# Appunti di informatica

Lezione 12

anno accademico 2015-2016

Mario Verdicchio

# Stinghe

- Le stringhe sono sequenze di caratteri

```
>>> len("ciao!")
5
>>> len("")
0
>>> name = "Mario Verdicchio"
>>> name[4]
'o'
>>> name[len(name)-1]
'o’
>>> for i in range(0,len(name)):
        print (i,name[i])
>>> for i in range(len(name)):
    print (i, name[len(name)-i-1])
```

# Stringhe

```
>>> name[0:]
'Mario Verdicchio'
>>> name[:len(name)]
'Mario Verdicchio'
>>> name[0:2]
'Ma'
>>> name[-3:]
'hio'
```

# Vari metodi per manipolare le stringhe

- center
- count
- endswith
- find
- isalpha
- isdigit
- join
- lower
- replace
- split
- startswith
- strip
- upper
- …

# Operatore “in”

```
>>> lista = ["Mario", "Gianni", "Luigi", "Paolo"]
>>> for nome in lista:
        if "o" in nome:
            print(nome)
```

```
Mario
Paolo
```

# Codifica caratteri - numeri

- chr(x)

per avere la lettera corrispondente al numero x

- ord(c)

per avere il numero corrispondente al carattere c

# raw_input

- Per far capire all'interprete Python versione 2.7 che l'input in arrivo deve essere preso così com'è sotto forma di sequenza di caratteri (e non interpretato come dei numeri interi)

# Crittografia (encrypt)

```
from __future__ import print_function

chiaro = raw_input("Digita una parola in lettere minuscole:”)
dist = input("Digita il valore della distanza:”)
codice = ""

for ch in chiaro:
    val = ord(ch)
    cval = val + dist
    if cval > ord('z'):
        cval = ord('a') + val  + dist - ord('z') - 1
    codice = codice + chr(cval)

print(codice)
```

# Crittografia (decrypt)

```
from __future__ import print_function

codice = raw_input("Digita il codice da decifrare:”)
dist = input("Digita il valore della distanza:”)
chiaro = ""

for ch in codice:
    cval = ord(ch)
    val = cval - dist
    if val < ord('a'):
        val = cval – dist + ord('z') - ord('a’) + 1
    chiaro = chiaro + chr(val)

print(chiaro)
```

# Funzioni

- Una funzione è un modo per organizzare un pezzo di codice sotto forma di un modulo con un nome
- Ogni volta che nel programma serve l'esecuzione di quel pezzo di codice, basterà scrivere il nome della funzione
- Come nelle funzioni matematiche, una funzione in Python riceve uno o più parametri in input e restituisce un risultato

# Esempio di funzione

Parola chiave che introduce la definizione di una funzione

Nome della funzione a nostra scelta (meglio se significativo)

Lista dei parametri in ingresso.
Se più di uno separati da virgole.

```
def quadrato(x):
    y = x*x
    return y
```

Corpo della funzione: tutte le operazioni che la funzione deve eseguire vanno scritte qui come se fosse un programma in Python. Di fatto una funzione è un "sottoprogramma": un programma all'interno del programma principale.

Quando si ha una variabile che contiene il risultato, si scrive la parola chiave "return" e poi il nome della variabile. L'esecuzione di return fa sì che il risultato venga dato al programma che si è avvalso della funzione, e la funzione termina. "Return" può anche essere seguito da un'espressione.

# Uso delle funzioni

```
from __future__ import print_function

def quadrato(x):
    return x*x

def media(v):
    somma = 0.0
    for num in v:
        somma = somma + num
    return somma/len(v)

print("ecco i primi dieci quadrati:")
for i in range(10):
    print(quadrato(i+1))

print("dammi la tua lista di voti, 0 per finire:")
lista = []
while True:
    voto = input()
    if voto == 0:
        break
    else:
        lista.append(voto)
print("la tua media e':", media(lista))
```

Nella definizione di una funzione, il parametro non è una vera propria variabile perché non ha un valore: è solo un segnaposto che serve a illustrare le operazioni eseguite dalla funzione. Il parametro si dice per questo “formale”.

Quando una funzione viene “chiamata” dal programma principale, ossia viene eseguita, deve ricevere i valori reali (“actual” in inglese) su cui eseguire le proprie operazioni. A causa di una traduzione errata oramai consolidata, questo parametro si chiama “attuale”.