**Rappel :**

Les dictionnaires, au même titre que les *listes* et les *tuples* sont des types de données *construits*.

Mais, contrairement à ces deux derniers types, les éléments enregistrés dans un dictionnaire ne sont pas accessibles par un indice mais par une clé.

Exemples :

L=[1,2,3,4] # création d’une liste

L[1] # accès à la valeur "2" avec l'indice **1**

T=(1,2,3,4) # création d’un tuple

T[2] # accès à la valeur "3" avec l'indice **2**

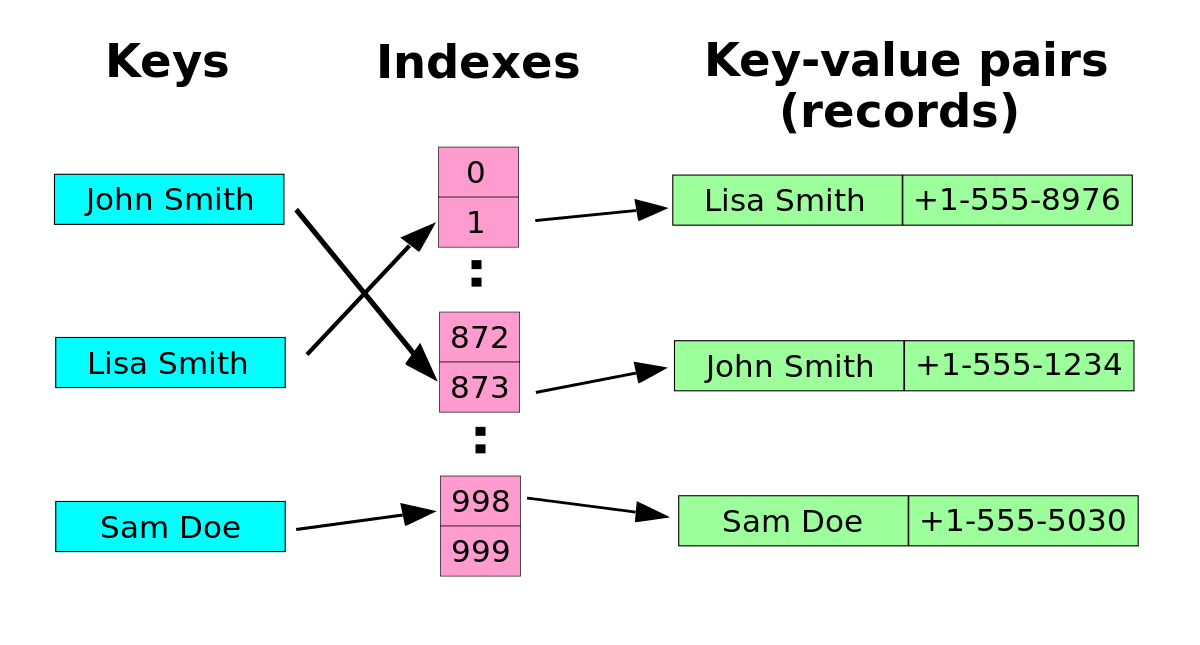
D={"un" : 1, "deux" : 2, "trois" : 3, "quatre" : 4} # création d’un dictionnaire

D["quatre"] # accès à la valeur "4" avec la clé **"quatre"**

**Objectif :**

Nous allons voir de façon simplifiée la façon dont Python accède dans un dictionnaire à une valeur à partir de la clef en temps constant O(1).

**En fait Python pour obtenir la valeur liée à la clef d’un dictionnaire, accède au rang d’une liste (table de hachage) correspondant à la clef de cette valeur, mais quel rang ? C’est ce que nous allons découvrir.**



Rang de liste

Contenu de la liste

*Liste représentant un dictionnaire*

Passage d'une clé à un indice (rang de liste)

La transformation d'une clé en un rang de liste (indice) se fait au moyen d'une fonction de hachage.

La **Clé - Valeur** sera enregistrée dans une table de hachage (liste python) à ce rang (indice).

On considère, dans un premier temps, que la clé est une valeur numérique.

* *Les clés seront stockées dans une table de hachage de taille m=11 (liste de taille 11).*

*On prend généralement une liste de taille environ double par rapport à ce que l’on souhaite stocker*

* *La fonction de hachage est la fonction h(x)=x mod 11 (modulo)*

1. Rappeler ce que fait l'opérateur modulo.

*5 modulo 5 =0 (en python 5%5=0)*

1. Calcul du rang de la liste correspondant à la clef. Compléter le tableau :

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Clé : x | 10 | 22 | 31 | 48 | 28 |
| Rang de la liste : h(x) | 10 | 0 | 9 | 4 | 6 |

1. Compléter la **table de hachage (Dictionnaire)** en plaçant les clés-valeurs suivantes :

10-Jess ; 22-Brad ; 31-Noé ; 48-Joe ; 28-Bob

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 22, Brad |  |  |  | 48, Joe |  | 28, Bob |  |  | 31, Noé | 10, Jesse |

Les collisions

1. On cherche maintenant à placer la clé-valeur : 15-Théo. Calculer la valeur de l’indice (rang) obtenue par la fonction de hachage et placer la clé-valeur dans la table. Quel est le problème ?

***Nous venons de créer une collision !*** *Il existe plusieurs méthodes pour traiter ce problème.*

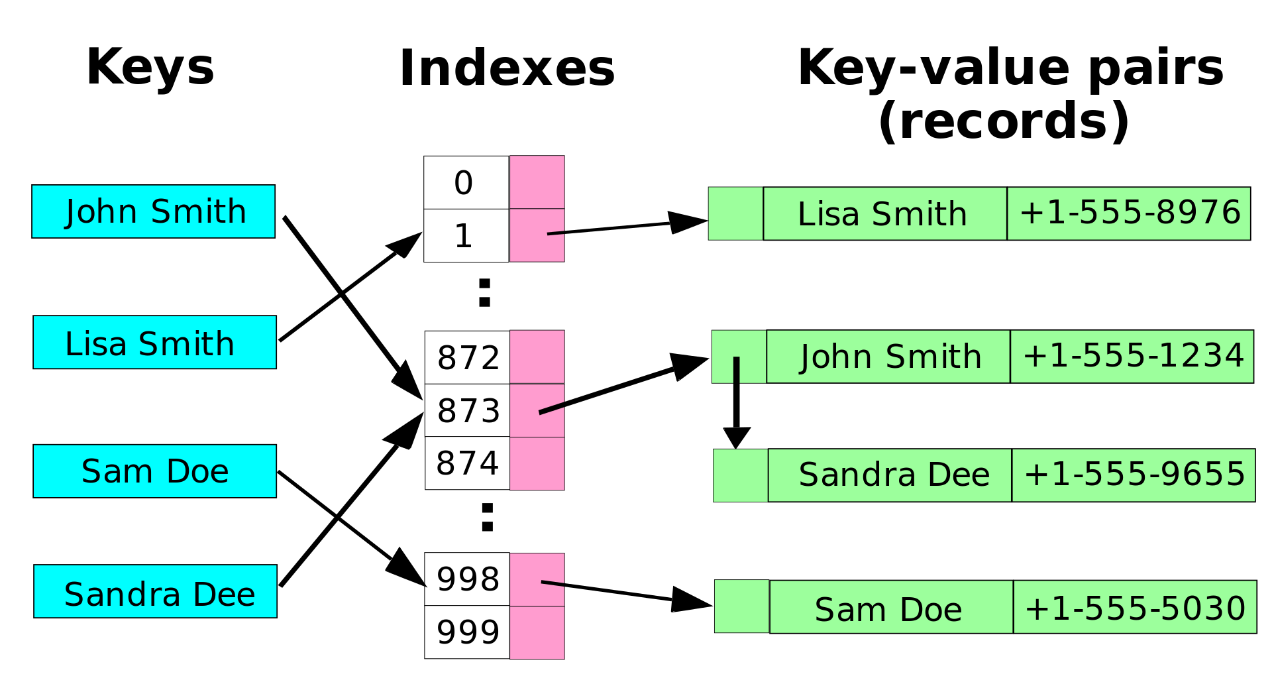
*α×BIG BANG×α*

1. Proposer une méthode pour gérer cette collision. Votre méthode permet-elle de gérer un grand nombre de collisions ?

Incrémenter de 1 la place voulu par Théo jusqu’à qu’il n’y ai plus de collision

Résolution des collisions par chaînage :

Cette méthode est la plus simple. Chaque cellule de la table de hachage est une liste chaînée qui peut accueillir plusieurs éléments en cas de collision.



1. Avec la même fonction de hachage, ajouter les clés-valeurs :

15-Théo, 37-Yoda, 59-Tim, 83-Léo, 88-Léia

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 22, Brad |  |  |  | 48, Joe |  | 28, Bob |  |  | 31, Noé | 10, Jesse |
| 88-Léia |  |  |  | 15-Théo |  | 83-Léo |  |  |  |  |
|  |  |  |  | 59-Tim |  |  |  |  |  |  |
|  |  |  |  | 37-Yoda |  |  |  |  |  |  |

1. Quel défaut comporte cette méthode ? *(Indice : mémoire)*

Ca multiplie les listes et surcharge trop la mémoire

Résolution des collisions par adressage ouvert (sondage) :

Lors d'une collision, l'élément est placé dans une cellule voisine libre. On doit donc "sonder" (inspecter) chaque cellule de la table de hachage jusqu'à en trouver une qui convienne.

Il existe plusieurs méthodes pour parcourir la table de hachage à la recherche d'une cellule vide :

* **Le sondage linéaire** : l'intervalle entre les alvéoles est fixe, souvent 1
* **Le sondage quadratique** : l'intervalle entre les alvéoles augmente linéairement
* **Le double hachage** : l'indice de l'alvéole est donné par une deuxième fonction de hachage



Nous allons utiliser, dans cet exemple, la méthode du double hachage. Elle consiste à utiliser une deuxième fonction de hachage afin de trouver une place libre.

*La première fonction est toujours h(x) = x mod m = x mod 11*

*La deuxième que nous allons utiliser est g(x) = 1 + x mod (m-1) = 1 + x mod (11-1) = 1 + x mod 10*

*La position finale (rang) est alors obtenue par la formule : x 🡪 [ h(x) + i x g(x) ] mod m*

*x 🡪 [ h(x) + i x g(x) ] mod 11*

*On teste pour toutes les valeurs de i entre 0 et m-1*

1. Quelle est la formule obtenue lorsque i=0 ?

On retrouve x 🡪 x mod 11 comme auparavant

**Remplissage de la table de hachage pour les premiers termes :**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 22-Brad |  | 88-Léo | 59-Tim | 48-Joe | 15-Théo | 28-Bob | 37-Yoda | 83-Léo | 31-Noé | 10-Jess |

On peut déjà placer les premiers éléments du travail 2car il n'y avait pas de collision.

La première collision se produisait pour 15, on sait donc que h(15) (pour i=0) est déjà occupée.

Calcul de la nouvelle position de 15 :

* On essaye avec i= 1 :

Ce qui donne :

* Il y a à nouveau collision, on essaye donc avec i=2 :

On peut alors placer la clé 15 dans la cellule 5.

1. Placer dans la table de hachage ci-dessus le reste des clés : 15, 37, 59, 83, 88