Challenges Python

(Discord « Docstring »)

@bucdany

2023

Table des matières

1 Compter le nombre de voyelles **5**
1.1 Énoncé 5
1.2 Solution et explications 6

2 Jeu du « Pierre - Papier - Ciseaux » **7**
2.1 Énoncé 7
2.2 Ma solution 8
2.3 Explication de l'algorithme 8

Avant propos

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Challenge N° 1

Compter le nombre de voyelles

1.1 Énoncé

Ce premier challenge est très simple, il est de niveau « débutant », mais si vous avez plus d'expérience, vous pouvez essayer de trouver de belles astuces pour un code propre, rapide et concis.

Ici, il va nous falloir créer une fonction `nb_voyelles(phrase: str)->int` qui retourne le résultat du nombre total de voyelles dans une phrase passée en paramètre.

Conditions

- Les voyelles sont : `aeiou`, `y` n'est pas pris en compte.
- Les voyelles accentuées ne sont pas prises en compte.
- La phrase passée en paramètre doit être écrite en minuscule.
- Une chaîne vide, passée en paramètre, doit renvoyer `0`.

Exemples

- `nb_voyelles("bonjour, comment allez-vous ?")` doit retourner `9`.
- `nb_voyelles("je vais à paris")` doit retourner `5`.
- `nb_voyelles("docstring")` doit retournet `2`.
- `nb_voyelles("")` doit retourner `0`.

1.2 Solution et explications

Voici donc ma solution [1] :

```python
def nb_voyelles(phrase: str)->int:
    return sum(phrase.count(el) for el in "aeiou")
```

- La phrase doit toujours être en minuscule, donc pas besoin de la méthode `lower()`.
- Ici, on compte chaque voyelle dans la phrase à l'aide de la méthode `count`.
- La fonction `sum()` renvoie ensuite la somme du résultat obtenu.
- Si l'on passe un générateur ou une liste de compréhension dans la fonction `sum()`, la paire de crochets supplémentaire peut-être éliminée [2]. De cette manière :

  `sum([phrase.count(el) for el in "aeiou"])`

  est l'équivalent de :

  `sum(phrase.count(el) for el in "aeiou")`

Voici aussi le code pour mes tests unitaires :

```python
import pytest

@pytest.mark.parametrize("sentence, expected", [
    ("", 0),
    ("docstring", 2),
    ("bonjour comment allez-vous ?", 9),
    ("je vais à paris", 5),
    ("vas-y !", 1),
])
def test_should_return_the_sum(sentence, expected):
    got = nb_voyelles(sentence)
    assert got == expected
```

`@OsKaR31415` a par ailleurs apporté plusieurs solutions pour résoudre ce challenge [3].

1. Fil de discussion de ce challenge: https://discord.com/channels/396825382009044994/1142617945139335189
2. Attention, car cela n'est par contre pas compatible avec la fonction `len()`. Pour plus d'information on se reportera au PEP-289: https://peps.python.org/pep-0289/#the-details.
3. https://discord.com/channels/396825382009044994/1142617945139335189/1145918185168609 98

Challenge N° 2

Jeu du « Pierre - Papier - Ciseaux »

2.1 Énoncé

On va jouer un peu en développant un petit jeu très simple.

Le but du challenge est de développer le célèbre jeu « *pierre - papier - ciseaux* »[1] en essayant de trouver un algorithme astucieux et un code à la fois simple, propre et efficace.

Étapes

1. Générer un choix aléatoire pour votre session de jeu : « pierre », « papier » ou « ciseaux ».
2. Demander au joueur d'écrire son choix entre trois propositions : « pierre », « papier » ou « ciseaux ».
3. Afficher qui a gagné en dévoilant le choix aléatoire du point n°1.

Conditions

- L'affichage, le prompt et la réponse seront affichées par écrit sur la console.
- Le fonctionnement du jeu est simple : la pierre gagne sur les ciseaux, les ciseaux gagnent sur le papier, le papier gagne sur la pierre, deux éléments identiques correspondent à une égalité.
- Toutes les chaînes de caractères, « pierre », « papier » et « ciseaux » doivent toujours être entrées en minuscule, le joueur devra donc écrire correctement ces mots, sinon vous devrez lui demander de redéfinir son choix.
- S'il y a égalité, vous devrez relancer automatiquement votre programme (en regénérant un nouveau choix aléatoire pour la nouvelle session de jeu), jusqu'à ce qu'il y ait un gagnant à la partie.

1. https://fr.wikipedia.org/wiki/Pierre-papier-ciseaux

Exemples

- Le choix aléatoire donne « pierre » et le joueur a choisi « papier » -> `Vous avez gagné ! Le papier enveloppe la pierre`
- Le choix aléatoire donne « ciseaux » et le joueur a choisi « papier » -> `Vous avez perdu ! Les ciseaux coupent le papier`
- Le choix aléatoire donne « pierre » et le joueur a choisi « pierre » -> `Égalité ! Recommencez...`

2.2 Ma solution

```
from random import randint

BDD = {
    "element": ["papier", "pierre", "ciseaux"],
    "gagnant": ["10", "21", "02"],
    "phrase": ["Le papier enveloppe la pierre",
               "Les ciseaux coupent le papier",
               "La pierre casse les ciseaux"]
}

while True:
    joueur = input("pierre, papier ou ciseaux: ")
    if joueur in BDD["element"]:
        choix, joueur = (randint(0, 2),
                         BDD["element"].index(joueur))

        if choix != joueur: break
        print("Ègalité, recommencez...")

    else: print("Faute de frappe !")

print(f'Vous avez {"gagné" if f"{choix}{joueur}" in BDD["g
agnant"] else "perdu"} ! {BDD["phrase"][choix + joueur - 1]
} !')
```

2.3 Explication de l'algorithme

Contexte

- Trois éléments : « pierre », « papier » ou « ciseaux ».
- Un choix aléatoire fait par la machine et un joueur qui entre son choix au clavier.

Il suffit donc de réfléchir à un algorithme sympathique pour présenter le code de manière élégante et éviter bien sûr les répétitions.

Rangement des données

Chaque élément est rangé dans cet ordre particulier dans la liste, afin de les faire correspondre en triade logique.

- papier = index(0)
- pierre = index(1)
- ciseaux = index(2)

Algorithme pour un choix triangulaire

Si on fait l'addition `0+1`, on obtient `1`, alors le jeu se fait entre « papier » et « pierre ». On cherche ensuite la phrase dans `phrase` en faisant juste un calcul grâce à la somme `-1` des deux éléments. Donc en index: `1-0 = 0`, et on trouve donc la chaîne de caractères `"Le papier enveloppe la pierre"`.

Si on fait l'addition `1+2`, on obtient `3`, alors le jeu se fait entre la « pierre » et les « ciseaux ». Donc en index, on obtient `3-1`, soit `2`. On trouve donc la chaîne de caractères `"La pierre casse les ciseaux"`.

De la même façon, si on fait l'addition `2+0`, on obtient `2` et le jeu se fait entre les « ciseaux » et le « papier ». En index cela donne `2-1`, soit `1`, et on tombe sur la chaîne de caractères `"Les ciseaux coupent le papier"`.

The Winner is...

Pour connaître qui gagne, il suffit de convertir en *string* et de joindre les deux caractères d'index du choix et du joueur. Ainsi, `10` dans `gagnant` veut dire que le choix aléatoire donne la « pierre » (index `1`) et que le joueur a saisi le « papier » (index `0`). La « pierre » contre le « papier » fait donc gagner le joueur.

On affiche ainsi `"gagné"` puis la phrase qui suit s'obtient grâce à l'index de la liste `phrase` de la `BDD`, calculée par l'addition des deux index: `1+(0-1)`, ce qui nous donne `0`, ce qui correspond à la chaîne de caractères `"Le papier enveloppe la pierre"`.

De la même façon pour `21` et `02`, cela représente la combinatoire complète des choix gagnants pour le joueur par rapport au choix aléatoire.

Conclusion

On utilise le calcul de la somme `-1` qui renvoie un objet de type `int` et qui pemret l'association des deux chaînes de caractères (*string*) pour connaître le gagnant.