PCSI

Devoir surveillé 1

Exercice 1

Un mot palindrome peut se lire de la même façon de gauche à droite ou de droite à gauche : bob, radar, et non sont des mots palindromes.
De même certains nombres sont eux aussi des palindromes : 33, 121, 345543.
L'objectif de cet exercice est d'obtenir un programme Python permettant de tester si un nombre est un nombre palindrome.
Pour remplir cette tâche, on vous demande de créer le code des trois fonctions ci- dessous sachant que la fonction `est_nbre_palindrome` s'appuiera sur la fonction `est_palindrome` qui elle-même s'appuiera sur la fonction `inverse_chaine`.

1. Ecrire la fonction `inverse_chaine(chaine)` qui inverse l'ordre des caractères d'une chaîne de caractères `chaine` et renvoie la chaîne inversée.

```
>>> inverse_chaine('PCSI')
'ISCP'
```

2. Ecrire la fonction `est_palindrome(chaine)` qui teste si une chaîne de caractères `chaine` est un palindrome. Elle renvoie `True` si c'est le cas et `False` sinon. Cette fonction s'appuie sur la fonction précédente.

```
>>> est_palindrome('PCSI')
False
>>> est_palindrome('POP')
True
```

3. Ecrire la fonction `est_nbre_palindrome(nbre)` qui teste si un nombre `nbre` est un palindrome. Elle renvoie `True` si c'est le cas et `False` sinon. Cette fonction s'appuie sur la fonction précédente.

```
>>> est_nbre_palindrome(214312)
False
>>> est_nbre_palindrome(213312)
True
```

Exercice 2

La probabilité qu'au moins deux étudiants d'une classe de $n \geqslant 2$ étudiants aient leur anniversaire le même jour de l'année est donnéee par la formule (que l'on ne demande pas de démontrer) :

$$p_n = 1 - \prod_{k=1}^{n-1}\left(1 - \frac{k}{365}\right) = 1 - \left(1 - \frac{1}{365}\right)\left(1 - \frac{2}{365}\right)\left(1 - \frac{2}{365}\right)\ldots\left(1 - \frac{n-1}{365}\right).$$

1. Ecrire une fonction `probabilite` qui prend en argument un entier naturel `n` et renvoie p_n.
2. On cherche à partir de quelle valeur de n cette valeur est supérieure à $\frac{1}{2}$. Ecrire un programme permettant d'afficher cette valeur de n cherchée.

Exercice 3

Un entier positif est dit distinct si il est composé de chiffres distincts (différents).

1. Écrire une fonction `occurences(e,ch)` prenant en paramètres un élément `e` et une chaine de caractère `ch` et renvoie le nombre de fois ou l'élément `e` apparait dans `ch`.
2. Écrire une fonction `distinct(n)` prenant en paramètre un entier positif `n` et renvoyant `True` si cet entier est distinct et `False` sinon.
   Cette fonction devra utiliser la fonction de la question précédente.

Exercice 4

On considère des mots à trous : ce sont des chaînes de caractères contenant uniquement des majuscules et des caractères `*`. Par exemple `INFO*MA*IQUE`, `***I***E**` et `*S*` sont des mots à trous. Programmer une fonction `correspond` qui :

- prend en paramètres deux chaînes de caractères `mot` et `mot_a_trous` où `mot_a_trous` est un mot à trous comme indiqué ci-dessus,
- renvoie :
  - ⋆ `True` si on peut obtenir mot en remplaçant convenablement les caractères `'*'` de `mot_a_trous`.
  - ⋆ `False` sinon.

Par exemple :

```
>>> correspond('INFORMATIQUE', 'INFO*MA*IQUE')
True
>>> correspond('AUTOMATIQUE', 'INFO*MA*IQUE')
False
```