Bases du langage

Un ensemble de petits exercices pour travailler les bases du langages (types de données de base, tests, boucles...).

Compteur de caractères

1. Ecrire une fonction Python `compter_lettre(txt, lettre)` qui compte le nombre de fois qu'une lettre `lettre` apparaît dans un texte `txt`. Quelques exemples d'appels de la fonction :

```
>>> compter_lettre("programmation", "p")
1
>>> compter_lettre("programmation", "p")
0
>>> compter_lettre("programmation", "p")
2
```

2. Nous cherchons à étendre notre programme précédent pour qu'il retourne le nombre de fois qu'apparaît chaque caractère dans une chaîne. Quelle structure de donnée Python vous paraît pour représenter le résultat ?
3. Codez la fonction `compter_lettres(txt)`. Exemple d'appel :

```
>>> compter_lettres("programmation")
{"p": 1, "r": 2, "o": 2, "g": 1, "a": 2, "m": 2, "t": 1, "i": 1, "n":
    1}
```

Le nombre mystère

> Remarque 1 : les instructions suivantes permettent de tirer et d'afficher un nombre entier aléatoire en 0 et 100.
>
> ```
> from random import randint
>
> a = randint(0, 100)
> print(a)
> ```
>
> Remarque 2 : pour récupérer une valeur saisie l'utilisateur, Python nous permet d'utiliser l'instruction `input()`

En vous aidant de la fonction `randint()` écrivez un programme qui doive faire deviner à l'utilisateur un nombre tirer aléatoirement.

— si le nombre saisi par l'utilisateur est inférieur au nombre mystère, le programme affichera *"Trop petit"* et proposera à l'utilisateur d'effectuer une nouvelle proposition (jusqu'à trouver le nombre mystère) ;

— si le nombre saisi par l'utilisateur est supérieur au nombre mystère, le programme affichera *"Trop grand"* et proposera à l'utilisateur d'effectuer une nouvelle proposition (jusqu'à trouver le nombre mystère) ;
— si le nombre saisi par l'utilisateur est égal au nombre mystère, le programme affichera *"Bravo, vous avez trouvé le nombre mystère en X coup"*, avec X le nombre de propositions effectuées par l'utilisateur.

Pierre/feuille/ciseaux

Nous souhaitons réaliser une jeu de pierre/feuille/ciseaux. Le principe est le suivant : * l'ordinateur tire aléatoirement une valeur parmi les suivantes : `"pierre"`, `"feuille"` ou `"ciseaux"` (cette valeur reste pour l'instant cachée à l'utilisateur) ; * l'utilisateur saisie l'une de ses trois valeurs ; * le programme affiche les valeurs choisies par l'utilisateur et l'ordinateur (par exemple `Ordinateur joue: "feuille"`) ; * le programme compare les valeurs de l'ordinateur et de l'utilisateur et affiche selon le résultat : * `Bravo! Vous avez gagné.` ; * `L'ordinateur vous a battu.` ; * `Egalité` * sachant que : * la feuille l'emporte sur la pierre ; * la pierre l'emporte sur les ciseaux : * les ciseaux l'emportent sur la feuille.

Nombre de lignes, mots, caractères

1. Ecrire une fonction `compter_caracteres(chemin_fichier)` qui retourne le nombre de caractères contenus dans un fichier texte.
2. De manière similaire, écrire une fonction `compter_mots(chemin_fichier)` retournant le nombre de mots dans un fichier texte.
3. Même chose pour `compter_lignes(chemin_fichier)` retournant le nombre de lignes.

Distance entre deux points

Nous considérons dans cette exercice des points en 2 dimensions, qui sont représentés par des tuples de la forme `(3, 2)` pour le points de coordonnées x=3 et y=2.

1. Ecrire une fonction `distance(point1, point2)` calculant la distance entre deux points.
2. Formatter le résultat de la manière suivante : `Distance entre (0, 0)et (2, 3)= 3.61`.
3. En utilisant la fonction précédente, écrire une fonction `longueur_ligne()` acceptant un nombre indéterminé de points en paramètre et retournant la longueur de la ligne qu'ils forment.
4. Formatter le résultat de la manière suivante : `Longueur de la ligne ((0, 0), (2, 3), (3, 4), (4, 4))= 14.262`