ... 8
129 ... 122
130 ... 29
131 ... 29
132 ... 29
133 ... 29
134 ... 29
135 ... 42
136 ... 16
137 ... 91
138 ... 16
139 ... 42
140 ... 16
```

Exercice 3. Ecrivez une fonction *recherche_longueur* qui prend en argument un entier l et qui retourne le plus petit entier n dont la suite de Syracuse a pour longueur l.

Exercice 4. (***) Cet exercice complémentaire est difficile.

Il semble y avoir des régularités ou au moins des répétitions dans les résultats. Ecrivez des fonctions et un programme qui affiche la plus longue répétition. Sur l'exemple précédent, le résultat mettrait en évidence la série de cinq 29 entre 130 et 134.

```
entrez la valeur minimale de N : 120
entrez la valeur maximale de N : 140

série demandée : [21, 96, 21, 47, 109, 109, 109, 47, 8, 122, 29, 29, 29, 29, 29, 42, 16, 91, 16, 42, 16]

plus grande série : 5 fois la valeur 29 entre N = 130 et N = 134
```

La plus longue série entre $N = 2$ et $N = 1000000$ est une série de 40 valeurs 98 consécutives entre $N = 596310$ et $N = 596349$!