7_23-11-2022

May 28, 2023

[ ]:
```
# Initialize a list.

a = [1.0, 2.0, 3.0]
print(type(a))

a = [1, 1.0, "hello"]
print(type(a))

a = [1, 1.0, ["hello", "world"]]
print(type(a))

# Initialize a dictionary.

a = {"name": "William", "surname": "Shockley"}
print(type(a))

a = {"name": "William", "surname": "Shockley", "birth": {"day": 13, "month": 2,␣
 ↪"year": 1910}}
print(type(a))

# Lists and dictionaries.

a = ["William", "Shockley", {"day": 13, "month": 2, "year": 1910}]
print(type(a))

a = {"name": "William", "surname": "Shockley", "birth": [13, 2, 1910]}
print(type(a))

# Spaces and newlines.

a = {
    "name": "William",
    "surname": "Shockley",
    "birth": [
        13,
        2,
        1910
```

```
    ]
}
print(type(a))

# Access single elements.

a = ["William", "Shockley", {"day": 13, "month": 2, "year": 1910}]
print(a[0])
print(a[1])

a = {"surname": "Shockley", "name": "William", "birth": [13, 2, 1910]}
print(a["name"])
print(a["surname"])
print(a["birth"][0])

# List slicing.

a = ["a", "b", "c", "d", "e", "f"]
print(a[0:2])

print(a[2:3])

print(a[2:])

print(a[:2])

print(a[:-1])

print(a[0::3])

print(a[1:-1:2])

print(a[0:-1])

nome = "alessia"

s = f"ciao {nome}, come stai?"


# Operations with lists.

# Modify an element.

a = ["William", "Shockley", {"day": 13, "month": 2, "year": 1910}]

a[0] = "William Bradford"
print(a)
```

```
# Append an element.
# Call a method of the class.

a.append("Physicist")
print(a)

# Delete the last element.

a.pop()
print(a)

# Insert an element.

a.insert(2, "Physicist")
print(a)

# Delete an element by index.

a.pop(2)
print(a)

# Delete an element by value.

a.remove("William Bradford")
print(a)

# Concatenate lists.
# Operator overload.

a = ["William"] + a
print(a)

# Extend list.

a.extend(["Physicist"])
print(a)

# Check a class methods.

#help(list)

# Strings as lists.

a = "Hello World."
print(type(a[0]))
print(a[0])
```

```python
# Lists and tuples.

a = ("William", "Shockley")
print(type(a))

#a.append("Physicist")  # error -> bisogna inserire le parentesi quadre e così diventa una lista, modificabile

# Operations with dictionaries.

# Delete key-value pair.

a = {"name": "William", "surname": "Shockley", "birth": [13, 2, 1910]}
del(a["surname"])
print(a)

# Initialize a new key-element pair.

a["surname"] = "Shockley"
print(a)


# Lists and sets.

a = ["William", "Shockley", "William", "William"]
s = set(a)
print(s)
print(type(s))

s = {"William", "Shockley"}
print(s)
print(type(s))

# Example: Union

a = {"a", "b"} | {"c", "d"}
print(a)

a = {"a", "b"}
a = a.union(["c", "d"])
print(a)

#s = set(["William", "Shockley", [13, 2, 1910]])  # error
```

```
<class 'list'>
<class 'list'>
<class 'list'>
```

```
<class 'dict'>
<class 'dict'>
<class 'list'>
<class 'dict'>
<class 'dict'>
William
Shockley
William
Shockley
13
['a', 'b']
['c']
['c', 'd', 'e', 'f']
['a', 'b']
['a', 'b', 'c', 'd', 'e']
['a', 'd']
['b', 'd']
['a', 'b', 'c', 'd', 'e']
['William Bradford', 'Shockley', {'day': 13, 'month': 2, 'year': 1910}]
['William Bradford', 'Shockley', {'day': 13, 'month': 2, 'year': 1910},
'Physicist']
['William Bradford', 'Shockley', {'day': 13, 'month': 2, 'year': 1910}]
['William Bradford', 'Shockley', 'Physicist', {'day': 13, 'month': 2, 'year':
1910}]
['William Bradford', 'Shockley', {'day': 13, 'month': 2, 'year': 1910}]
['Shockley', {'day': 13, 'month': 2, 'year': 1910}]
['William', 'Shockley', {'day': 13, 'month': 2, 'year': 1910}]
['William', 'Shockley', {'day': 13, 'month': 2, 'year': 1910}, 'Physicist']
<class 'str'>
H
<class 'tuple'>
{'name': 'William', 'birth': [13, 2, 1910]}
{'name': 'William', 'birth': [13, 2, 1910], 'surname': 'Shockley'}
{'Shockley', 'William'}
<class 'set'>
{'Shockley', 'William'}
<class 'set'>
{'d', 'c', 'b', 'a'}
{'d', 'c', 'b', 'a'}
```

RIASSUNTO:

La principale cosa da ricordare è la differenza tra i quattro tipi di ordinamento dei dati in python e cioè:

- lista: collezione di dati ordinata, modificabile -> [];
- tupla: collezione di dati ordinata, non modificabile -> ();
- dizionario: collezione di dati nella forma di coppie “chiave” - “valore” -> {};
- set: collezione di dati non ordinata -> {}.

LISTA 1. Può essere formata da solo numeri, solo parole, oppure un misto di entrambi (talvolta anche con una lista di parole):

- a = [1.0, 2.0, 3.0]
- a = [1, 1.0, “hello”]
- a = [1, 1.0, [“hello”, “world”]]

le parole vanno messe tra “ “, le liste di parole tra [];

2. Sono possibili operazioni di slicing (N.B. quando si indica un intervallo, per definizione il secondo estremo è escluso); data a una lista:

- print(a[i:j]) -> stampa i valori compresi tra i e j;
- print(a[i:]) -> stampa i valori da i in poi;
- print(a[:i]) -> stampa i valori prima di i (di nuovo, i escluso);
- print(a[:-1]) -> stampa i valori fino all’ultimo escluso (a partire da destra);
- print(a[i::n]) -> stampa i valori a partire da i prendendone uno ogni n volte;
- print(a[i:-1:n]) -> stampa i valori da i all’ultimo prendendone uno ogni n volte

3. modifiche interne - data a una lista:

- a[0] = …/“…” -> va a modificare il primo elemento della lista (che sia un numero o una parola);
- a.append(…/“…”) -> aggiunge alla fine della lista il nuovo elemento;
- a.pop() -> toglie dalla lista l’ultimo elemento;
- a.insert(i, …/“…”) -> inserisce un elemento nella posizione i desiderata (non sostituendo, ma facendo scalare gli indici);
- a.pop(i) -> toglie l’elemento nella posizione i (con indice);
- a.remove(…/“…”) -> toglie l’elemento (con il valore)

4. si possono concatenare liste fra di loro: a = […/“…”] + a

5. si può estendere una lista aggiungendo un elemento sottoforma di lista: a.extend([…/“…”])

6. è possibile vedere le stringhe come liste di cui ogni elemento a[i] rappresenta una lettera

7. si può accedere a singoli elementi della lista tramite il comando print(a[i])

TUPLA a = (…, …) -> non si possono operare le modifiche riportate ai punti 3, 4, 5, 6 della sezione LISTA;

DIZIONARIO 1. è formato da coppie chiave - valore; è possibile inserire inoltre un dizionario nel dizionario: - a = {“…” : “…”, “…” : “…”} - a = {“…” : “…”, “…” : {“…” : “…”, “…” : “…”}}

2. è possibile inserire un dizionario all’interno di una lista come suo elemento, oppure una lista in un dizionario

3. andando a capo e indentando si ottiene l’a capo anche nella stampa

4. così come nelle liste, si può accedere a singoli elementi del dizionario tramite il comando print(a[i]); in particolare, se una coppia del dizionario è formata da una chiave e una lista, per accedere ad un particolare elemento della lista interna basta eseguire il comando print(a[“nome della chiave”][i]) dove i è l’elemento della lista legato alla chiave.

5. sono possibili operazioni di modifica simili a quelle disponibili per le liste, per cui data a lista:

- del(a[“chiave da eliminare”]) -> elimina una coppia chiamando la sua chiave

- a[“chiave”] = …/“…” -> inizializza una nuova coppia chiave - valore

SET 1. Quando si stampa un set, se al suo interno ci sono dei valori ripetuti, essi non vengono ripetutamente stampati. Infatti, i set non contengono duplicati e non mantengono l’ordine degli elementi; inoltre, contengono solo elementi singoli. Per questo motivo, non è possibile inserire delle liste nel set in quanto, poichè modificabili, potrebbero portare a duplicati; set di tuple sono invece possibili.

2. sono possibili operazioni di unione:

- a = {“a”, “b”} | {“c”, “d”} -> tramite |
- a = a.union([“c”, “d”]) -> conosciuto a a priori