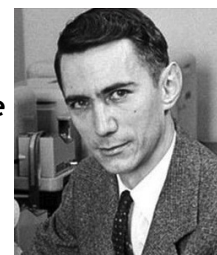


## Partie 1. Robustesse d'un mot de passe

En informatique, la robustesse d'un mot de passe aléatoire est exprimée en terme d'entropie de **Shannon**, mesurée en bits.

Au lieu de mesurer la robustesse par le **nombre de combinaisons de caractères** qu'il faut tester pour trouver le mot de passe avec certitude, on utilise le **logarithme en base 2 de ce nombre**. Cette mesure est appelée l'**entropie du mot de passe**. Un mot de passe avec une entropie de 42 bits calculée de la sorte serait aussi robuste qu'une chaîne de 42 bits choisie au hasard.



Claude Shannon,  
Mathématicien  
(1916 - 2001)

**Question 1** : Pour un mot de passe de 42 bits de robustesse, quel est le nombre de tentatives à réaliser pour le "briser" de façon certaine lors d'une attaque par force brute ?

**Question 2** : Qu'implique l'ajout d'un bit d'entropie à un mot de passe sur le nombre de tentatives lors d'une attaque par force brute ?

Soit  $N$ , le nombre de symboles possibles et  $L$ , le nombre de symboles du mot de passe. L'entropie d'un mot de passe, notée  $H$ , se calcule de la manière suivante :

$$H = L \cdot \log_2(N) = L \cdot \frac{\ln(N)}{\ln(2)}, \text{ où } H \text{ est mesurée en bits.}$$

**Question 3** : Calculer l'entropie pour les valeurs de  $N$  (26, 52, 62, 95) et pour  $L$  valant 6, en complétant le tableau suivant.

$N$ = Nombre de symboles	A-Z (26)	a-zA-Z(52)	a-zA-Z0-9 (62)	a-zA-Z0-9,;:~!... (95)
$L$ = 6 caractères	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

**Question 4** : Soit les valeurs d'entropie pour différentes valeurs de  $N$  et  $L$ .

$N$ = Nombre de symboles	A-Z (26)	a-zA-Z(52)	a-zA-Z0-9 (62)	a-zA-Z0-9,;:~!... (95)
$L$ = 8 caractères	38	46	48	53
$L$ = 10 caractères	47	57	59 <sup>(A)</sup>	66
$L$ = 12 caractères	56 <sup>(B)</sup>	68	71	79
$L$ = 20 caractères	94	114	119	131

Quelle interprétation faites-vous des 2 entropies annotées <sup>(A)</sup> et <sup>(B)</sup> ?

## Partie 2. Calcul de la robustesse d'un mot en passe en Python

Le site [suivant](#) permet de calculer l'entropie de vos mots de passe, selon le principe vu dans l'exercice précédent.

L'objectif de l'exercice est d'implémenter en Python une fonction similaire au site, i.e qui prend en entrée un *mot* de passe, sous la forme d'une chaîne de caractères, et renvoie le *niveau* de robustesse, sous la forme d'un tuple(Entropie, Niveau) selon le principe suivant :


Entropie	Niveau
$H \leq 45$	Faible
$46 < H < 80$	Moyen
$H \geq 80$	Fort <sup>(*)</sup>

(\*) : seuil défini et préconisé par la CNIL dans une recommandation du [14 Octobre 2022](#).

 **Question 1** : Compléter les 2 fonctions suivantes selon les spécifications ci-dessous.

```
def types_symboles(mot):
    """ Indique les types de symboles que comporte le mot de passe donné
    :param mot: (str) Un mot de passe
    :return: (tuple) (H, niveau) où H est l'entropie du mot de passe et niveau, une chaîne de caractère
    :doctests:
        >>> types_symboles('1234567890')
        (False, False, True, False, False)
        >>> types_symboles('p4ssw0rd')
        (True, False, True, False, False)
        >>> types_symboles('M3m3Lord!')
        (True, True, True, True, False)
        >>> types_symboles('1b*nM*t2Passe!')
        (True, True, True, True, True)"""
    pass
```

```
def robustesse(mot):
    """Indique le niveau de robustesse du mot de passe donné
    :param mot: (str) Un mot de passe
    :return: (tuple) (H, niveau) où H est l'entropie du mot de passe et niveau, une chaîne de caractère
    :doctests:
        >>> robustesse('123456')
        (20, 'Faible')
        >>> robustesse('M3m3Lord!')
        (56, 'Moyen')
        >>> robustesse('(NS1)Rule$.')
        (79, 'Moyen')
        >>> robustesse('1b*nM*t2Passe!')
        (92, 'Fort')"""
    pass
```

 **Question 2** : Donner 2 recommandations possibles quant à la taille  $N$  du mot de passe et  $L$  le nombre de symboles à utiliser pour obtenir un mot de passe "Fort".