

**Bloc 1 Représentation des données et programmation Langages et programmation** — Traitement de données en tables

***Exercice*** *:* ***Recherche séquentielle dans un annuaire***

1 Avec LibreOffice, créer un annuaire dans une nouvelle feuille de tableur en saisissant à partir de la ligne 1 : une série de prénoms dans la colonne A associée à une série de numéros de téléphones dans la colonne B. Enregistrer ce fichier au format csv sous le nom annuaire.csv en choisissant ’, comme séparateur de champ et rien comme séparateur de texte. Modifier le fichier pour qu’en première ligne on trouve le prénom ‘Alphonse’ et en dernière ligne ‘Zied’

1. Compléter la fonction ext rai re fi chi e r) qui retourne une liste (ou tableau) de prénoms et une liste (ou tableau) de numéros extraites d’un fichier de type annuaire . csv.

Veuillez consulter le code ci-dessous ou task02.py

1. Rajouter une fonction recherche\_sequentielle(clef,liste1,liste2) au script précédent. Cette fonction prend pour paramètre une clef qu’elle recherche dans liste1.

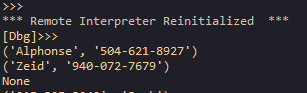
Elle retourne le couple (celf, list2[i]) si clef apparaît en position i dans liste1.

Si liste1 and liste2 sont respectivement la liste de prénoms et la liste de numéros extraites du fichier annuaire.csv, on peut ainsi faire une recherche dans l’annuaire par prénoms (on suppose que tous les prénoms sont distincts quitte à les numéroter comme ’Fred 1’, ’Fred2’). Tester cette fonction pour faire une recherche par prénom dans le fichier annuaie.csv.

Answer :

Veuillez consulter le code ci-dessous ou task02.py

Output :



1. Comment peut-on utiliser la fonction recherche\_sequentielle(clef,liste1,liste2) pour faire une recherche inversée (par numéro) dans le fichier annuaire.csv.

Answer:

Pour utiliser la fonction search\_sequential pour faire l'inverse, nous devons passer le nombre comme clé, la liste des nombres au paramètre «liste1» et la liste des noms au paramètre «liste2» respectivement.

Tel que,

Recherche\_sequentielle(‘215-325-3042’, n, p)

#while n is list of numbers and p is list of names

Output :

temp.bmp

1. Tester la fonction recherche\_sequentielle pour rechercher les clefs ’Alphonse’ puis ’Zied’ dans le fichier annuaire annuaire.csv. Combien de comparaisons avec les éléments du tableau p renom sont effectuées dans chaque cas ?

Answer :

Dans le cas de «alphonse», une seule comparaison et pour «zied» n comparaisons auront lieu, où n est le nombre total de prénoms dans la liste, selon l’algorithme de recherche séquentielle.

1. Si l’annuaire comporte n prénoms, combien cet algorithme effectue-t-il de comparaisons si le prénom cherché n’est pas dans l’annuaire ? est en première position (meilleur des cas) ? est en dernière position(pire des cas)?

Combien de comparaisons sont effectuées en moyenne si on choisit un prénom au hasard parmi les prénoms dans l’annuaire?

Answer :

Si le dictionnaire a n prénoms, et dans le cas où first\_name n'est pas présent dans le répertoire, il faut effectuer n comparaisons.

Dans le cas où le prénom se trouve en première position (meilleur des cas), une seule comparaison sera effectuée.

Si le prénom se trouve en dernière position (pire des cas), n nombre de comparaisons doivent être effectuées.

Si nous choisissons le prénom au hasard (cas moyen). En moyenne, nous trouverons l'article à mi-chemin de la liste. Nous pouvons dire que nous devons effectuer environ n / 2 comparaisons.

Si nous ellobratons cela en utilisant une table

|  |  |  |  |
| --- | --- | --- | --- |
| **Case** | Best Case  (prénom en première position) | Wrost Case  (prénom à la dernière position) | Average Case  (prénom d'une position aléatoire) |
| Le prénom est présent | **1** | **n** | **n/2** |
| Le prénom n'est pas présent | **n** | **n** | **n** |

1. Estimer le temps nécessaire pour une recherche séquentielle dans un annuaire de 30 000 prénoms : en moyenne et dans le pire des cas. Chaque tour de boucle de l’algorithme s’effectuant à coût constant en terme de comparaisons et d’affectations on peut mesurer sa complexité (ou son coût) par le nombre d’itérations de la boucle Tant Que. Expliquer pourquoi on peut qualifier la complexité (ou coût) moyenne de cet algorithme de linéaire.

Answer :

Le temps requis pour une recherche séquentielle dans un dictionnaire de 30 000 prénoms en moyenne casse est O (n / 2) soit environ 15 000 comparaisons en moyenne. Et pour le plus mauvais cas, O (n) ou environ 30 000 comparaisons.

La complaxité temporelle de l'algorithme de recherche séquentielle est toujours linéaire car, comme dans le meilleur des cas, elle est de 1 et le plus mauvais cas est n. tandis que pour le cas moyen est n / 2. On peut aussi dire que la forme généralisée de complexité temporelle pour l'algorithme de recherche séquentielle est T (n) = 1 + 2 + 3 +…. + n / 2 + …… .. + (n-2) + (n-1) + n ou O (n). C'est simplement linéaire, car le nombre de comparaisons augmente assez constamment.

def extraire(fichier):

"""Extrait d’un fichier texte annuaire avec du type prenom, tel la liste des prenoms et celle des numéros de telephone""" f = open(fichier,’r’)

prenom,tel = [],[] #initialisation des listes de prenoms et de numeros for ligne in f :

p,n =ligne.rstrip().split(’,’) #write your code here

prenom.append(p)

tel.append(n)

return prenom, tel

def recherche sequentielle(clef,liste1,liste2):

"""recherche de séquentielle si clef appartient à la listel de l’annuaire constitué des deux listes listel et liste2.

Si clef est trouvée dans listel à la position i, retourne le couple clef,liste2[i], sinon retourne None"""

#write your code here

i = 0

m = len(liste1)

while i < m:

if liste1[i] == clef:

return (clef, liste2[i])

i += 1

else:

return None

l1, l2 = extraire('annuaire.csv')