Corso ITS: *ARTIFICIAL INTELLIGENCE SPECIALIST*

Modulo: **Programmazione ad oggetti in Python e librerie esterne**

Docente: *Andrea Ribuoli*

Mercoledì **26 Febbraio** 2025

09:00 - 14:00

In [89]:
```
import urllib.request
address ='https://www.comuni-italiani.it/province.html'
response = urllib.request.urlopen(address)
theBytes = response.read()
html = theBytes.decode(encoding='iso-8859-1')
from bs4 import BeautifulSoup
t1 = BeautifulSoup(html, "html.parser").table
t2 = t1.find_next("table")
t3 = t2.find_next("table")
t4 = t3.find_next("table")
riga = t4.find_next("tr")
riga = riga.find_next("tr")
import pandas as pd
province = pd.DataFrame(columns = ["sigla", "nome", "abitanti"])
i = 0
while riga != None :
    tdx  = riga.find_next("td")
    tdx  = tdx.find_next("td")
    prov = tdx.get_text()
    tdx  = tdx.find_next("td")
    abit = int(tdx.get_text().replace(".",""))
    tdx  = tdx.find_next("td")
    tdx  = tdx.find_next("td")
    tdx  = tdx.find_next("td")
    tdx  = tdx.find_next("td")
    tdx  = tdx.find_next("td")
    sigla = tdx.get_text()
    province.loc[i] = [sigla, prov, abit]
    if sigla == 'VT' : break
    i += 1
    riga = riga.find_next("tr")
province.sort_values(by="abitanti", ascending=False).to_csv("province.csv",
province.to_json("province.json")
```

In [90]:
```
from csv import reader
infile = open("province.csv")
csvReader = reader(infile)
for riga in csvReader :
    print(riga)
```

```
['sigla', 'nome', 'abitanti']
['RM', 'Roma', '4353738']
['MI', 'Milano', '3218201']
['NA', 'Napoli', '3107006']
['TO', 'Torino', '2277857']
['PA', 'Palermo', '1268217']
['BS', 'Brescia', '1262318']
['BA', 'Bari', '1260142']
['CT', 'Catania', '1113303']
['BG', 'Bergamo', '1109933']
['SA', 'Salerno', '1104731']
['FI', 'Firenze', '1014423']
['BO', 'Bologna', '1009210']
['PD', 'Padova', '936274']
['CE', 'Caserta', '924166']
['VR', 'Verona', '921557']
['VA', 'Varese', '890043']
['TV', 'Treviso', '885972']
['MB', 'Monza e della Brianza', '868859']
['VI', 'Vicenza', '865082']
['VE', 'Venezia', '854275']
['GE', 'Genova', '850071']
['LE', 'Lecce', '802082']
['CS', 'Cosenza', '711739']
['MO', 'Modena', '700862']
['PG', 'Perugia', '660690']
['ME', 'Messina', '636653']
['FG', 'Foggia', '628556']
['CO', 'Como', '600190']
['CN', 'Cuneo', '589108']
['TA', 'Taranto', '583479']
['LT', 'Latina', '574891']
['CA', 'Cagliari', '560373']
['RC', 'Reggio Calabria', '553861']
['PV', 'Pavia', '547251']
['TN', 'Trento', '538604']
['RE', 'Reggio Emilia', '532483']
['UD', 'Udine', '531466']
['BZ', 'Bolzano', '524256']
['FR', 'Frosinone', '493067']
['AN', 'Ancona', '474124']
['PR', 'Parma', '448899']
['AG', 'Agrigento', '442049']
['TP', 'Trapani', '434476']
['AL', 'Alessandria', '426658']
['AV', 'Avellino', '423506']
['PI', 'Pisa', '421851']
['MN', 'Mantova', '412610']
['SR', 'Siracusa', '402822']
['BR', 'Brindisi', '397083']
['FC', 'Forlì-Cesena', '394067']
['BT', 'Barletta-Andria-Trani', '392546']
['RA', 'Ravenna', '391414']
['LU', 'Lucca', '390042']
['CH', 'Chieti', '389169']
['PZ', 'Potenza', '370680']
```

```
['NO', 'Novara', '370143']
['CZ', 'Catanzaro', '362343']
['PU', 'Pesaro e Urbino', '360711']
['CR', 'Cremona', '359388']
['FE', 'Ferrara', '348362']
['AR', 'Arezzo', '344374']
['LC', 'Lecco', '339238']
['LI', 'Livorno', '337334']
['RN', 'Rimini', '336786']
['SS', 'Sassari', '333116']
['RG', 'Ragusa', '321359']
['PE', 'Pescara', '321309']
['VT', 'Viterbo', '319008']
['MC', 'Macerata', '318921']
['PN', 'Pordenone', '312051']
['TE', 'Teramo', '309859']
['AQ', "L'Aquila", '301910']
['PT', 'Pistoia', '291839']
['PC', 'Piacenza', '286758']
['BN', 'Benevento', '279675']
['SV', 'Savona', '279408']
['CL', 'Caltanissetta', '269710']
['SI', 'Siena', '268341']
['PO', 'Prato', '254608']
['RO', 'Rovigo', '238588']
['TS', 'Trieste', '234682']
['LO', 'Lodi', '229338']
['TR', 'Terni', '228218']
['CB', 'Campobasso', '224644']
['GR', 'Grosseto', '223045']
['SP', 'La Spezia', '220698']
['AT', 'Asti', '216677']
['IM', 'Imperia', '215130']
['AP', 'Ascoli Piceno', '209450']
['BL', 'Belluno', '205781']
['MT', 'Matera', '199685']
['MS', 'Massa-Carrara', '196580']
['SO', 'Sondrio', '181437']
['BI', 'Biella', '178551']
['KR', 'Crotone', '175566']
['FM', 'Fermo', '174849']
['VC', 'Vercelli', '173868']
['EN', 'Enna', '168052']
['VV', 'Vibo Valentia', '161619']
['OR', 'Oristano', '160746']
['OT', 'Olbia-Tempio', '160672']
['VB', 'Verbano-Cusio-Ossola', '159664']
['RI', 'Rieti', '157420']
['NU', 'Nuoro', '156096']
['GO', 'Gorizia', '139673']
['AO', 'Aosta', '126883']
['CI', 'Carbonia-Iglesias', '126324']
['VS', 'Medio Campidano', '98623']
['IS', 'Isernia', '85805']
['OG', 'Ogliastra', '57185']
```

In [91]:
```
import json
infile = open("province.json")
stringone = infile.read()
data = json.loads(stringone)
data
```

Out[91]:

```
{'sigla': {'0': 'AG',
           '1': 'AL',
           '2': 'AN',
           '3': 'AO',
           '4': 'AR',
           '5': 'AP',
           '6': 'AT',
           '7': 'AV',
           '8': 'BA',
           '9': 'BT',
           '10': 'BL',
           '11': 'BN',
           '12': 'BG',
           '13': 'BI',
           '14': 'BO',
           '15': 'BZ',
           '16': 'BS',
           '17': 'BR',
           '18': 'CA',
           '19': 'CL',
           '20': 'CB',
           '21': 'CI',
           '22': 'CE',
           '23': 'CT',
           '24': 'CZ',
           '25': 'CH',
           '26': 'CO',
           '27': 'CS',
           '28': 'CR',
           '29': 'KR',
           '30': 'CN',
           '31': 'EN',
           '32': 'FM',
           '33': 'FE',
           '34': 'FI',
           '35': 'FG',
           '36': 'FC',
           '37': 'FR',
           '38': 'GE',
           '39': 'GO',
           '40': 'GR',
           '41': 'IM',
           '42': 'IS',
           '43': 'SP',
           '44': 'AQ',
           '45': 'LT',
           '46': 'LE',
           '47': 'LC',
           '48': 'LI',
           '49': 'LO',
           '50': 'LU',
           '51': 'MC',
           '52': 'MN',
           '53': 'MS',
           '54': 'MT',
           '55': 'ME',
```

```
 '56': 'MI',
 '57': 'MO',
 '58': 'MB',
 '59': 'NA',
 '60': 'NO',
 '61': 'NU',
 '62': 'OT',
 '63': 'OR',
 '64': 'PD',
 '65': 'PA',
 '66': 'PR',
 '67': 'PV',
 '68': 'PG',
 '69': 'PU',
 '70': 'PE',
 '71': 'PC',
 '72': 'PI',
 '73': 'PT',
 '74': 'PN',
 '75': 'PZ',
 '76': 'PO',
 '77': 'RG',
 '78': 'RA',
 '79': 'RC',
 '80': 'RE',
 '81': 'RI',
 '82': 'RN',
 '83': 'RM',
 '84': 'RO',
 '85': 'SA',
 '86': 'VS',
 '87': 'SS',
 '88': 'SV',
 '89': 'SI',
 '90': 'SR',
 '91': 'SO',
 '92': 'TA',
 '93': 'TE',
 '94': 'TR',
 '95': 'TO',
 '96': 'OG',
 '97': 'TP',
 '98': 'TN',
 '99': 'TV',
 '100': 'TS',
 '101': 'UD',
 '102': 'VA',
 '103': 'VE',
 '104': 'VB',
 '105': 'VC',
 '106': 'VR',
 '107': 'VV',
 '108': 'VI',
 '109': 'VT'},
'nome': {'0': 'Agrigento',
 '1': 'Alessandria',
```

```
'2': 'Ancona',
'3': 'Aosta',
'4': 'Arezzo',
'5': 'Ascoli Piceno',
'6': 'Asti',
'7': 'Avellino',
'8': 'Bari',
'9': 'Barletta-Andria-Trani',
'10': 'Belluno',
'11': 'Benevento',
'12': 'Bergamo',
'13': 'Biella',
'14': 'Bologna',
'15': 'Bolzano',
'16': 'Brescia',
'17': 'Brindisi',
'18': 'Cagliari',
'19': 'Caltanissetta',
'20': 'Campobasso',
'21': 'Carbonia-Iglesias',
'22': 'Caserta',
'23': 'Catania',
'24': 'Catanzaro',
'25': 'Chieti',
'26': 'Como',
'27': 'Cosenza',
'28': 'Cremona',
'29': 'Crotone',
'30': 'Cuneo',
'31': 'Enna',
'32': 'Fermo',
'33': 'Ferrara',
'34': 'Firenze',
'35': 'Foggia',
'36': 'Forlì-Cesena',
'37': 'Frosinone',
'38': 'Genova',
'39': 'Gorizia',
'40': 'Grosseto',
'41': 'Imperia',
'42': 'Isernia',
'43': 'La Spezia',
'44': "L'Aquila",
'45': 'Latina',
'46': 'Lecce',
'47': 'Lecco',
'48': 'Livorno',
'49': 'Lodi',
'50': 'Lucca',
'51': 'Macerata',
'52': 'Mantova',
'53': 'Massa-Carrara',
'54': 'Matera',
'55': 'Messina',
'56': 'Milano',
'57': 'Modena',
```

```
 '58': 'Monza e della Brianza',
 '59': 'Napoli',
 '60': 'Novara',
 '61': 'Nuoro',
 '62': 'Olbia-Tempio',
 '63': 'Oristano',
 '64': 'Padova',
 '65': 'Palermo',
 '66': 'Parma',
 '67': 'Pavia',
 '68': 'Perugia',
 '69': 'Pesaro e Urbino',
 '70': 'Pescara',
 '71': 'Piacenza',
 '72': 'Pisa',
 '73': 'Pistoia',
 '74': 'Pordenone',
 '75': 'Potenza',
 '76': 'Prato',
 '77': 'Ragusa',
 '78': 'Ravenna',
 '79': 'Reggio Calabria',
 '80': 'Reggio Emilia',
 '81': 'Rieti',
 '82': 'Rimini',
 '83': 'Roma',
 '84': 'Rovigo',
 '85': 'Salerno',
 '86': 'Medio Campidano',
 '87': 'Sassari',
 '88': 'Savona',
 '89': 'Siena',
 '90': 'Siracusa',
 '91': 'Sondrio',
 '92': 'Taranto',
 '93': 'Teramo',
 '94': 'Terni',
 '95': 'Torino',
 '96': 'Ogliastra',
 '97': 'Trapani',
 '98': 'Trento',
 '99': 'Treviso',
 '100': 'Trieste',
 '101': 'Udine',
 '102': 'Varese',
 '103': 'Venezia',
 '104': 'Verbano-Cusio-Ossola',
 '105': 'Vercelli',
 '106': 'Verona',
 '107': 'Vibo Valentia',
 '108': 'Vicenza',
 '109': 'Viterbo'},
'abitanti': {'0': 442049,
 '1': 426658,
 '2': 474124,
 '3': 126883,
```

```
'4': 344374,
'5': 209450,
'6': 216677,
'7': 423506,
'8': 1260142,
'9': 392546,
'10': 205781,
'11': 279675,
'12': 1109933,
'13': 178551,
'14': 1009210,
'15': 524256,
'16': 1262318,
'17': 397083,
'18': 560373,
'19': 269710,
'20': 224644,
'21': 126324,
'22': 924166,
'23': 1113303,
'24': 362343,
'25': 389169,
'26': 600190,
'27': 711739,
'28': 359388,
'29': 175566,
'30': 589108,
'31': 168052,
'32': 174849,
'33': 348362,
'34': 1014423,
'35': 628556,
'36': 394067,
'37': 493067,
'38': 850071,
'39': 139673,
'40': 223045,
'41': 215130,
'42': 85805,
'43': 220698,
'44': 301910,
'45': 574891,
'46': 802082,
'47': 339238,
'48': 337334,
'49': 229338,
'50': 390042,
'51': 318921,
'52': 412610,
'53': 196580,
'54': 199685,
'55': 636653,
'56': 3218201,
'57': 700862,
'58': 868859,
'59': 3107006,
```

```
'60': 370143,
'61': 156096,
'62': 160672,
'63': 160746,
'64': 936274,
'65': 1268217,
'66': 448899,
'67': 547251,
'68': 660690,
'69': 360711,
'70': 321309,
'71': 286758,
'72': 421851,
'73': 291839,
'74': 312051,
'75': 370680,
'76': 254608,
'77': 321359,
'78': 391414,
'79': 553861,
'80': 532483,
'81': 157420,
'82': 336786,
'83': 4353738,
'84': 238588,
'85': 1104731,
'86': 98623,
'87': 333116,
'88': 279408,
'89': 268341,
'90': 402822,
'91': 181437,
'92': 583479,
'93': 309859,
'94': 228218,
'95': 2277857,
'96': 57185,
'97': 434476,
'98': 538604,
'99': 885972,
'100': 234682,
'101': 531466,
'102': 890043,
'103': 854275,
'104': 159664,
'105': 173868,
'106': 921557,
'107': 161619,
'108': 865082,
'109': 319008}}
```

Pag	Approfondimento
82	SymPy
123	email

Pag	Approfondimento
149	matplotlib
319	Turtle Graphics
377	SciPy
434	csv
447	os
469	urllib
534	json
650	pygame
702	BeautifulSoup

- avendo salvato nei passi precedenti un file in formato **CSV**
- il cui nome è *province.csv*
- lo apriamo con un **reader** offerto dal modulo **csv** (Python standard library)
- scorriamo i record:
- - escludendo la testata
- - stampando solo le province aventi **sigla** nella lista **marche**
- - adottiamo l'operatore di formattazione `%`
- - per incolonnare i risultati emessi

In [92]:
```
from csv import reader
infile = open("province.csv")
csvReader = reader(infile)
marche = ['PU', 'AN', 'MC', 'AP', 'FM']
formato = "%-26s %11d"
header = True
for riga in csvReader :
    if header :
        header = False
        continue
    if riga[0] in marche :
        print(formato % (riga[1], int(riga[2])) )
infile.close()
```

```
Ancona                           474124
Pesaro e Urbino                  360711
Macerata                         318921
Ascoli Piceno                    209450
Fermo                            174849
```

Novità Python in formattazione stringhe

- la versione **3.6** di Python ha introdotto le cosiddette **f-string**
- sono ormai il metodo prederito per formattare il testo
- l'esempio precedente viene così riscritto:

In [93]:
```
from csv import reader
infile = open("province.csv")
csvReader = reader(infile)
marche = ['PU', 'AN', 'MC', 'AP', 'FM']
header = True
for riga in csvReader :
    if header :
        header = False
        continue
    if riga[0] in marche :
        provincia = riga[1]
        abitanti = int(riga[2])
        formato = f"{provincia:26.26s} {abitanti:11d}"
        print(formato)
infile.close()
```

```
Ancona                           474124
Pesaro e Urbino                  360711
Macerata                         318921
Ascoli Piceno                    209450
Fermo                            174849
```

- sono offerte molte nuove possibilità
- - come allineamento rispetto allo spazio disponibile:
- `---` `:<` (*sinistra*), `:^` (*centro*), `:>` (*destra*)
- - valore corrispondente **Unicode**: `:c`

In [94]:
```
testo = "caso di studio"
print(f"INIZIO>|{testo:<26.26s}|<FINE")
print(f"INIZIO>|{testo:^26.26s}|<FINE")
print(f"INIZIO>|{testo:>26.26s}|<FINE")
```

```
INIZIO>|caso di studio           |<FINE
INIZIO>|      caso di studio      |<FINE
INIZIO>|           caso di studio|<FINE
```

In [95]:
```
for seme in range(9824, 9831) :
    print(f"{seme:c}")
```

```
♠
♡
♢
♣
♤
♥
♦
```

NumPy

- dopo **Pandas** facciamo alcuni accenni al pacchetto **NumPy**
- è dedicato al supporto per strutture dati multidimensionali
- utili nei calcoli statistici e nella analisi dei dati

- La proprità commutativa dell'AND e dell'OR è analoga a quella dell'algebra tradizionale
- la dimostrazione è immediata dall'analisi delle rispettive tabelle della verità:
- - scambiando l'ordine di A e B il risultato non cambia

In [96]:
```
import numpy as np
andMatrix = np.array([[False and False, False and True] ,
                      [True and False,  True and True ]])

print(andMatrix)
```

```
[[False False]
 [False  True]]
```

In [97]:
```
andMatrix.transpose()
```

Out[97]:
```
array([[False, False],
       [False,  True]])
```

In [98]:
```
if (andMatrix == andMatrix.transpose()).all() :
    print("La AND è commutativa")
else :
    print("La AND non è commutativa")
```

```
La AND è commutativa
```

In [99]:
```
orMatrix = np.array([[False or False, False or True] ,
                     [True or False,  True or True ]])

print(orMatrix)
```

```
[[False  True]
 [ True  True]]
```

In [100…
```
if (orMatrix == orMatrix.transpose()).all() :
    print("La OR è commutativa")
else :
    print("La OR non è commutativa")
```

```
La OR è commutativa
```

In [101…
```
xorMatrix = np.array([[False ^ False, False ^ True] ,
                      [True ^ False,  True ^ True ]])
print(xorMatrix)
if (xorMatrix == xorMatrix.transpose()).all() :
    print("La XOR è commutativa")
```

```
else :
    print("La XOR non è commutativa")
```

```
[[False  True]
 [ True False]]
La XOR è commutativa
```

In [102…
```
riga = np.array([list(range(1, 11))])
print(riga)
```

```
[[ 1  2  3  4  5  6  7  8  9 10]]
```

In [103…
```
colonna = riga.transpose()
print(colonna)
```

```
[[ 1]
 [ 2]
 [ 3]
 [ 4]
 [ 5]
 [ 6]
 [ 7]
 [ 8]
 [ 9]
 [10]]
```

In [104…
```
matrice = riga * colonna
print(matrice)
```

```
[[  1   2   3   4   5   6   7   8   9  10]
 [  2   4   6   8  10  12  14  16  18  20]
 [  3   6   9  12  15  18  21  24  27  30]
 [  4   8  12  16  20  24  28  32  36  40]
 [  5  10  15  20  25  30  35  40  45  50]
 [  6  12  18  24  30  36  42  48  54  60]
 [  7  14  21  28  35  42  49  56  63  70]
 [  8  16  24  32  40  48  56  64  72  80]
 [  9  18  27  36  45  54  63  72  81  90]
 [ 10  20  30  40  50  60  70  80  90 100]]
```