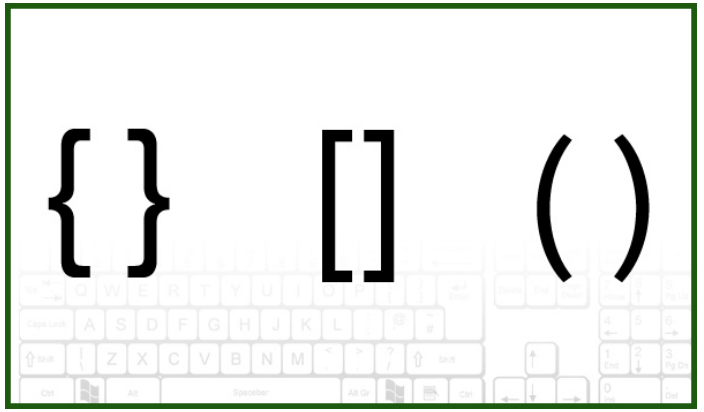
sequenze



sequenze

- liste
  - funzioni e metodi su liste
- uguaglianza e identità
- ciclo for
- range
- tuple

lista

- ***sequenza*** di elementi (*di solito dello stesso tipo*)
- l'intera lista può essere assegnata ad una variabile
- i singoli ***elementi*** sono ***numerati*** per ***posizione***
- gli ***indici*** partono da ***0***

```
lista_spesa = ["burro", "uova", "fagioli"]

piovosità = [13, 24, 18, 15]

mesi = ["Gen", "Feb", "Mar",
        "Apr", "Mag", "Giu",
        "Lug", "Ago", "Set",
        "Ott", "Nov", "Dic"]
```

accesso agli elementi

- attenzione ad usare ***indici validi!***
  - lunghezza attuale di una lista x: **`len(x)`**
  - elementi numerati da **`0`** a **`len(x)-1`**
  - indici negativi contano dalla fine

```
n = len(mesi)
print('numero mesi',n)
print('mesi[3] = ',mesi[3])              # "Apr"
print('mesi[-2] = ',mesi[-2])            # "Nov", -2 identico a n - 2
print('mesi[n-2] = ',mesi[n-2])
print('mesi[10] = ',mesi[10])
```

appartenenza, inserimento, rimozione

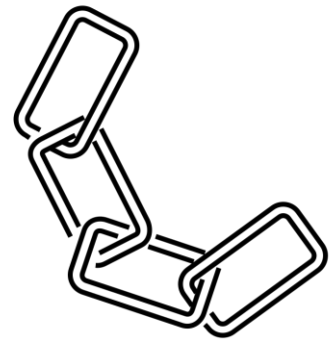
```
lista_spesa = ["burro", "uova", "fagioli"]
print('lista_spesa = ', lista_spesa)
print('"uova" in lista_spesa = ',"uova" in lista_spesa)  # True

lista_spesa.append("bacon")    # aggiunge in coda

lista_spesa.pop()    # elimina (e restituisce) l'ultimo elemento

lista_spesa.insert(1, "bacon")  # sposta gli elementi successivi

eliminato = lista_spesa.pop(1)  # elimina (e restituisce) elemento 1

lista_spesa.remove("uova")      # elimina l'elemento in base al valore
```

slice: porzioni di lista

```
mesi = ["Gen", "Feb", "Mar", "Apr", "Mag", "Giu",
        "Lug", "Ago", "Set", "Ott", "Nov", "Dic"]

primavera = mesi[2:5]        # ["Mar", "Apr", "Mag"]

primi3 = mesi[:3]            # ["Gen", "Feb", "Mar"]

ultimi3 = mesi[-3:]          # ["Ott", "Nov", "Dic"]

copia_mesi = mesi[:]         # Copia in una nuova lista
```

concatenazione e ripetizione

```
lista1 = ["burro", "uova", "fagioli"]
lista2 = ["carne", "pesce"]
lista_spesa = lista1 + lista2      # Concatenazione

ripetizioni = [1, 2] * 3           # [1, 2, 1, 2, 1, 2]

dieciZeri = [0] * 10 # [0, 0, 0, 0, 0, 0 , 0, 0, 0, 0]
```



uguaglianza e identità

```
a = [3, 4, 5]
b = a[:]          # b = [3, 4, 5] -- a new list!
b == a            # True, they contain the same values
b is a            # False, they are two objects in memory
                  # (try and modify one of them...)
c = a
c is a            # True, same object in memory
                  # (try and modify one of them...)
```

funzioni (metodi) sulle liste

- **sort():** Sorts the list in ascending order.
- **type(list):** It returns the class type of an object.
- **append():** Adds one element to a list.
- **extend():** Adds multiple elements to a list.
- **index():** Returns the first appearance of a particular value.
- **max(list):** It returns an item from the list with a max value.
- **min(list):** It returns an item from the list with a min value.
- **len(list):** It gives the overall length of the list.

- **clear():** Removes all the elements from the list.
- **insert():** Adds a component at the required position.
- **count():** Returns the number of elements with the required value.
- **pop():** Removes the element at the required position.
- **remove():** Removes the primary item with the desired value.
- **reverse():** Reverses the order of the list.
- **copy():** Returns a duplicate of the list.

stringhe e liste

- ***stringa***: sequenza ***immutabile*** di caratteri
- ***join*** e ***split***: da lista a stringa e viceversa

```
txt = "Monty Python's Flying Circus"
txt[0]    # 'M'
txt[1]    # 'o'
txt[-1]   # 's'
txt[6:12] # "Python"
txt[-6:]  # "Circus"

days = ["tue", "thu", "sat"]
txt = "|".join(days)  # "tue|thu|sat"

days = "mon|wed|fri".split("|")  # ["mon", "wed", "fri"]
```

for - ciclo su liste

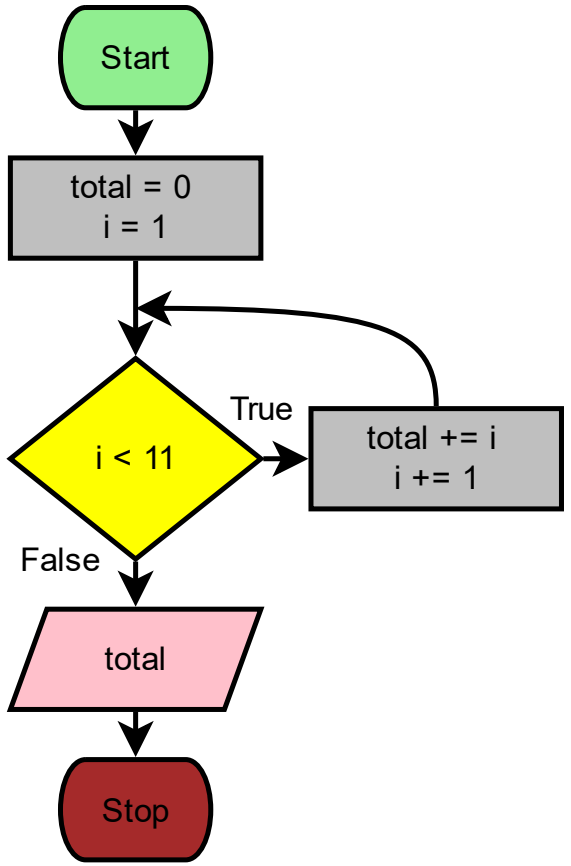
- il ciclo ***for*** permette di iterare su qualsiasi tipo di sequenza
  - ***lista, stringa, tupla, range…***
- *nell'esempio* ad **ogni iterazione**, alla variabile ***t*** è assegnato un elemento della lista ***corsi***

```
corsi = ['Artificial Intelligence','Machine Learning','Deep Learning']
for c in corsi:
    print(c)
```

intervallo di valori: range

- ***range***: intervallo di valori aperto a destra
  - estremo inferiore incluso
  - estremo superiore escluso
  - iterabile con un ciclo for

```
# somma dei numeri da 1 a 10
totale = 0
for i in range(1, 11):
    totale += i
print(totale)

# totale = 0;  i = 1
# while i < 11:
#     totale += i
#     i += 1
```



funzioni applicate a liste

```
def limitaValori(lis,limite):
    ''' fissa un limite massimo ai valori della lista lista '''
    for i in range(len(lis)):
        if lis[i] > limite:
            lis[i] = limite

def stampaValori(lis):
    for i, val in enumerate(lis):
        print('indice', i, 'valore', val)

dati = [5, 4, 2]
print(dati)
limitaValori(dati, 4)
print(dati)
stampaValori(dati)
```

The enumerate() method adds counter to an iterable and returns it (the enumerate object)

tuple

	Mutable	Ordered	Indexing / Slicing	Duplicate Elements
List	✓	✓	✓	✓
Tuple	✗	✓	✓	✓
Set	✓	✗	✗	✗

tupla

- sequenza ***immutabile*** di valori (*anche di tipo diverso*)

```
# Tuple packing
pt = 5, 6, "red"
pt[0]  # 5
pt[1]  # 6
pt[2]  # "red"

# multiple assignments, from a tuple
x, y, colour = pt  # sequence unpacking
a, b = 3, 4
a, b = b, a
```

passaggio dei parametri – call by object

- parametri passati «per oggetto»
  - se il parametro è una ***variabile*** le modifiche non si ripercuotono all'esterno
  - se il parametro è una ***lista*** o un ***oggetto*** le modifiche si ripercuotono

```
def inc(f):
    f = f + 1
    print(f) # 11


a = 10
inc(a)
print(a)     # 10
```

```
def inc(f):
    for i in range(0,len(f)):
        f[i] = f[i] + 1
    print(f) # [3,4,6]

a = [2,3,5]
inc(a)
print(a) # [3,4,6]
```

restituzione di più valori

- la funzione restituisce una tupla

```
def min_max(f):
    '''
    restituisce valore minimo e massimo della lista f
    '''
    minimo = massimo = f[0]
    for i in range(1,len(f)):
        if f[i] < minimo:
            minimo = f[i]
        if f[i] > massimo:
            massimo = f[i]
    return minimo, massimo

a = [2,13,5,-3,8]
x , y = min_max(a)
print("minimo: ",x," massimo: ",y)
```