Ricordiamo come si definisce una funzione

- Definiamo una funzione che restituisce il massimo di una lista passata come argomento:

```
def massimo_lista(lista) :
    max = lista[0]
    pos = 1
    while pos < len(lista) :
        if lista[pos] > max :
            max = lista[pos]
        pos = pos + 1
    return max
```

- La parola chiave **def** introduce una definizione di funzione
- Deve essere seguita dal nome della funzione e dalla lista dei parametri formali racchiusa tra parentesi; in questo caso

```
def massimo_lista(lista) :
```

- L'istruzione **return** restituisce una funzione con un valore. **return** senza un'espressione come argomento restituisce **None**

Ricordiamo come si definisce una funzione

- Vediamo come possiamo utilizzare la funzione appena definita per riprodurre il comportamento del codice scritto prima

```
def massimo_lista(lista) :
    max = lista[0]
    pos = 1
    while pos < len(lista) :
        if lista[pos] > max :
            max = lista[pos]
        pos = pos + 1
    return max
mia_lista = [4,24,-89,81,3,0,-12,31]
max = massimo_lista(mia_lista)
print  max
```

definizione funzione;
prima di poter utilizzare una funzione bisogna definirla!

corpo del programma che utilizza la funzione

Ricordiamo come si definisce una funzione

- Potevamo anche chiamare direttamente la funzione come argomento di **print**:

```
def massimo_lista(lista) :
    max = lista[0]
    pos = 1
    while pos < len(lista) :
        if lista[pos] > max :
            max = lista[pos]
        pos = pos + 1
    return max
mia_lista = [4,24,-89,81,3,0,-12,31]
print  massimo_lista(mia_lista)
```

definizione funzione;
prima di poter utilizzare una funzione bisogna definirla!

corpo del programma che utilizza la funzione

Ricordiamo come si definisce una funzione

- ...e anche chiamare la funzione con argomento dato da una lista:

```
def massimo_lista(lista) :
    max = lista[0]
    pos = 1
    while pos < len(lista) :
        if lista[pos] > max :
            max = lista[pos]
        pos = pos + 1
    return max
print  massimo_lista([4,24,-89,81,3,0,-12,31])
```

definizione funzione;
prima di poter utilizzare una funzione bisogna definirla!

corpo del programma che utilizza la funzione

Eseguire un programma in un file

- Invece di inserire i comandi a linea di comando, possiamo inserirli in un file
- Ad esempio, provate a copiare I seguenti comandi in un file con nome *massimo_lista.py* (l'estensione .py indica che il file contiene codice Python)

```
def massimo_lista(lista) :
    max = lista[0]
    pos = 1
    while pos < len(lista) :
        if lista[pos] > max :
            max = lista[pos]
        pos = pos + 1
    return max
print  massimo_lista([4,24,-89,81,3,0,-12,31])
```

- Per eseguire il programma digitare da linea di comando:

```
python  massimo_lista.py
81
```

risultato dell'esecuzione

Parametri di un programma

- Nell'esempio precedente si calcola il massimo di una lista definita in modo costante all'interno del codice
- È possibile passare la lista come *parametro* del programma, i.e.

```
python massimo_lista.py 4,24,-89,81,3,0,-12,31
```

- Ma come facciamo a leggere la lista all'interno del programma ?
- Quando si esegue l'interprete python, i parametri (o argomenti) sono inseriti in un array (lista) chiamata sys.argv
- Nel caso della chiamata

```
python massimo_lista.py 4,24,-89,81,3,0,-12,31
```

notare che non sono presenti spazi nella definizione della lista

```
sys.argv = ['massimo_lista.py','4,24,-89,81,3,0,-12,31']
```

- Notare che i parametri sono individuati utilizzando gli spazi: per questo motivo è importante definire la lista senza introdurre spazi fra gli elementi
- Per poter utilizzare sys.argv si devono importare nell'interprete delle procedure di sistema predefinite tramite il comando

```
import sys
```

Parametri di un programma

- Cerchiamo di capire come funziona il passaggio di parametri con il seguente programma da copiare in un file che chiameremo *parametri.py*

```
import sys

pos = 0

while pos < len(sys.argv) :

    print 'parametro',pos,':',sys.argv[pos]

    pos = pos + 1
```

notare l'uso della virgola per stampare più argomenti separati da uno spazio

- Provate ad eseguire il programma passando uno o più parametri separati da uno spazio, ad esempio

```
python parametri.py

python parametri.py pippo

python parametri.py pippo topolino

python parametri.py pippo,topolino

python parametri.py pippo, topolino
```

Parametri di un programma

- Ora siamo (quasi!) pronti per definire il nostro programma parametrico

```
import sys

def massimo_lista(lista) :
    max = lista[0]
    pos = 1
    while pos < len(lista) :
        if lista[pos] > max :
            max = lista[pos]
        pos = pos + 1
    return max

lista_in = [int(x) for x in sys.argv[1].split(',')]

print 'Il valore massimo di',lista_in,'e\'',massimo_lista(lista_in)
```

merita una spiegazione...

stringa.split(delimitatore)

```
>>> import string
>>> Verso = "Nel mezzo del cammin..."
>>> Verso.split()
['Nel', 'mezzo', 'del', 'cammin...']


>>> Verso = "Nel,mezzo,del,cammin..."
>>> Verso.split(',')
['Nel', 'mezzo', 'del', 'cammin...']
```

Inoltre...

- `for`
  - Guardiamo il libro online a pagina 99 del file pdf

- Da stringa a numero:

  `int('123')` restituisce il numero `123`