Block 1

Data representation and programming languages ​​and programming - Data processing in tables

Exercise: Sequential search in a directory

1. With LibreOffice, create a directory in a new spreadsheet by entering from row 1: a series of first names in column A associated with a series of telephone numbers in column B. Save this file in csv format under the name directory.csv by choosing ', as the field separator and nothing as the text separator. Modify the file so that in the first line we find the first name "Alphonse" and in the last line "Zied"

2. Complete the extraire(fichier)function which returns a list (or array) of first names and a list (or array) of numbers extracted from a directory type file. Directory.csv.

Please see the code below or task02.py

3. Add a search\_sequential function (key, list1, list2) to the previous script. This function takes as parameter a key that it searches for in list1.

It returns the pair (celf, list2[i]) if key appears in position i in list1.

If list1 and list2 are respectively the list of first names and the list of numbers extracted from the directory.csv file, we can thus search the directory by first names (we assume that all the first names are distinct even if it means numbering them like 'Fred 1 ',' Fred2 '). Test this function to search by first name in the annuaie.csv file.

Answer :

Please see the code below or task02.py

4. How can we use the search\_sequential function (key, list1, list2) to do a reverse search (by number) in the directory.csv file.

Answer :

To use the search\_sequential function to do reverse, we should pass number as key, list of numbers to parameter ‘liste1’ and list of names to parameter ‘liste2’ resectively.

Such that,

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

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

Output:

temp.bmp

5. Test the search\_sequential function to find the "Alphonse" and then "Zied" keys in the directory directory.csv file. How many comparisons with the elements of the renowned p array are made in each case?

Answer :

In case of ‘alphonse’ only **1** comparison and for ‘zied’ **n** comparisons will take place, where n is the total number of first names in the list, according to sequential search algorithem.

6. If the directory has n first names, how many comparisons does this algorithm make if the first name sought is not in the directory? is in first position (best case)? is in last position (worst case)?

How many comparisons are made on average if we choose a first name at random among the first names in the directory?

Answer:

If the dictionary has n first names, and in case of first\_name is not present it the directory, it is needed to perform **n** comparisions.

In case of first name lies in the first position(best case), only **1** comparison will performed.

In case of first name lies in the last position(worst case), **n** number of comparisons should performed.

If we choose the first name at random(Average case). On average we will find the item about half way to the list. We can say that we have to perform aproximatly **n/2** comparisons.

If we ellobrate this using a table :

|  |  |  |  |
| --- | --- | --- | --- |
| **Case** | Best Case  (first name at first position) | Wrost Case  (first name at last position) | Average Case  (first name from a random position) |
| First name is present | **1** | **n** | **n/2** |
| First name is not present | **n** | **n** | **n** |

1. Estimate the time required for a sequential search in a directory of 30,000 first names: on average and in the worst case. Each round of the algorithm loop being performed at constant cost in terms of comparisons and assignments, its complexity (or cost) can be measured by the number of iterations of the while loop. Explain, why we can qualify the average complexity (or cost) of this algorithm as linear.

Answer:

The time reqiured for sequential search in a dictionary of 30,000 first names on average case is O(n/2) or about 15,000 comparaisons on average. And for wrost case is O(n) or about 30,000 comparisons.

The time complaxity for sequential search algorithm is always linear because, as for best case it is 1 and wrost case is n. while, for average case is n/2. We can also say that the generalized form of time complexity for the sequential search algorithem is T(n) = 1 + 2 + 3 + …. + n/2 + ……..+ (n-2)+(n-1)+n or O(n). This is simply linear, because the number of comparisons increasesing quite constantly.

