Calcolare il massimo di una lista

- Lunedì abbiamo definito un algoritmo per calcolare il valore massimo fra gli elementi di una lista predefinita di interi. In particolare, abbiamo:
  - deciso di rappresentare la lista di interi utilizzando le parentesi quadrate per delimitare l'inizio e la fine della lista, e la virgola come separatore fra gli elementi della lista:

    `[4,24,-89,81,3,0,-12,31]`
  - introdotto la funzione `len( )` che ritorna la lunghezza della lista:

    `len([4,24,-89,81,3,0,-12,31])` ritorna il valore 8
  - deciso di iniziare la numerazione delle posizioni degli elementi a partire da 0 (in analogia alla memoria centrale, dove la prima locazione di memoria è riferita dall'indirizzo con tutti i bit a 0):

    `[4,24,-89,81,3,0,-12,31]`

    `↑ ↑ ↑ ↑ ↑ ↑ ↑ ↑`

    `0 1 2 3 4 5 6 7 ← indice`
  - definito una notazione per accedere direttamente ad un elemento:

    `L ← [4,24,-89,81,3,0,-12,31]`

    `L[0]` ritorna il valore 4, `L[1]` ritorna il valore 24, `L[2]` ritorna il valore -89, …; se `i ← 4`, `L[i]` ritorna il valore 3, ...

Calcolare il massimo di una lista

- Ecco la definizione dell'algoritmo in pseudocodice, assumendo che `L` sia una variabile che riferisce una lista:

```
Max ← L[0]          Max ← L[len(L)-1]

i ← 1               i ← len(L)-2

lung ← len(L)

while (i < lung) do          ( i > -1)  (i >= 0)

    if (L[i] > Max) then  Max ← L[i]

    i ← i + 1       i ← i - 1

print Max
```

- Alcuni commenti:
  - molti di voi hanno trovato difficoltà a capire come “scorrere” la lista (uso di `i`): se `i` vale 2 allora `L[i]` vale `L[2]`, etc.
  - la condizione del **`while`** è vera fino a che `i` è minore o uguale a `len(L)-1`, infatti l'indice dell'ultima posizione è pari alla lunghezza della lista meno 1

Definizione di una procedura

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

```
procedure massimo_lista(L)
    Max ← L[0]
    i ← 1
    while (i < len(L)) do
        if (L[i] > Max) then Max ← L[i]
        i ← i + 1
    return Max
```

- La parola chiave (dello pseudocodice) **`procedure`** introduce una definizione di procedura
- Deve essere seguita dal nome della procedura e dalla lista dei parametri in input racchiusa tra parentesi; in questo caso

```
procedure massimo_lista(L)
```

- L'istruzione **`return`** restituisce il valore in output della procedura

Definizione di una procedura

- Esempio di chiamata di procedura

```
mia_lista ← [4,24,-89,81,3,0,-12,31]   # definisco una lista di numeri
max ← massimo_lista(mia_lista)         # chiamo la procedura con input mia_lista
    # la procedura restituirà il valore 81, che sarà assegnato alla variabile max
print max          # stampo a video il valore assegnato alla variabile max, cioé 81
```

- La definizione di una procedura è utile quando la si utilizza più volte nello stesso programma; esempio: somma massimi

```
esperimento_1 ← [1.3, 2.4, 0.9, 3.5, 1.4]   # risultati primo esperimento
esperimento_2 ← [3.1, 2.5, 1.9, 2.1, 2.4]   # risultati secondo esperimento
somma_max ← massimo_lista(esperimento_1) + massimo_lista(esperimento_2)
print somma_max   # stampa 6.6, cioè la somma di 3.5 con 3.1
```

Esercizio

Scrivere un algoritmo, utilizzando la procedura massimo_lista, che stampa gli elementi di una lista numerica, passata come parametro, dal valore più grande al valore più piccolo

- AIUTO: si può cancellare un elemento da una lista utilizzando la procedura predefinita

```
remove(lista,elemento)  # restituisce lista senza la prima occorrenza di
                        # elemento, se elemento è contenuto in lista
```

- Ad esempio

```
mia_lista ← [35, 5, -12, 8, 44]
remove(mia_lista, 5)
```

rimuove l'elemento 5, restituendo in output la lista `[35, -12, 8, 44]`

Introduzione al Python

- **Libro online**
  - **Differenza fra compilatore ed interprete**
    - Capitolo 1, pagina 26 del pdf
  - **Variabili, espressioni ed istruzioni**
    - Capitolo 2, pagina 34 del pdf
  - **Definizione di funzioni (procedure**
    - Capitolo 3, Sezione 3.6, pagina 46 del pdf

Calcolare il massimo di una lista

- Ecco la definizione dell'algoritmo in pseudocodice, assumendo che `L` sia una variabile che riferisce una lista:

```
Max ← L[0]
i ← 1
while (i < len(L)) do
    if (L[i] > Max) then  Max ← L[i]
    i ← i + 1
print Max
```

- Alcuni commenti:
  - molti di voi hanno trovato difficoltà a capire come “scorrere” la lista (uso di `i`): se `i` vale 2 allora `L[i]` vale `L[2]`, etc.
  - la condizione del **`while`** è vera fino a che `i` è minore o uguale a `len(L)-1`, infatti l'indice dell'ultima posizione è pari alla lunghezza della lista meno 1

Vediamo ora come implementare l'algoritmo in Python!

Implementare l'algoritmo in Python

- **Liste**
  - Capitolo 8, pagina 96 del pdf
- **Selezione**
  - Capitolo 4, pagina 55 del pdf
- **Iterazione**
  - Capitolo 6, Sezione 2, pagina 78 del pdf

Calcolare il massimo di una lista

Implementazione in Python

```
L = [4,24,-89,81,3,0,-12,31]

Max = L[0]   # questo e' un commento: primo elemento di lista

i = 1   # i si utilizza per accedere ai restanti elementi della lista

while i < len(L) :   # si controlla che il valore di i sia valido

    if L[i] > Max :  # se il valore "puntato" da i e' > Max

        Max = L[i] # tale valore deve diventare il massimo

    i = i + 1   #  incrementa i per "puntare all'elemento successivo

print Max   #  stampa il valore massimo assegnato a max
```

Cosa facciamo ora ?

- Impariamo a:
  - definire una funzione (procedura) che calcola il massimo di una lista
  - scrivere un programma in un file da far eseguire all'interprete Python
  - utilizzare in un programma i parametri passati dalla linea di comando
  - leggere e scrivere da programma uno o più file
  - utilizzare altre istruzioni di controllo del flusso
  - utilizzare altre *strutture dati*