



Home ► I miei corsi ► Offerta Formativa ► Corsi di Laurea ►
Dipartimento di Scienze Ambientali, Informatica e Statistica ► Informatica [CT3] ► CT0304 ►
Laboratorio ► Esercitazione3

ESERCITAZIONE3

SI RICORDA CHE SI POSSONO USARE SOLO I CONCETTI VISTI A LEZIONE E NON SI POSSONO USARE LE LIBRERIE

Alcune note utili:

Nelle esercitazioni potreste aver bisogno delle nozioni di Media Aritmetica, Media Geometrica e Media Armonica.

https://it.wikipedia.org/wiki/Media_%28statistica%29

Queste le pagine specifiche (in inglese) delle varie medie, in cui trovate degli esempi nella prima parte delle pagine.

Media Aritmetica:

https://en.wikipedia.org/wiki/Arithmetic_mean

Media Geometrica:

https://en.wikipedia.org/wiki/Geometric_mean

Media Armonica:

https://en.wikipedia.org/wiki/Harmonic_mean

Inoltre potreste aver bisogno della nozione di Moda Statistica.

https://it.wikipedia.org/wiki/Moda_%28statistica%29

Esercizio 1:

Data una lista non vuota di numeri interi positivi diversi da 0, si richiede di scrivere una funzione che: presa in *input* la lista, restituisca come *output* una tupla contenete in ordine la media aritmetica, la media geometrica e la media armonica.

`means [1;3;5;2;4] ==> (3.0; 2.605171; 2.189781)`

means [1..100] ==> (50.5; 37.992689; 19.277563)

means [4;5;2;1;6;2;2] ==> (3.142857; 2.667096; 2.245989)

means [3;4;4;1;4] ==> (3.20; 2.861938; 2.4)

means [2;2;2] ==> (2.0; 2.0; 2.0)

means [7] ==> (7.0; 7.0; 7.0)

Firma:

means: int list -> (float * float * float)

Esercizio 2:

Data una lista non vuota di numeri interi, si richiede di scrivere una funzione che:
presa in *input* la lista, restituisca come *output* la coppia contenente in ordine il numero minimo e il numero massimo della lista.

Esempio:

min_max [1;0;-1;2;0;-4] ==> (-4; 2)

min_max [7] ==> (7; 7)

min_max [3;2;1;0;-1;-2] ==> (-2; 3)

min_max [7;7;7;7;7;2;1;0] ==> (0; 7)

Firma:

min_max: int list -> (int * int)

Esercizio 3:

Data una lista non vuota di numeri interi si richiede di scrivere una funzione che:
presa in *input* la lista, restituisca come *output* la moda della lista. In caso più numeri abbiano la stessa occorrenza, si ritorni il numero che compare per primo nella lista tra quelli con l'occorrenza massima.

Esempio:

mode [1;2;5;1;2;3;4;5;5;4;5;5] ==> 5

mode [2;1;2;1;1;2] ==> 2

mode [-1;2;1;2;5;-1;5;5;2] ==> 2

mode [7] ==> 7

Firma:

mode: int list -> int

DOWNLOAD TEMPLATE ESERCITAZIONE:

Esercitazione3.zip

SOLUZIONI

Esercizio 1

//Calcola la lunghezza di una lista

```
let llength xs =  
  let rec llength_tail s xs =  
    match xs with  
    | [] -> s  
    | _::xs -> llength_tail (1 + s) xs  
  llength_tail 0 xs
```

//Trasforma una lista da int a float

```
let rec int_to_float_list (xs : int list) : float list =  
  match xs with  
  | [] -> []  
  | x::xs -> float x :: int_to_float_list xs
```

//Calcola la somma di tutti gli elementi di una lista

```
let rec lsum xs =  
  match xs with  
  | [] -> failwith "Empty list"  
  | [x] -> x  
  | x::xs -> x + lsum xs
```

//Calcola il prodotto di tutti gli elementi di una lista

```
let rec lprod xs =  
  match xs with  
  | [] -> failwith "Empty list"  
  | [x] -> x  
  | x::xs -> x * lprod xs
```

//Calcola la somma dei reciproci di tutti gli elementi di una lista

```
let rec lsum_reverse xs =  
  match xs with  
  | [] -> failwith "Empty list"  
  | [x] -> 1. / x  
  | x::xs -> (1. / x) + lsum_reverse xs
```

```

let rec means (l : int list) : (float * float * float) =
  if l = [] then
    failwith "Empty list"
  else
    let l = int_to_float_list l
    let arith l = float (lsum l) / float (llength l)
    let geometric l = (lprod l) ** (1. / float (llength l))
    let harmonic l =
      let up = float (llength l)
      let down = lsum_reverse l
      up / down
    (arith l,geometric l,harmonic l)

```

Esercizio 2

//Calcola il minimo di una lista

```

let lmin xs =
  let rec min_tail s xs =
    match xs with
    | [] -> s
    | x::xs ->
      if x < s then
        min_tail x xs
      else
        min_tail s xs
  in match xs with
  | [] -> failwith "Empty list"
  | x::xs -> min_tail x xs

```

//Calcola il massimo di una lista

```

let lmax xs =
  let rec max_tail s xs =
    match xs with
    | [] -> s
    | x::xs ->
      if x > s then
        max_tail x xs
      else
        max_tail s xs
  in match xs with
  | [] -> failwith "Empty list"
  | x::xs -> max_tail x xs

```

```

let rec min_max (l : int list) : (int * int) =
  lmin l, lmax l

```

Esercizio 3

```

let rec mode (l : int list) : int =

```

```
let rec count_occurrences e l =
  match l with
  | [] -> 0
  | x::xs when x = e -> 1 + count_occurrences e xs
  | x::xs -> count_occurrences e xs

let rec max_el l =
  match l with
  | [] -> failwith "Empty list"
  | [x] -> x, 1
  | x::xs ->
    let e, c = max_el xs
    let c_this = count_occurrences x l
    if c > c_this then
      e, c
    else
      x, c_this

let mode, _ = max_el l
mode
```

STATO CONSEGNA

Stato consegna	Nessun tentativo
Stato valutazione	Non valutata
Termine consegne	giovedì, 5 novembre 2015, 15:30
Tempo rimasto	Consegna in ritardo da: 18 giorni 1 ora

NAVIGAZIONE

Home

- My home
- Pagine del sito
- Il mio profilo
- Corso in uso
 - CT0304
 - Partecipanti
 - Introduzione
 - ORARIO DELLE LEZIONI - MODULO 1 (Prof.ssa Sabina R...
 - CALENDARIO DELLE LEZIONI
 - MATERIALE DIDATTICO

Laboratorio



Esercitazione0



Esercitazione1



Esercitazione2



Esercitazione3



Esercitazione4



Esercitazione5



Progetto F#

I miei corsi

■ [Uso dei Cookie](#)

AMMINISTRAZIONE



Amministrare il corso

Testo

Impostazioni profilo

Sei collegato come FRANCESCO BENETELLO. (Esci)

CT0304