

 POLITECNICO DI MILANO

Dipartimento di
Elettronica e Informazione

Undicesima esercitazione

Riccardo Cattaneo

Alessandro A. Nacci

Santambrogio M. D.

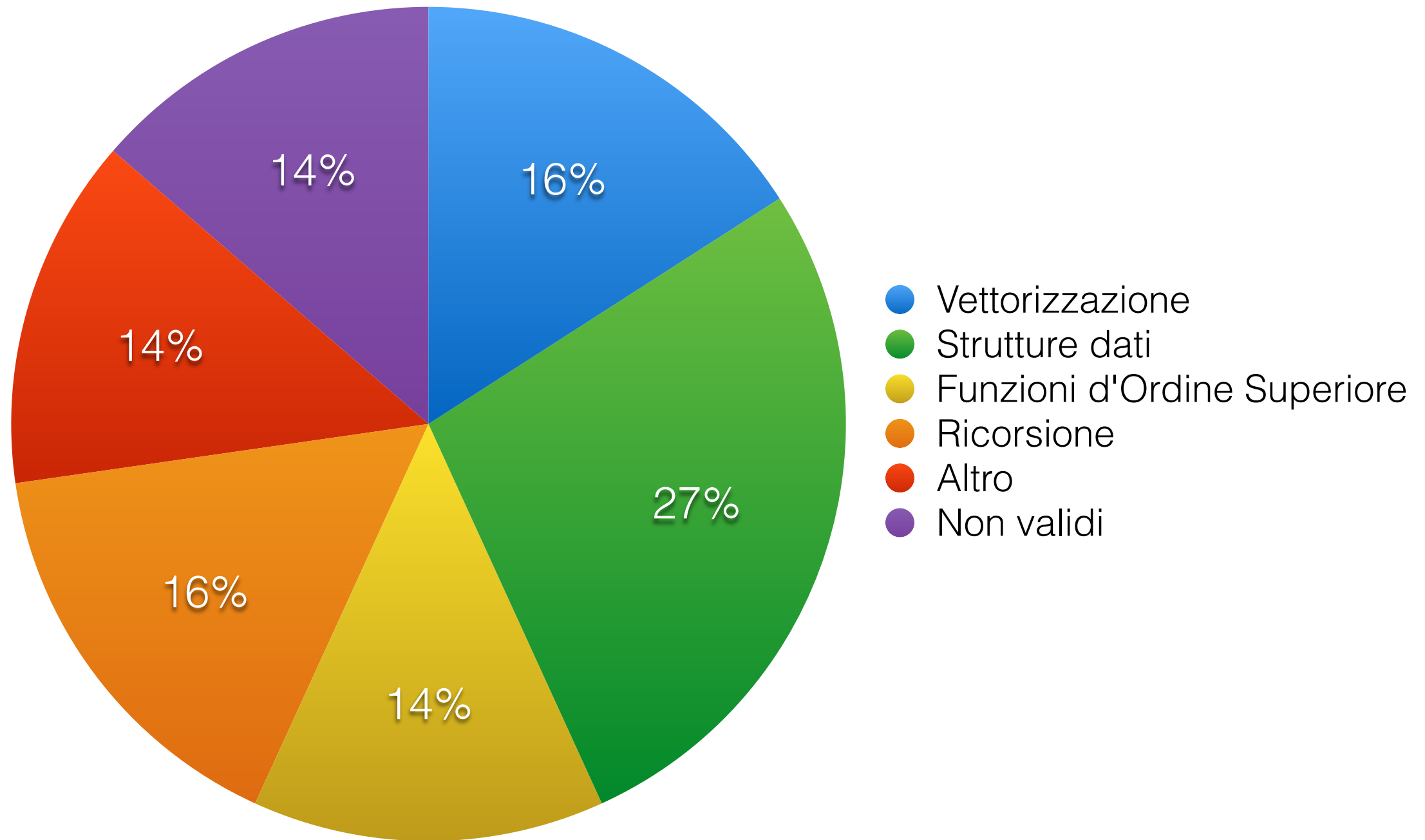


POLITECNICO
DI MILANO

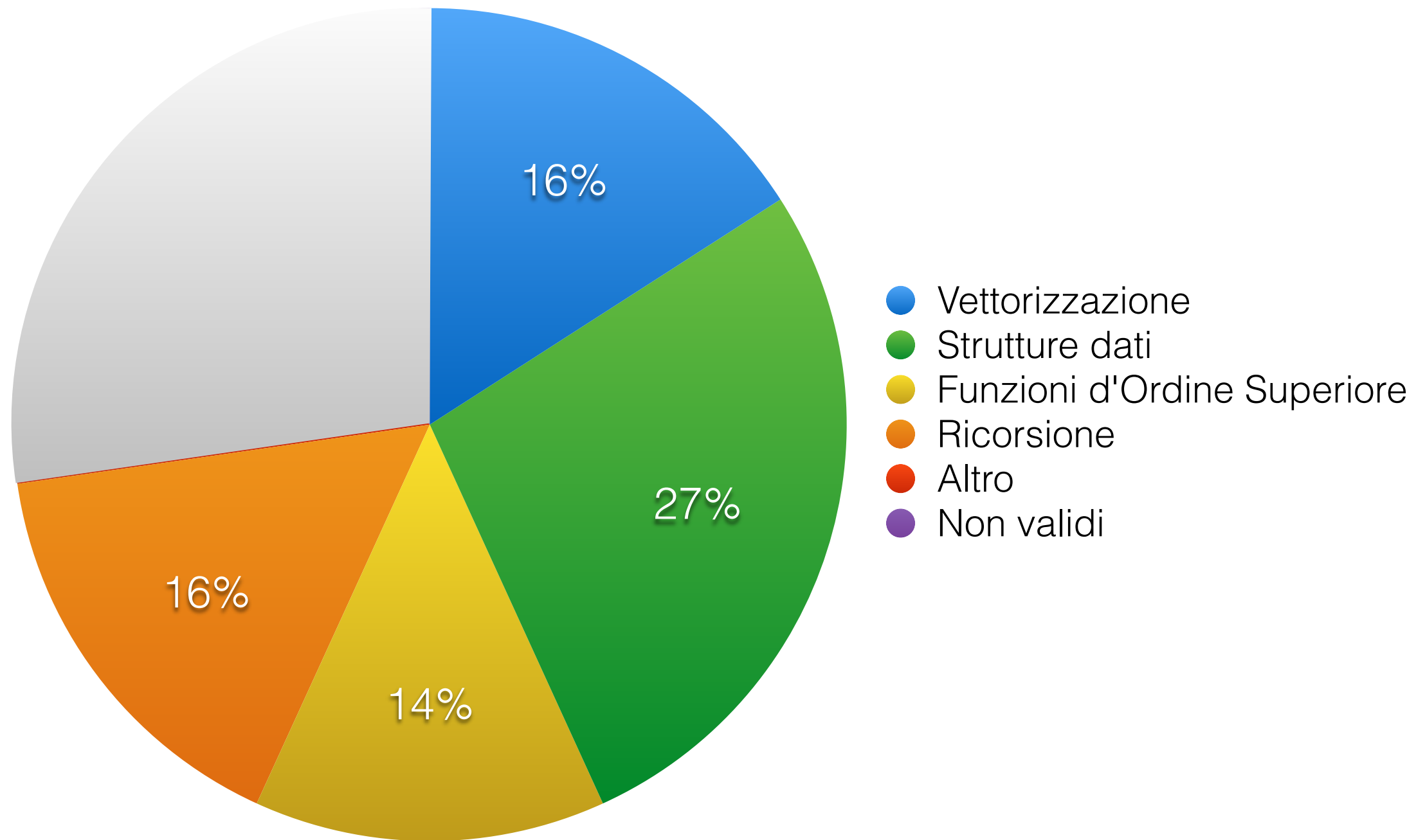
28/01/2014



11^{ma} esercitazione: feedback



11^{ma} esercitazione: feedback



11^{ma} esercitazione: agenda

- (20') Blocco 0: strutture dati
- (35') Blocco 1: ordine superiore
- (75') Blocco 2: ricorsione
- (30') Blocco 3: vettorizzazione
- (10') Question time
- Note conclusive

Blocco 0 / 1 (20')

```
% Parte 1: creare un array arrayProfessori, un array di struct
% Parte 2: ogni elemento dell'array ha i seguenti campi:
%           1) matricola
%           2) nome
%           3) secondoNome
%           4) cognome
% Parte 3: aggiungere dinamicamente il campo
%           4) corsi (una struct; non dichiariamo subito i campi)
% Parte 4: aggiungere dinamicamente a corsi i campi necessari
```

Blocco 0 / 1 (20')

CODICE 0

11^{ma} esercitazione: agenda

- (20') Blocco 0: strutture dati
- (35') Blocco 1: ordine superiore
- (75') Blocco 2: ricorsione
- (30') Blocco 3: vettorizzazione
- (10') Question time
- Note conclusive

Blocco 1 / 1 (20')

```
% Scrivere una funzione O che prenda in ingresso un vettore di interi ed una  
% funzione C, nello specifico una funzione di confronto tra due numeri.  
% La funzione C puo' confrontare i numeri come preferisce, fintanto che il  
% valore restituito (0 o 1) rappresenti la relazione di maggiore e minore  
% generalizzato, ossia la funzione C sia un ordine parziale.  
% O ordina il suo vettore di dati in ingresso nel senso della funzione C.
```


Blocco 1 / 1 (20')

CODICE 1

Blocco 1 / 2 (15')

```
% Facendo uso della funzione di ordine superiore accumulatore
% Codificare la funzione tmp(v)
%
% INPUT vettore di numeri v=[v1, ..., vn]
%
% OUTPUT Calcola e restituisce come risultato il valore:
%         PRODUTTORIA{1,n} v_i * SOMMATORIA{1,n} v_i
%
% PRODUTTORIA e SOMMATORIA vengono calcolati come termini isolati,
% ottenuti da una opportuna chiamata di una funzione, detta
% accumulatore, che prende una funzione (prodotto o somma) ed i
% limiti tra cui applicare produttoria e sommatoria, e restituisce i
% valori parziali accumulati
```

Blocco 1 / 2 (15')

CODICE 2

accumulatore.m

prod2.m

sum2.m

tmp.m

tmp2.m

11^{ma} esercitazione: agenda

- (20') Blocco 0: strutture dati
- (35') Blocco 1: ordine superiore
- (75') Blocco 2: ricorsione
- (30') Blocco 3: vettorizzazione
- (10') Question time
- Note conclusive



sottoMatrici: problema (20')



Creare una matrice di dimensione $n \times n$ che nel suo centro contiene un quadrato 2×2 che contiene il valore 1 e, andando verso l'esterno, i valori 2, 3, .. fino a $n/2$ nella cornice più esterna

Es.:

6	6	6	6	6	6	6	6	6	6	6	6
6	5	5	5	5	5	5	5	5	5	5	6
6	5	4	4	4	4	4	4	4	4	5	6
6	5	4	3	3	3	3	3	3	4	5	6
6	5	4	3	2	2	2	2	3	4	5	6
6	5	4	3	2	1	1	2	3	4	5	6
6	5	4	3	2	1	1	2	3	4	5	6
6	5	4	3	2	2	2	2	3	4	5	6
6	5	4	3	3	3	3	3	3	4	5	6
6	5	4	4	4	4	4	4	4	4	5	6
6	5	5	5	5	5	5	5	5	5	5	6
6	6	6	6	6	6	6	6	6	6	6	6



sottoMatrici: soluzione1 (20')



```
function [M] = sottoMatr(n)
```

```
if(n == 1)
```

```
    M = ones(2,2);
```

```
else
```

```
    M = sottoMatr(n-1)
```

```
    r = ones(1, size(M, 2)) * n
```

```
    M = [r; M ; r]
```

```
    c = ones(size(M, 1), 1) * n
```

```
    M = [c M c]
```

```
end
```



```
function [matrRis]=sottoMatr(n)

if n==1
    matrRis=ones(2);
else
    matrRis=n*ones(2*n);
    matrRis(2:2*n-1,2:2*n-1)=sottoMatr(n-1);
end
```

Blocco 2 / 1 (30')

```
% Ricerca per bisezione
```

```
% Effettuare una ricerca per bisezione di un valore all'interno di un array ordinato
```

```
% Il valore F é intero e l'array a é di interi.
```

```
% Metodo:
```

```
% a = [0 4 5 8 10 13 18 33 73]
```

```
% F = 4
```

```
% bisez(a, 1, length(a))
```

```
% floor((1+9)/2) = 5
```

```
% a(5) != F; a(5) > F (F si trova a "sinistra" di a in posizione 5)
```

```
%
```

```
% bisez(a, 1, 5 - 1 = 1)
```

```
% floor((4+1)/2) = floor(2,5) = 2
```

```
% a(2) == F;
```

```
% Ossia, di volta in volta cerco nel punto medio di un intervallo che si dimezza in
```

```
% lunghezza ad ogni invocazione dell'algoritmo, tenendo opportunamente la parte destra o
```

```
% sinistra a seconda che il punto medio fosse a destra o a sinistra del valore F
```

```
% RICORSIVO
```


Blocco 2 / 1 (30')

CODICE 3

Blocco 2 / 2 (10')

Funzione 91 di McCarthy

$$M(n) = \begin{cases} n - 10, & \text{se } n > 100 \\ M(M(n + 11)), & \text{se } n \leq 100 \end{cases}$$

Blocco 2 / 2 (10')

CODICE 4

Blocco 2 / 3 (15')

% AVETE UN ARRAY DI PIGNONI (STRUTTURA DATI)

% PER CONFRONTARE DUE PIGNONI USARE UNA FUNZIONE DI CONFRONTO CUSTOM (ES: SUL NUMERO DI
% DENTI DELLA RUOTA DENTATA)

% ORDINARE UN ARRAY DI PIGNONI USANDO LA FUNZIONE "O" DI CUI SOPRA, PASSANDO UN HANDLE
% OPPORTUNO; OSSIA, ORDINA RISPETTO AL NUMERO DI DENTI (DAL MINORE AL MAGGIORE)
% (TOTALE RIGHE NECESSARIE: *5*)

Blocco 2 / 3 (15')

CODICE 5

11^{ma} esercitazione: agenda

- (20') Blocco 0: strutture dati
- (35') Blocco 1: ordine superiore
- (75') Blocco 2: ricorsione
- (30') Blocco 3: vettorizzazione
- (10') Question time
- Note conclusive

Blocco 3 / 1 (20')

Si conviene di rappresentare un'immagine mediante una matrice rettangolare di numeri, corrispondenti al colore dei suoi pixel (punti luminosi che compongono la figura). Si vuole progettare una funzione MATLAB/Octave di nome `combinaImmagini` che, ricevendo come parametri due matrici $f1$ ed $f2$ rappresentanti due immagini e due valori numerici C ed S , con $C < S$, rappresentanti due diversi colori, produce come risultato una terza figura $f3$, ottenuta da $f1$ ed $f2$ secondo la seguente regola.

Nelle posizioni (r, c) in cui $f1(r, c) < C$ ed $f2(r, c) > C$ si ha $f3(r, c) = f2(r, c) - f1(r, c)$ nelle posizioni (r, c) in cui $f1(r, c) > S$ ed $f2(r, c) < S$ si ha $f3(r, c) = f1(r, c) - f2(r, c)$ nelle posizioni rimanenti si ha $f3(r, c) = f1(r, c)$.

- . a) Codificare la funzione `combinaImmagini`.
- . b) Scrivere uno script che acquisisce le due matrici di partenza rispettivamente dai file `file1.mat` e `file2.mat` (si supponga che le due variabili contenenti le matrici al momento del salvataggio si chiamino `matr1` e `matr2`), richiama la funzione `combinaImmagini` e salva sul file `file3.mat` la matrice risultante.

Blocco 3 / 1 (20')

CODICE 6

Blocco 2 / 2 (10')

Funzione 91 di McCarthy

$$M(n) = \begin{cases} n - 10, & \text{se } n > 100 \\ M(M(n + 11)), & \text{se } n \leq 100 \end{cases}$$

VETTORIZZAZIONE

Scrivetela la funzione ricorsiva in modo che possa accettare un array di valori invece di un singolo n . La funzione restituisce un valore per ogni elemento dell'array iniziale, $M(n)$ per ogni n dell'array

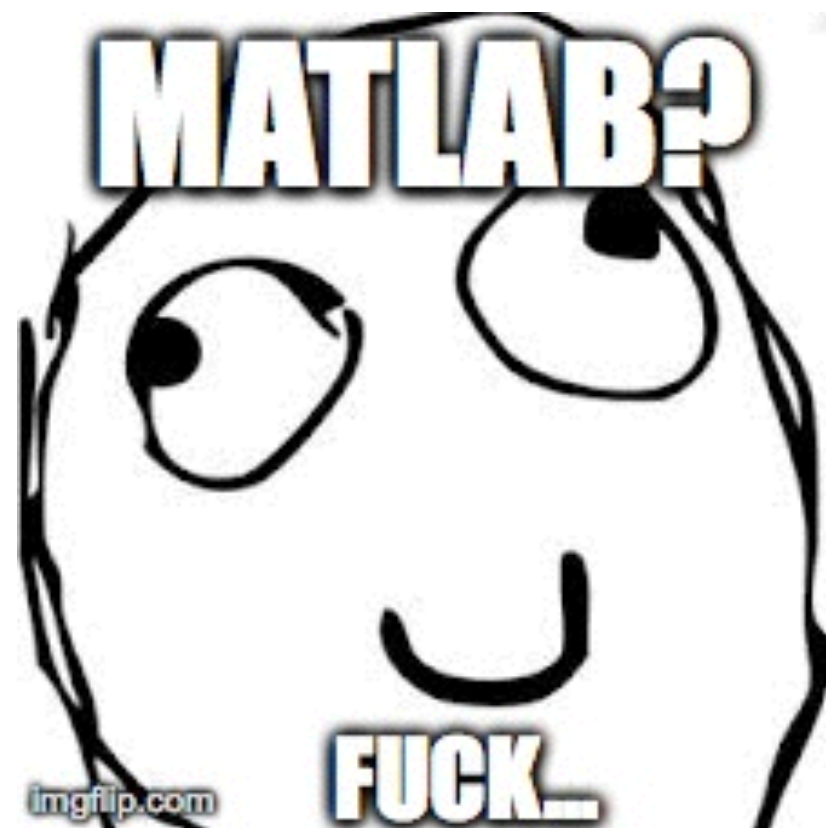
Blocco 3 / 2 (10')

CODICE 7

11^{ma} esercitazione: agenda

- (20') Blocco 0: strutture dati
- (35') Blocco 1: ordine superiore
- (75') Blocco 2: ricorsione
- (30') Blocco 3: vettorizzazione
- (10') Question time
- Note conclusive

Question Time



11^{ma} esercitazione: agenda

- (20') Blocco 0: strutture dati
- (35') Blocco 1: ordine superiore
- (75') Blocco 2: ricorsione
- (40') Blocco 3: vettorizzazione
- (10') Question time
- Note conclusive

Ultima esercitazione: done!
In bocca al lupo per l'esame :)!
Demo esame come da calendario!
Alcuni slot di exe extra: vedi calendario!

I Feedback, please :)!