TP8 : fonctions

Initiation à l'algorithmique et la programmation

L3 TAL — Semestre 5 (2025)

Partie 1 : Premières fonctions !

Problème 1: Hello world!

Écrire une fonction `print_greetings()` qui prend en entrée une liste de prénoms, et affiche pour chaque prénom un message de salutations.

Problème 2: Opérations mathématiques

1. Écrire une fonction `multiply()` qui prend en entrée deux nombres et qui renvoie leur produit.
2. Écrire une fonction `divide()` qui prend en entrée deux nombres a et b et qui renvoie la division de a par b.
3. Écrire une fonction `eculidian_()` qui prend en entrée deux nombres a et b et qui renvoie le quotient et le reste de la division euclidienne de a par b.

Problème 3: Compter le caractère...

Écrire une fonction `count_char()` qui prend en entrée une chaîne de caractère `text`, et un caractère `c`, et qui renvoie le nombre de fois où le caractère `c` apparaît dans `text`.

Problème 4: Statistiques

Toutes les fonctions décrites ci-dessous prennent en entrée une liste de nombres (entiers, décimaux).

1. Écrire une fonction `get_average()` qui renvoie la moyenne de la liste.
2. Écrire une fonction `get_min_max()` qui renvoie le minimum et le maximum de la liste.
3. Écrire une fonction `get_sum()` qui renvoie la somme des éléments de la liste.
4. Finalement, écrire une fonction `summarize()` qui prend en entrée une liste de nombres, qui appelle les fonctions définies précédemment et qui affiche le résultat de chaque fonction. (La fonction affiche le minimum, le maximum, la moyenne et la somme des éléments de la liste)

Rappel : on appelle une fonction dans une autre fonction ainsi :

```
    def fonction1(arg):
        r = fonction2(x)
        ...
```

Problème 5: Palindrome

1. Écrire une fonction `is_palindrome()` qui prend en entrée une chaîne de caractères et renvoie un booléen indiquant si la chaîne est un palindrome. On ignorera les espaces et la casse (majuscules ou minuscules).
2. Écrire une fonction `filter_palindromes()` qui prend en entrée une liste de chaînes de caractères et renvoie une nouvelle liste contenant uniquement les palindromes.

Exemples :
`is_palindrome("kayak")` renvoie `True`
`is_palindrome("A man a plan a canal Panama")` renvoie `True`
`filter_palindromes(["kayak", "python", "radar", "code"])` renvoie `["kayak", "radar"]`

Problème 6: Voyelles (encore ?!)

1. Écrire une fonction `count_vowels()` qui prend en entrée une chaîne de caractères, et qui renvoie le nombre de voyelles qu'elle contient.
2. Écrire une fonction `count_all_vowels()` qui prend en entrée une liste de chaînes de caractères et qui renvoie une liste dont les éléments sont les comptes des voyelles de mots de la liste donnée en entrée.

Exemple : pour l'entrée `["python", "code"]`, on obtient la sortie `[2, 2]`.

Problème 7: Mots communs

Écrire une fonction `common_words()` qui prend en entrée deux listes de chaînes de caractères et renvoie une liste contenant les mots qui apparaissent dans les deux listes (sans doublons).

Exemples :
`common_words(["chat", "chien", "oiseau"], ["chien", "poisson", "chat"])` renvoie `["chat", "chien"]`
`common_words(["un", "deux"], ["trois", "quatre"])` renvoie `[]`

Partie 2 : Exercices plus difficiles

Problème 8: Guess the word (final version)

Écrire une fonction `guessing_game()` qui prend en entrée l'argument positionnel `word` (le mot à deviner), et l'argument positionnel `n_trials` (un entier, le nombre d'essais pour deviner le mot). La fonction renvoie un booléen qui indique si le joueur a gagné ou non.

Problème 9: Somme des chiffres

Écrire une fonction `sum_digits()` qui prend en entrée un entier positif et renvoie la somme de ses chiffres.

Exemples :
`sum_digits(123)` renvoie `6` (car $1 + 2 + 3 = 6$)
`sum_digits(9)` renvoie `9`
`sum_digits(1000)` renvoie `1`

Ici, on peut utiliser la méthode `.isdigit()` pour s'assurer qu'un caractère soit bien numérique !

Problème 10: Mots de longueur croissante

Écrire une fonction `longest_words()` qui prend en entrée une liste de chaînes de caractères et renvoie le nombre de mots qui sont plus longs que leur prédécesseur dans la liste.

Exemples : `longest_words(["my", "beautiful", "catterpillar"])` renverra 2.
`longest_words(["four", "three", "two", "one"])` renverra 0.

Problème 11: Répéter les caractères

Écrire une fonction `repeat_chars()` qui prend en entrée une chaîne de caractères et un entier `n`, et renvoie une nouvelle chaîne où chaque caractère de la chaîne originale est répété `n` fois.

Exemples :
`repeat_chars("hello", 2)` renvoie `"hheellllloo"`
`repeat_chars("ok", 3)` renvoie `"oookkk"`
`repeat_chars("python", 1)` renvoie `"python"`

Problème 12: Premier mot

Écrire une fonction `first_word()` qui prend en entrée une chaîne de caractères `text` et renvoie la sous-chaîne comprenant tous les premiers caractères de `text` jusqu'au premier espace (cela revient à renvoyer le premier mot de la chaîne!). Si la chaîne ne contient aucun espace, elle est retournée telle quelle.

Exemples :
`first_word("it snowed yesterday")` renvoie `"it"`
`first_word("ayooo")` renvoie `"ayooo"`
`first_word("no_snow")` renvoie `"no_snow"`

Problème 13: Compteur de mots

1. Écrire une fonction `count_words()` qui prend en entrée une chaîne de caractères et renvoie le nombre de mots qu'elle contient (on considère que les mots sont séparés par des espaces).
2. Stocker un message dans une variable `text`. Tester la méthode `text.split()`. Que renvoie-t-elle ? Écrire une autre version de la fonction `count_words()`.