

Corso ITS: *ARTIFICIAL INTELLIGENCE SPECIALIST*

Modulo: Programmazione Procedurale in Python

Docente: *Andrea Ribuoli*

Martedì 28 Gennaio 2025

09:00 - 14:00

- per memorizzare raccolte di valori Python usa le **liste**
- trattandosi di contenitori verranno scandite tramite `for`
- supponiamo di leggere (con un ciclo) una serie di numeri
- vogliamo stamparli evidenziando il **valore massimo**
- non possiamo farlo prima di aver acquisito l'intero elenco
- una lista si inizializza con le **parentesi quadre** `[]`
- i valori iniziali sono separati da virgole
- accedo agli elementi con l'**indice** `(lista[indice])`
- per accedere tramite indice la posizione **deve già esistere**
- `esempio = []` crea un **lista vuota**
- ha senso? Sì, perchè posso **aggiungere** elementi
- `esempio.append(valore)` metodo **append()**
- il metodo `append()` accoda alla fine della lista
- col metodo **insert(posizione, valore)** si aggiunge ovunque
- gli elementi successivi vengono spostati
- non posso indicare come posizione un indice fuori dal range
- `if 30 in esempio` : è vero se uno dei valori presenti è 30
- `esempio.index(30)` ci dice la prima posizione in cui è presente

```
In [2]: esempio = []
esempio.append(24)
esempio.append(27)
esempio.append(30)
esempio.append(22)
```

```

esempio.append(28)
esempio.append(30)
n = esempio.index(30)
n2 = esempio.index(30, n + 1)
print("Ho preso 30 negli esami", n, "e", n2)

```

Ho preso 30 negli esami 2 e 5

- se non presente la *index* genera una eccezione
- nell'incertezza devo verificare con `if 30 in esempio :`
- è possibile eliminare (estrarre) elementi con `pop(indice)`
- i successivi "*slittano*"
- se ometto l'indice, `pop()`, estraggo l'ultimo elemento
- il metodo `remove(valore)` elimina in base al valore
- il valore deve esistere

```
In [5]: a = esempio.remove(30)
```

```
In [10]: a = esempio.remove(30)
```

```
In [11]: a = esempio.remove(30)
```

```

-----
ValueError                                Traceback (most recent call last)
Input In [11], in <cell line: 1>()
----> 1 a = esempio.remove(30)

ValueError: list.remove(x): x not in list

```

- il confronto tra due liste considera l'ordine dei valori

Prima di rimuovere i due 30:

```
In [3]: print(sum(esempio)/len(esempio))
```

26.833333333333332

dopo:

```
In [6]: print(sum(esempio)/len(esempio))
```

25.25

- il metodo **list()** crea "*veramente*" una nuova lista
- l'operatore **porzione**(*slice*) si indica coi **due punti** (:)
- `esempio[a : b]`
- **a**: indice elemento da includere
- **b**: indice elemento da escludere

- **entrambi facoltativi**
- `esempio[: b]` : tutti fino alla posizione `b` **esclusa**
- `esempio[a :]` : tutti dalla posizione `a` **inclusa**

```
valori = []
print('Inserisci i voti degli esami separati da Invio')
print('Termina con Q:')
inputUtente = input("")
while inputUtente.upper() != "Q" :
    valori.append(int(inputUtente))
    inputUtente = input("")
```

- esiste un metodo per l'ordinamento: **sort()**
- in Python non esistono **tabelle**
- possiamo però creare **liste di liste**

```
In [17]: scacchiera = []
RIGHE   = 8
COLONNE = 8

for i in range(RIGHE) :
    riga = [" "] * COLONNE
    scacchiera.append(riga)
```

```
In [7]: lista = [ "Mi", "piace", "molto", "programmare", "in", "Python" ]
lista2 = lista
lista3 = list(lista)
lista[5] = "Ruby"
msg = ""
for elem in lista2 :
    msg += elem + " "
print(msg)
msg = ""
for elem in lista3 :
    msg += elem + " "
print(msg)
```

Mi piace molto programmare in Ruby
 Mi piace molto programmare in Python

```
In [8]: !python3 resoconto.py statistica
```

- 1 Silvia Amicucci
- 2 Juan Sebastian Baffoni
- 3 Maria Beatrice Bozzi
- 4 Giacomo Bramucci
- 5 Lorenzo Cassiani
- 6 Alex Fattorini
- 7 Alessia Fella
- 8 Andrian Ghiba
- 9 Erika Guerra
- 10 Iron Italiani
- 11 Davide Izzo
- 12 Daniele Malpassi
- 13 Roberto Marzi
- 14 Christian Paperi
- 15 Massimo Pontellini
- 16 Stefano Pontellini
- 17 Pierangelo Quarato
- 18 Elena Rapisardi
- 19 Olesia Rudenko
- 20 Lorenzo Scavolini
- 21 Fabio Solari
- 22 Grent Sota
- 23 Riccardo Tamanti

Esercizio

Un negozio di animali domestici vuole fare uno sconto ai clienti che acquistano un animale (o più) e almeno cinque altri articoli. Lo sconto è il 20% del costo degli altri articoli, mentre gli animali sono esclusi.

Scrivete la funzione `discount(prices, isPet, nItems)` che calcoli lo sconto sulla base delle informazioni ricevute sulla vendita in esame, costituita da `nItems` articoli: per l'articolo `i`-esimo, `prices[i]` è il prezzo prima dell'eventuale sconto e `isPet` è `True` se l'articolo `i`-esimo è un animale.

Scrivete un programma che chieda al cassiere di digitare tutti i prezzi, ciascuno seguito da una `Y` se si tratta di un animale e da una `N` per tutti gli altri articoli, usando il prezzo `-1` come sentinella. Memorizzate in una lista i dati acquisiti, poi invocate la funzione che avete progettato e visualizzate lo sconto.

In []: