Liste e dizionari

Rachele Sprugnoli

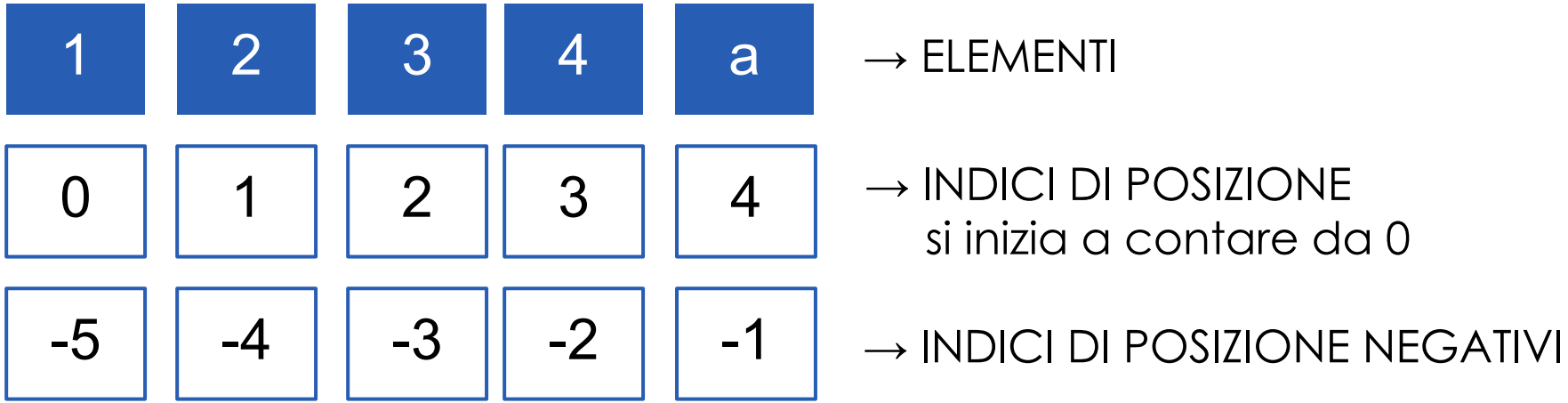
rachele.sprugnoli@unipr.it

UNIVERSITÀ
DI PARMA

Cosa sono le liste

- Sequenze di elementi ciascuno dei quali è associato a una posizione (indice) che è un numero intero: gli elementi vengono memorizzati nello stesso ordine in cui vengono forniti

```
list =[1,2,3,4,a]
```



Liste: usi

N.B. Doppio uso delle parentesi quadre!

- Creazione:
  - `list = []` → lista vuota
  - `list = [elemento,elemento,elemento]` → lista con 3 elementi
  - `len(list)` → restituisce il numero di elementi di list
- Accesso agli elementi:
  - `list[indice]` → recupera l'elemento alla posizione dell'indice
- Modifica di un elemento:
  - `list[indice]= nuovo_elemento` → assegna un nuovo elemento alla posizione indicata dall'indice (ATTENZIONE: sovrascrive!)

Liste: usi

- Aggiungere un elemento:
  - `list.append(elemento)` → aggiunge l'elemento alla fine lista
- Inserire un elemento:
  - `list.insert(indice,elemento)` → aggiunge l'elemento nella posizione specificata, gli elementi successivi vengono spostati di una posizione in avanti
- Ordinare gli elementi:
  - `list.sort()` → ordina gli elementi in senso crescente
  - `list.sort(reverse = True)` → ordina in senso decrescente
  - `max(list)`
  - `min(list)`

Liste: usi

- Eliminare un elemento:
  - `list.pop()` → elimina l'ultimo elemento
  - `list.pop(indice)` → elimina l'elemento nella posizione specificata, gli elementi successivi vengono spostati di una posizione all'indietro
  - `list.remove(elemento)` → elimina un elemento in base al suo valore

Liste: usi

- Cercare un elemento:
  - `elemento in list` → controllare la presenza di un elemento e, se presente, restituisce True, altrimenti False
  - `list.count(element)` → restituisce il numero delle volte in cui è presente un elemento
  - `list.index(elemento)` → restituisce la posizione dell'elemento alla sua prima occorrenza

    N.B. Come fare se lo stesso elemento appare più volte e si vuole recuperare ogni posizione delle sue varie occorrenze?

Enumerate

- Funzione predefinita che attraversa gli elementi di una sequenza e i loro indici, https://www.geeksforgeeks.org/enumerate-in-python/

```
for indice,elemento in enumerate(list):
    print(indice, elemento)
```

OUTPUT:

```
indice    elemento
indice    elemento
indice    elemento
…
```

Enumerate

- Funzione predefinita che attraversa gli elementi di una sequenza e i loro indici, https://www.geeksforgeeks.org/enumerate-in-python/

```
list = [3,1,5,5,5]
for indice,elemento in enumerate(list):
    if elemento == 5:
        print(f"{indice}")
```

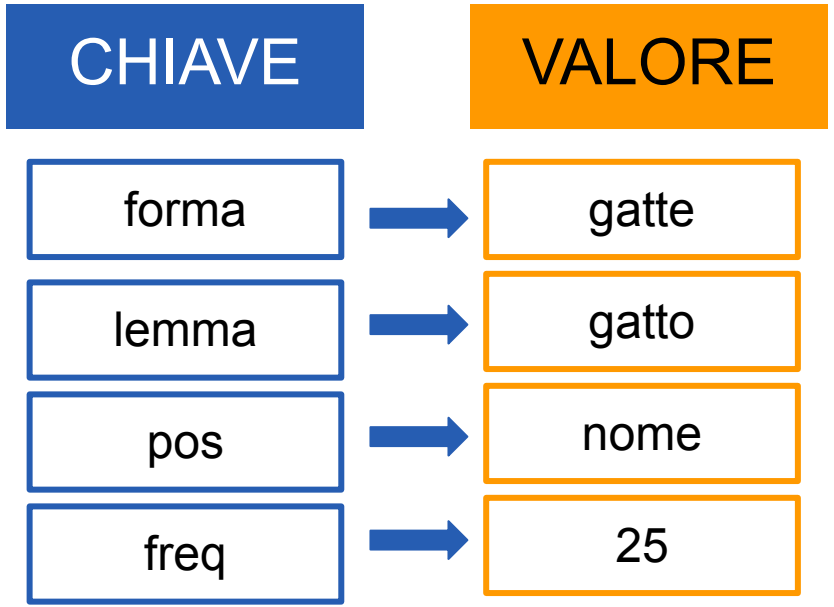
OUTPUT:

```
2
3
4
```

Cosa sono i dizionari

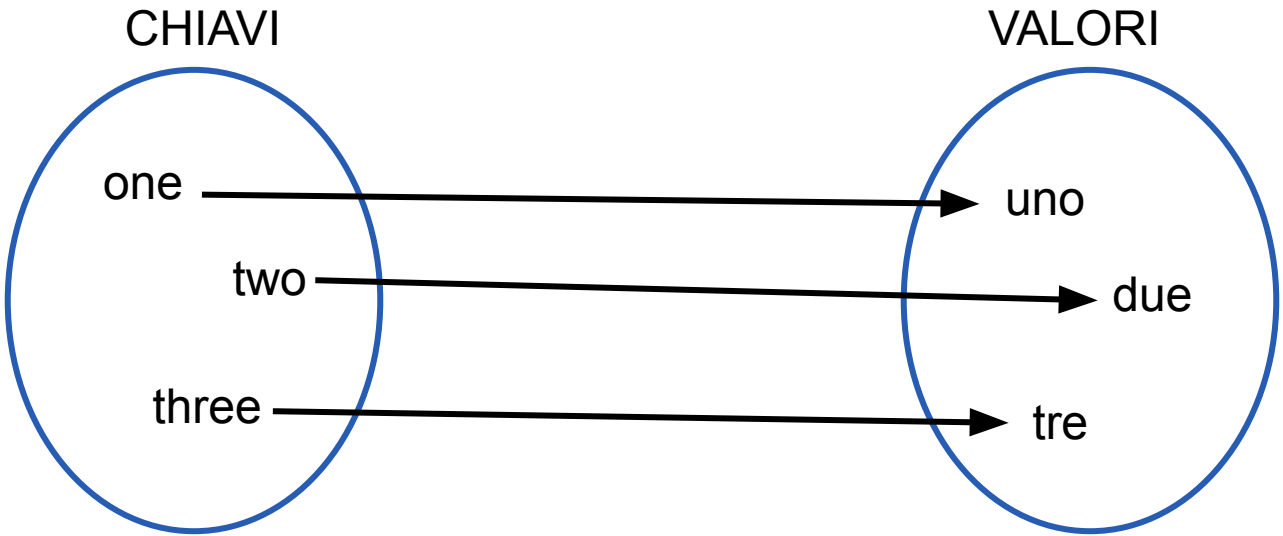
- Contenitori che memorizzano associazioni tra chiavi (keys) e valori (values): ogni chiave del dizionario è associata a un valore ed è univoca mentre un valore può essere associato a più chiavi

```
word1 = {
"forma" :  "gatte",
"lemma" :  "gatto",
"pos" :  "nome",
"freq" : 25
 }
```



Dizionario = mappa

- Un dizionario rappresenta una relazione di corrispondenza, o mappatura, da una chiave a un valore



Dizionari: usi

- Creare:
  - `eng2it = {'one': 'uno', 'two': 'due', 'three': 'tre'}`
  - `eng2it = {}` → dizionario vuoto
  - `old_eng2it = dict(eng2it)` → funzione per creare una copia
  - `len(eng2it)` → numero di coppie chiave-valore, es. 3
  - `dict.fromkeys(keys, value)` → restituisce un dizionario con le chiavi e il valore specificati (il valore è opzionale, se non specificato viene assegnato un valore “None”

Dizionari: usi

- Accedere ai valori:

N.B. NON si può accedere agli elementi di un dizionario con un indice di posizione come nelle liste (la posizione è ininfluente) ma solo attraverso la sua chiave

  - `nome_dict['key']` → es. `eng2it['two']` darà 'due'
  - `nome_dict.keys()` → restituisce la sequenza di chiavi
  - `nome_dict.values()` → restituisce la sequenza di valori

Dizionari: usi

- Modificare:
  - `nome_dict['key'] = value` → aggiunta di una nuova coppia, es. `eng2it['four']='quattro'`

    N.B. Per modificare il valore, si assegna un valore diverso a una chiave esistente
  - `nome_dict.pop('key')` → rimuove dal dizionario la chiave specificata e il suo valore

Dizionari: usi

- Operatore `in`:
  - `nome_chiave in nome_dict` → controlla se la chiave è presente nel dizionario e restituisce True o False
  - `value in nome_dict.values()` → controlla se il valore è presente nel dizionario e restituisce True o False

Dizionari: scandire gli elementi

- Per scandire gli elementi si usa il ciclo for che attraversa le chiavi o i valori del dizionario
  - usare il ciclo `for`

```
for key in nome_dict:
    print(key)
```

→ sequenza di chiavi

```
for value in nome_dict.values():
    print(value)
```

→ sequenza di valori

Dizionari: tipi di dati ammessi

- I valori possono essere di qualunque tipo, anche una lista mentre le chiavi NON possono essere liste perché queste ultime sono oggetti modificabili

```
dizionario = { keyName1 : value1, keyName2: value2,
        keyName3: [val1, val2, val3] }
```

Un po’ di pratica

- Lezione5.ipynb

  https://colab.research.google.com/drive/1pg9yIe8uDjsntYOICCAtWWn9j2ITMdwr?usp=sharing

Esercizio

- Riprendere l'esercizio 1 della lezione precedente: estrarre tutte le parole con suffisso "-mente" dal file "canzonierePetrarca.txt"
  - contare quanti elementi ci sono nella lista di output
  - controllare se "velocemente" è presente nell'output
  - contare le occorrenze di "dolcemente"
  - per ogni elemento della lista stampare l'elemento stesso insieme alla sua frequenza di occorrenza, es:

    ```
    Frequenza di 'visibilmente': 2
    ```

Soluzione esercizio

- https://colab.research.google.com/drive/13id3qGI--WkrS8PTuyWTbdofe0uqsWLK?usp=sharing