UNIVERSITÉ TOULOUSE III PAUL SABATIER

Université de Toulouse

Faculté des Sciences et d'Ingénierie

Structures de données

CTD 10 : SDD Partie 2

Algorithmique

Semestre 2 - 2019-2020

Le dictionnaire

- Introduction
- Abstraction
- Implémentation
- Intérêt
- Exercices

Les ensembles

- Introduction
- Abstraction
- Implémentation en Python
- Exercices

Le dictionnaire - Introduction

Prenons deux exemples

- Un répertoire téléphonique
  - Etablit l'association entre un contact et les informations qui le décrivent

Mes contacts	
0561556611	Nom_0561556611
	Prénom_0561556611
	Adresse_0561556611
	Email_0561556611
0561556613	Nom_0561556613
	Prénom_0561556613
	Adresse_0561556613
	Email_0561556613
....	
15	Nom_15
	Prénom_15
	Adresse_15
	Email_15

- Un répertoire (dossier,directory)
  - Etablit l'association entre un nom de fichier et les informations de ce fichier

Motivation

Appelé aussi tableau associatif ou table d'associations

- Gère des associations (clé, valeur)
- Permet de retrouver une valeur à partir de la clé qui la référence
- Peu importe l'ordre d'ajout des associations dans le dictionnaire

Le dictionnaire - Introduction

Hypothèse :

Une clé n'apparaît qu'au plus une fois dans un dictionnaire

Objectif

- Ajout d'une association, de manière à pouvoir la retrouver de manière efficace plus tard
- Suppression d'une association du dictionnaire
- Retrouver la valeur associée à une clé
- Obtenir l'ensemble des clés actuellement référencées dans un dictionnaire

$\Rightarrow$ **Gérer et organiser " intelligemment " les données**
$\Rightarrow$ Recherche d'une valeur en O(log(n)) au lieu de O(n) pour une liste

Abstraction

Opérations définies :

- Opérations de construction
  - Création d'un dictionnaire vide
  - Initialisation d'un dictionnaire à partir de plusieurs associations clé / valeur
- Opérations d'accès
  - Savoir si une clé apparaît dans un dictionnaire
  - Accéder à la valeur associée à une clé donnée
  - Savoir si un dictionnaire est vide
  - Parcourir les clés du dictionnaire
- Opérations de modification
  - Ajouter une association (clé,valeur)
  - Supprimer une association (clé,valeur)

Implémentation

- Peut être implémenté en utilisant les structures de données précédemment présentées
- Par exemple un tableau de tuples, représentant chacun une association/paire (clé, valeur)
- Par la suite, nous allons utiliser les dictionnaires Python

Le dictionnaire - Implémentation

Opérations de construction :

- Création d'un dictionnaire vide :

```
1  mon_dict = {}
```

- Initialisation d'un dictionnaire :

```
1  mon_dict = {cle_1: val_1, ..., cle_N: val_N}
```

Opérations d'accès

- Savoir si une clé apparaît dans un dictionnaire :

```
1  if cle in mon_dict:
2    #cas ou la cle apparait
3  else:
4    #cas ou la cle n'apparait pas
5
```

Opérations d'accès

- Accéder à la valeur associée à une clé donnée :

```
valeur= mon_dict[cle]
```

- Savoir si un dictionnaire est vide

```
if mon_dict == {}: #ou if len(mon_dict) == 0:
  #cas ou le dictionnaire est vide
else:
  #cas ou le dictionnaire n'est pas vide
```

- Parcourir les clés du dictionnaire

```
for key in mon_dict:
  <instructions>
```

Fonctions de modification

- Ajouter une association (clé,valeur) (écrase la valeur précédente si la clé existe déjà)

```
1 mon_dict[cle]=valeur
```

- Supprimer une association (clé,valeur)

```
1 del mon_dict[cle]
```

Exercice 1 : Jeu de cartes

On se propose d'évaluer le nombre de points que valent les cartes détenues dans une main de jeu. A chaque carte, désignée par une chaîne de caractères ("as ", "roi", "valet"...), est associée une valeur entière. Ecrire les fonctions suivantes :

- `initialiser_valeurs_cartes` qui initialise et retourne un dictionnaire d' associations de carte à leur valeur numérique
- `saisir_main` qui saisit les cartes composant une main et les retourne dans une liste Python. Les cartes sont des couples (valeur,couleur) telles que ("valet","pique") ... . Le nombre de cartes sera donné en paramètre, ainsi que le dictionnaire des associations carte/valeur. Il ne peut y avoir qu'un seul couple d'une certaine valeur et d'une certaine couleur dans la main du joueur.

Exercices

Exercice 1 : Jeu de cartes

- `calculer_valeur_main` qui prend en paramètre une main et les associations carte/valeur et qui retourne la valeur en points de la main donnée.
- `test_calculer_valeur_main` qui teste l'évaluation d'une main de jeu

Exercice 2 : Traduction de fichier

Pour cet exercice, nous allons manipuler un fichier de texte. Pour ceci nous allons utiliser plusieurs fonctions Python dont

```
1 with open('f_anglais.txt') as f : #permet d'ouvrir le
      fichier f_anglais.txt en lecture
2   for line in f: #parcourt chaque ligne du fichier
3     <instructions>
```

Le code suivant permet de retourner dans une liste les mots contenu dans la variable `une_ligne`

```
1 une_ligne.split()
```

Exercice 2 : Traduction de fichier

- Ecrire le programme Python qui en affiche la traduction mot à mot en français du texte contenus dans f_anglais.txt grâce au dictionnaire suivant

```
1 anglais_francais = {'are':'sont', 'boiled':'bouillie',
      'is':'est',
2 'flesh':'viande', 'frenchies':'francais',
      'little':'petits', 'my':'mon', 'rich':'riche',
      'tailor':'tailleur', 'toomuch':'trop', 'the':'les',
      'this':'cette' }
```

Les ensembles - Introduction

Exemples

- Ensemble des étudiants qui suivent les cours C1 et C2,
- Ensemble des nombres premiers compris entre v1 et v2,
- Ensemble des événements attendus par un processus
- Ensemble des signaux bloqués par un processus

Description

- Séquence finie d'objets, dans laquelle un objet ne se retrouve qu'une fois
- Les éléments sont tous de même type
- Seul compte le fait qu'une valeur soit présente ou non dans l'ensemble
  - Peu importe sa position dans l'ensemble
  - Peu importe l'ordre d'ajout des éléments dans l'ensemble

Description

- Contenu d'un ensemble
  - Évolue dynamiquement pendant l'exécution par ajout/suppression d'éléments
  - Cardinal C d'un ensemble (dimension)
    - Nombre d'éléments actuellement contenus dans l'ensemble
    - Ensemble vide $\Leftrightarrow$ C == 0
- Opérations entre ensembles
  - Intersection
  - Union
  - Différence
  - ...

Abstraction

Opérations définies :

- Opérations de construction
  - Initialisation de l'ensemble à vide
  - Initialisation avec des valeurs
- Opérations d'accès
  - Connaître le cardinal de l'ensemble
  - Savoir si une valeur est présente ou non dans l'ensemble
  - Savoir si l'ensemble est vide
  - Parcourir les valeurs d'un ensemble
- Opérations de modification
  - Ajouter une valeur dans l'ensemble
  - Retirer une valeur

Opérations de construction :

- Initialiser un ensemble à vide ou avec des valeurs :

```
1 mon_ens = set()
2 mon_ens = {v1, v2, ..., vn}
3 mon_ens = set(liste_valeurs) #ou liste_valeurs est une
      sequence, liste ou tuple ou chaine
4 mon_ens = set(autre_ens) #mon_ens est une copie de autre_ens
```

Opérations d'accès :

- Connaître le cardinal de l'ensemble :

```
1 len(mon_ens)
```

- Savoir si l'ensemble est non vide :

```
1 if mon_ens: #ou if (len(mon_ens) != 0):
2   # cas non vide
3 else:
4   # cas vide
```

Opérations d'accès :

- Savoir si une valeur appartient à l'ensemble ou non :

```
1 if valeur in mon_ens : #cas ou la valeur appartient a
    l'ensemble
```

```
1 if valeur not in mon_ens : #cas ou la valeur n'appartient
    pas a l'ensemble
```

- Parcourir les valeurs d'un ensemble

```
1 for elem in mon_ens:
2   <instructions>
3
```

Opérations de modification :

- Ajouter une valeur dans l'ensemble :

```
1 mon_ens.add(valeur)
```

- Retirer une valeur de l'ensemble :

```
1 mon_ens.remove(valeur)#erreur si valeur n'y est pas
2 mon_ens.discard(valeur)#pas d'erreur si valeur n'y est pas
3 une_valeur = mon_ens.pop() # retire un element quelconque,
    dont la valeur est retournee par pop
4 mon_ens.clear() #retire tous les elements (l'ensemble
    redevient vide)
```

Exercices

- Ecrire une fonction qui, étant donné un fichier texte,retourne l'ensemble des mots utilisés dans ce fichier.
- Ecrire les fonctions suivantes sans utiliser les fonctions décrites par Python :
  - Intersection de deux ensemble ens1 et ens2
  - Union de deux ensemble ens1 et ens2
  - Différence de deux ensembles ens1 et ens2

Exercice 2

Calculer la moyenne des notes des étudiants à une épreuve, par année d'obtention du bac. Le système d'information est organisé par année de bac : un dictionnaire associe à chaque année de bac une liste Python des étudiants ayant obtenu le bac cette année là. On suppose fournies les fonctions suivantes :

- `initialiser_etudiant(num_etudiant)` qui, étant donné un numéro d'étudiant, retourne un dictionnaire qui décrit cet étudiant avec les clés nom, prénom, date du bac, notes dans ses 5 modules de S1).
- `annee_bac(un_etudiant)` qui, étant donnée une description d'étudiant, retourne l'entier correspondant à l'année d'obtention du bac.

Exercices

- `note_etudiant(un_etudiant, num_epreuve)` qui, étant donnés une description d'étudiant et un numéro de module, retourne la note obtenue par cet étudiant à ce module.
- `afficher_etudiant(un_etudiant)` qui, étant donnée une description d'étudiant, affiche à l'écran les informations relatives à cet étudiant (nom, prénom, notes, etc.)
- `afficher_liste_etudiants(liste_etudiants)` qui, étant donnée une liste Python contenant des descriptions d'étudiant, affiche les informations relatives à l'ensemble de ces étudiants

Exercice 2

Ecrire les fonctions suivantes :

- `creer_et_initialiser_dictionnaire` qui enregistre dans le système d'informations un certain nombre d'étudiants, en gérant les associations année du bac liste d'étudiants qui en découlent. Le nombre d'étudiants sera généré de manière aléatoire dans cette fonction.
- `afficher_dico` qui affiche le contenu du dictionnaire qui représente le système d'informations.
- `calculer_moyenne` qui, pour une année de bac `annee_du_bac` et pour une épreuve `num_epreuve` données, calcule la moyenne à cette épreuve pour les étudiants ayant obtenu le bac à cette année là.

Exercices

- `main` donnée qui :
  1. Initialise le dictionnaire représentant le système d'information.
  2. Affiche le contenu de ce dictionnaire.
  3. Calcule et affiche la moyenne pour une épreuve donnée pour les étudiants ayant obtenu le bac une année donnée. Le numéro de l'épreuve et l'année du bac seront saisies au clavier.

Fonctions fournies

```
1  import numpy as np
2  from random import randint
3  nb_epreuves = 5
4  def initialiser_etudiant(num_etudiant):
5    ses_notes = []
6    for i in range(nb_epreuves):
7      note=float(input())
8      ses_notes.append(note)
9    nom=input()
10   prenom=input()
11   annee_bac=int(input())
12   un_etudiant = { "nom" : nom ,"prenom":prenom,
       "annee_bac":annee_bac, "notes":ses_notes,
       "numero":num_etudiant}
13   return un_etudiant
```

Fonctions fournies

```
1 def annee_bac(un_etudiant):
2   return un_etudiant["annee_bac"]
```

```
1 def note_etudiant(un_etudiant, num_epreuve):
2   return un_etudiant["notes"][num_epreuve]
```

```
1  def afficher_etudiant(un_etudiant):
2    print("\tNom:", un_etudiant["nom"], "␣Prenom:␣",
       un_etudiant["prenom"], "␣Numero:␣",
3    un_etudiant["numero"])
4    date_bac = un_etudiant["annee_bac"]
5    print("\t\tBac:␣", date_bac)
6    ses_notes = un_etudiant["notes"]
7    nb_notes = len(ses_notes)
8    print('\t\t', end='', sep='')
9    for num_note in range(nb_notes):
10     print(ses_notes[num_note], '␣', end='')
11   print('')
```

Exercices

Fonctions fournies

```
1 def afficher_liste_etudiants(liste_etudiants):
2   for un_etudiant in liste_etudiants:
3     afficher_etudiant(un_etudiant)
```