Année Universitaire : 2022/2023 Master 2 : SII Module : Recherche d'Information	Université des Sciences et de la Technologie Houari Boumediene Faculté d'Informatique Département d'Intelligence Artificielle et Sciences des Données	**TP N°1** Représentation de l'Information : Indexation Partie 1

Indexation :

1. Extraction automatique de termes

Pour l'extraction automatique de termes, nous pouvons utiliser la méthode `split()` comme suit :

```
>>> Texte  = "That D.Z. poster-print costs 120.50DA..."
>>> Termes = Texte.split()
>>> Termes
>>> ['That', 'D.Z.', 'poster-print', 'costs', '120.50DA...']
```

Pour l'extraction automatique de termes, il est recommandé d'utiliser la bibliothèque NLTK (Natural Language ToolKit) avec Python :

```
>>> import nltk
```

Pour l'extraction automatique de termes avec NLTK, il faut définir des expressions régulières à l'aide de la méthode `nltk.RegexpTokenizer()` comme suit :

```
>>> ExpReg = nltk.RegexpTokenizer('\w+') # \w : équivalent à [a-zA-Z0-9_]
>>> Termes = ExpReg.tokenize(Texte)
>>> Termes
>>> ['That', 'D', 'Z', 'poster', 'print', 'costs', '120', '50DA']

>>> ExpReg = nltk.RegexpTokenizer('\w+|(?:[A-Z]\.)+') # ?: nécessaire pour l'utilisation des parenthèses
>>> Termes = ExpReg.tokenize(Texte)
>>> Termes
>>> ['That', 'D', 'Z', 'poster', 'print', 'costs', '120', '50DA']

>>> ExpReg = nltk.RegexpTokenizer('(?:[A-Z]\.)+|\w+')
>>> Termes = ExpReg.tokenize(Texte)
>>> Termes
>>> ['That', 'D.Z.', 'poster', 'print', 'costs', '120', '50DA']

>>> ExpReg = nltk.RegexpTokenizer('(?:[A-Z]\.)+|\w+|\.{3}')
>>> Termes = ExpReg.tokenize(Texte)
>>> Termes
>>> ['That', 'D.Z.', 'poster', 'print', 'costs', '120', '50DA', '...']
```

```
>>> ExpReg = nltk.RegexpTokenizer('(?:[A-Z]\.)+|\d+(?:\.\d+)?DA?|\w+|\.{3}') # \d : équivalent à [0-9]
>>> Termes = ExpReg.tokenize(Texte)
>>> Termes
>>> ['That', 'D.Z.', 'poster', 'print', 'costs', '120.50DA', '...']
```

Pour plus de détails sur l'extraction automatique de termes à l'aide de NLTK, veuillez consulter le livre ***Natural Language Processing with Python***.

2. Suppression des mots-vides

Pour la suppression des mots-vides, il est recommandé d'utiliser la bibliothèque NLTK (Natural Language ToolKit) avec Python :

```
>>> import nltk
```

Télécharger et installer la liste des mots-vides à l'aide de la méthode `nltk.download()` :

```
>>> nltk.download()
```

Suppression des mots-vides :

```
>>> Texte  = "That D.Z. poster-print costs 120.50DA..."
>>> ExpReg = nltk.RegexpTokenizer('(?:[A-Z]\.)+|\d+(?:\.\d+)?DA?|\w+|\.{3}')
>>> Termes = ExpReg.tokenize(Texte)
>>> Termes
>>> ['That', 'D.Z.', 'poster', 'print', 'costs', '120.50DA', '...']
>>> MotsVides = nltk.corpus.stopwords.words('english')
>>> TermesSansMotsVides = [terme for terme in Termes if terme.lower() not in MotsVides]
>>> TermesSansMotsVides
>>> ['D.Z.', 'poster', 'print', 'costs', '120.50DA', '...']
```

3. Normalisation (stemming) des termes extraits

Pour la normalisation des termes extraits, il est recommandé d'utiliser la bibliothèque NLTK (Natural Language ToolKit) avec Python :

```
>>> import nltk
```

Normalisation à l'aide de la méthode `nltk.PorterStemmer()`:

```
>>> Porter = nltk.PorterStemmer()
>>> TermesNormalisation = [Porter.stem(terme) for terme in TermesSansMotsVides]
>>> TermesNormalisation
>>>['d.z.', 'poster', 'print', 'cost', '120.50da', '...']
```

Normalisation à l'aide de la méthode `nltk.LancasterStemmer()`:

```
>>> Lancaster = nltk.LancasterStemmer()
>>> TermesNormalisation = [Lancaster.stem(terme) for terme in TermesSansMotsVides]
>>> TermesNormalisation
>>> ['d.z.', 'post', 'print', 'cost', '120.50da', '...']
```

4. Pondération (stemming) des termes normalisés

5. Création d'index

Création d'un dictionnaire :

```
>>> TermesFrequence = {}
>>> for terme in TermesNormalisation:
        if (terme in TermesFrequence.keys()):
              TermesFrequence[terme] += 1
        else:
              TermesFrequence[terme]  = 1
>>> TermesFrequence
>>> {'d.z.': 1, 'post': 1, 'print': 1, 'cost': 1, '120.50da': 1, '...': 1}
>>> TermesFrequence.keys()
>>> dict_keys(['d.z.', 'post', 'print', 'cost', '120.50da', '...'])
>>> TermesFrequence.items()
>>> dict_items([('d.z.', 1), ('post', 1), ('print', 1), ('cost', 1), ('120.50da', 1), ('...', 1)])
```

```
>>> TermesFrequence = nltk.FreqDist(TermesNormalisation)
>>> TermesFrequence
>>> FreqDist({'d.z.': 1, 'post': 1, 'print': 1, 'cost': 1, '120.50da': 1, '...': 1})
```

Collection

Créez un dossier appelé « TPRI », créez dans ce dossier **N** documents de type texte. Chaque document **i** est nommé « Di ». Ce dossier sera notre collection de travail. Réalisez l'exercice suivant avec le langage Python.

Exercice 1 :

Ecrire un programme python qui permet la création des structures suivantes : (en ignorant les mots vides et les ponctuations (. , ! ? ` … etc).

1. Créer pour chaque document une structure, contenant les termes (les mots) du document avec leurs fréquences. Dans cette structure la clé sera le terme, la valeur sera sa fréquence d'apparition dans le document.

2. Créer une seule structure, contenant tous les termes des documents avec leurs fréquences (chaque terme est associé avec sa fréquence d'apparition dans son document). Dans cette structure la clé sera une paire (terme, N° du document), la valeur sera la fréquence d'apparition de ce terme dans ce document. Si le terme n'existe pas dans ce document, sa valeur sera 0. Nous appelons cette structure le Fichier Inverse de fréquences de cette collection.