9 Funzioni MATLAB

Le funzioni in un programma sono utilizzate per strutturare il codice in sottoparti e per evitare di replicare inutilmente il codice. In MATLAB le funzioni vengono identificate con la parola chiave function ed è buona regola che abbiano lo stesso nome dello script che le contiene. In generale sono strutturate nel seguente modo:

```
1 function [output1, output2, ..] = nome(input1, input2, ..)
2 %istruzioni
3 output1 = ..
4 output2 = ..
5 ...
```

dove:

- output1, output2, .. sono gli output (opzionali), che, se dichiarati, devono essere inizializzati dalla funzione
- input1, input2, .. sono gli input (opzionali) che servono per il calcolo degli output

Prima di iniziare a scrivere il corpo di una funzione, la prima cosa da stabilire è quali siano input (argomenti) e output (valori restituiti) necessari.

MATLAB vede solo le funzioni delle proprie librerie (come mean(),min()) o le funzioni dichiarate nella stessa cartella di dove viene eseguito lo script in cui si richiamano le funzioni.

Un elenco (non esaustivo) delle funzioni di MATLAB è il seguente:

```
zeros(m,n) %crea una matrice di zeri di dimensioni m*n
ones(m,n) %crea una matrice di uni di dimensioni m*n
eye(n) %crea una matrice identita' di ordine n
rand(m,n) %crea una matrice di numeri casuali in [0,1] di dimensioni m*n
randi(p,m,n) %crea una matrice di numeri interi casuali in [1,p] di dimensioni m*n

length(v) %restituisce la lunghezza del vettore v
size(M) %restituisce le dimensioni della matrice M

ceil(x) %arrotonda x all'intero superiore
floor(x) %arrotonda x all'intero inferiore
fix(x) %arrotonda x all'intero piu' vicino a 0

[M, idxM] = max(v) %restituisce il massimo M del vettore v e il suo indice idxM
[m, idxm] = min(v) %restituisce il minimo m del vettore v e il suo indice idxm

mean(v) %restituisce la media del vettore v
median(v) %restituisce la mediana del vettore v
mod(m,n) %restituisce il modulo n di m

find(p) %restituisce gli indici degli elementi che soddisfano p
```

9.1 Esercizi

Esercizio 9.1

La *radice numerica* di un numero intero è ottenuta sommando le sue cifre, poi sommando le cifre del risultato, eccetera, fino a quando non si ottiene una singola cifra.

Esempio: $65536 \to 6+5+5+3+6 = 25 \to 2+5 = 7$.

Scrivere una **funzione** che calcoli la radice numerica di un numero intero. Quindi, scrivere uno **script** che, dato un array di interi, calcola la radice numerica di ogni suo elemento.

Esercizio 9.2

Scrivere una funzione `closestVal` che prende in ingresso:

- un vettore `v`
- un valore `n`

e restituisce un valore dell'elemento di `v` più vicino a `n`, ossia a distanza minima.

Ad esempio:

```
v = [1 4 40];
n = 20;
closestVal(v, n)
```

restituisce 4, mentre

```
v = [1 4 40];
n = 32;
closestVal(v, n)
```

restituisce 40.

Utilizzare in uno script la suddetta funzione e commentare il caso in cui si debbano restituire tutti i valori a distanza minima (nel caso ce ne sia più di uno).

Esercizio 9.3

Scrivere uno script che chiede all'utente di inserire un numero positivo a (nel caso in cui il numero non sia positivo ripetere l'inserimento) e verifichi se il numero è perfetto, in caso contrario dice se è abbondante o difettivo.

Un numero è perfetto se corrisponde alla somma dei suoi divisori, escluso se stesso. Ad esempio 6 è perfetto perchè $1 + 2 + 3 = 6$. Un numero è abbondante se è minore della somma dei suoi divisori e altrimenti è difettivo. Ad esempio 20 è abbondante visto che $1 + 2 + 4 + 5 + 10 > 20$, mentre 19 (come tutti i numeri primi) è difettivo.

Richiede quindi un secondo numero b e controllare se a e b sono due numeri amici.

Due numeri a e b sono amici se la somma dei divisori di a è uguale a b e viceversa. Ad esempio 284 e 220 sono amici.

Strutturare lo script con delle funzioni:

- `inserisciPositivo` che legge un numero intero positivo;
- `sommaDivisori` che dato un numero restituisce la somma dei suoi divisori;
- `controllaSePerfetto` che restituisce se un numero è perfetto, se è abbondante o difettivo:
- `controllaSeAmici` che restituisce se due numeri sono amici.

Esercizio 9.4

Scrivere uno script che chiede all'utente di inserire due parole e che controlli se una è l'anagramma dell'altra.

Strutturare lo script scrivendo una funzione `istogramma` che legge una parola e ne restituisce l'istogramma delle frequenze delle lettere presenti nella parola.

Esercizio 9.5

(TdE 27 Giugno 2016) Si vogliono costruire in `MATLAB` opportune funzioni per gestire un vettore numerico come se fosse una coda (si pensi alla coda di persone davanti allo sportello di un ufficio pubblico). Assumendo che l'elemento 1 del vettore corrisponda al primo della coda, sviluppare le seguenti funzioni:

- `function [nuovaCoda] = accoda(coda, valore)`: restituisce una nuova coda contenente tutti gli elementi di coda più il nuovo elemento valore, che viene inserito alla fine della nuova coda;
- `function [nuovaCoda, valore] = estraiPrimo(coda)`: estrae da coda il primo valore. Restituisce la nuova coda che non contiene più il primo valore, e il valore stesso. Per esempio, se `coda = [10, 3, 4, 12, 17]`, dopo l'esecuzione della funzione si avrà che: `nuovaCoda = [3, 4, 12, 17]` e `valore = 10`;
- `function [lunghezza] = calcolaLunghezza(coda)`: restituisce lunghez-

za, che rappresenta il numero di elementi in coda.

Si considerino due vettori numerici che rappresentino due code, normale e prioritaria, definite come:

```
codaNormale = [10, 3, 4, 12, 17, 13, 68, 45, 32, 22];
codaPrioritaria = [11, 5, 6, 8, 9, 15];
```

Si ipotizzi che gli elementi di una coda siano estratti partendo dal primo valore del vettore che rappresenta la coda, e che la variabile `ultimoEstratto` indichi se l'ultimo valore estratto sia stato estratto da `codaNormale` (in questo caso `ultimoEstratto` assume il valore 0) oppure da `codaPrioritaria` (in questo caso `ultimoEstratto` assume il valore 1).

Si implementi uno script MATLAB in modo che richieda all'utente di inserire le due code e il valore di `ultimoEstratto`, quindi estragga un singolo numero da una delle due code e ne stampi a video il valore, seguendo il seguente criterio per decidere da che coda estrarre il numero:

- se l'ultimo valore è stato estratto da `codaNormale`, allora il nuovo valore sarà estratto da `codaPrioritaria`;
- se l'ultimo valore è stato estratto da `codaPrioritaria` e se la lunghezza di `codaPrioritaria` è maggiore della lunghezza di `codaNormale`/3 (opportunamente arrotondato), allora estrae il valore da `codaPrioritaria`; altrimenti, estrae il valore da `codaNormale`.

Ad esempio, se venissero inseriti i due vettori precedentemente definiti e la variabile `ultimoEstratto` avesse valore 0, il programma dovrebbe stampare a video il valore 11 e modificare il secondo vettore nel seguente modo: `codaPrioritaria = [5, 6, 8, 9, 15]`.

Per lavorare con le due code si utilizzino le funzioni definite al punto precedente. Inoltre, lo script dovrà aggiornare anche il valore delle variabili `codaNormale`, `codaPrioritaria` e `ultimoEstratto` in modo coerente con quanto definito.

Esercizio 9.6

(TdE 04 Febbraio 2016) Si implementi in linguaggio MATLAB una funzione `partition` che prende in ingresso un vettore `V` di valori numerici non ordinati, e restituisce in uscita un nuovo vettore `T` (composto dagli stessi elementi di `V`) e un valore numerico `m`, definiti nel modo seguente:

1. Se `V` ha lunghezza dispari, `m` è l'elemento in posizione centrale di `V` e il vettore `T` è composto dagli stessi elementi di `V` disposti nel seguente modo: a sinistra di `m` ci sono tutti gli elementi di `V` minori di `m`, poi compare `m` (in tutte le sue occorrenze),

quindi a destra tutti gli elementi maggiori di `m`. Ad esempio, dato il vettore `V = [6 5 3 4 3 1 2]`, la funzione restituisce il vettore `T = [3 3 1 2 4 6 5]` e il valore centrale `m = 4`:

2. Se `V` ha lunghezza pari, il valore `m` è definito come la media dei due elementi centrali di `V`, e tutti gli elementi di `V` vengono disposti in `T` in modo tale che tutti quelli minori o uguali di `m` precedono quelli maggiori di `m`. Ad esempio, dato il vettore `V = [5 6 1 2 3 4]`, la funzione restituisce il vettore `T = [1 5 6 2 3 4]` e il valore centrale `m = 1.5` (cioè la media dei due valori in posizione centrale 1 e 2).

Si scriva uno script MATLAB che svolge le seguenti operazioni:

- Acquisisce da tastiera un vettore numerico `vett` e un intero positivo `n`;
- Crea un vettore `med` vuoto;
- Svolge n volte le seguenti operazioni:
  - Invoca la funzione `partition` passando come parametro `vett`, sovrascrivendo `vett` con il vettore restituito da `partition` e memorizza nella prima posizione libera di `med` il secondo valore restituito;
  - Stampa il nuovo valore di `vett`;
- Al termine dell'esecuzione del ciclo, stampa il vettore `med`.

Esercizio 9.7

Quest'anno Babbo Natale ha deciso di farsi aiutare per la consegna dei regali da KwanzaBot [1] e Superman. Babbo natale ha un elenco di bambini (`elenco` matrice di caratteri) e deve consegnare un regalo particolare ad ognuno di essi tra piccolo (`1`), medio (`2`) e grande (`3`) a seconda di quanto sono stati buoni (`buono` vettore di interi). Per assegnare il regalo ci sono delle soglie di punteggi: fino a 700 punti il bambino ha un regalo piccolo, fino a 900 medio, altrimenti grande. Ad ogni consegna si deve stampare a video il nome del bambino, il suo regalo e chi l'ha portato.

Per dividersi il lavoro i tre hanno deciso che Babbo Natale avrebbe consegnato i regali partendo dal primo dell'elenco, KwanzaBot dal fondo e Superman scegliendo uno dei bambini a caso (`randi()`). I bambini che hanno il regalo grande lasciano anche dei biscotti a chi consegna il regalo, quindi ringrazia riportando il suo apprezzamento per i biscotti (stampandolo a video).

Strutturare lo script che consegni i regali a tutti i bambini in funzioni che dividano in maniera logica l'azione di selezione del regalo (`selezionaRegalo`), la consegna del

[1] `http://futurama.wikia.com/wiki/Kwanzaabot`

regalo (`consegnaRegalo`) e l'eliminazione dall'elenco di chi ha già ricevuto il regalo (`cancellaBambino`).

Soluzioni

Soluzione dell'esercizio 9.1

```
function somma = f_somma_cifre(n)
% somma le cifre di un intero positivo

    somma = 0;
    while n > 0
        somma = somma + mod(n, 10);
        n = floor(n / 10);
    end
end
```

```
function radice = f_radice_numerica(n)
% calcolo della radice numerica

    % elimina segno e parte decimale
    radice = abs(round(n));
    % ciclo calcolo radice
    while radice > 9
        radice = f_somma_cifre(radice);
    end
end
```

```
clear
clc

n = input('inserire un numero: ');

% chiamo funzione di calcolo radice numerica
radnum = f_radice_numerica(n);

disp(['la sua radice numerica è: ' num2str(radnum)]);
```

Soluzione dell'esercizio 9.2

```
clc
clear
close all

%% Sezione 1
```

```
v = [1 4 40];
n = 20;
res = closestVal(v, n);
res

n = 32;
res = closestVal(v, n);
res

%% Sezione 2
vett = [10 13 -5 7 5];
valore = 0;

res = closestVal(vett, valore);
res

%% Sezione 3
vett = [10 13 -5 7 5];
valore = 0;

res = closestVal2(vett, valore);
res
```

```
function [val] = closestVal(v, n)
% CLOSESTVAL prende in ingresso un vettore  v ed un valore n e
% restituisce il valore di v più vicino a n

dist = abs(v - n);
[~, pos] = min(dist);
val = v(pos);
```

```
function [val] = closestVal2(v, n)
% CLOSESTVAL prende in ingresso un vettore  v ed un valore n e
% restituisce i valori di v più vicini a n

dist = abs(v - n);
pos = dist == min(dist);
val = v(pos);
```

Soluzione dell'esercizio 9.3

```
clc
clear
```

```
3
4  % richiedere numero
5  n = inserisciPositivo();
6
7  [perf, abb, dif] = controllaSePerfetto(n);
8
9  if(perf == 1)
10     disp([num2str(n), ' e'' perfetto']);
11 else
12     disp([num2str(n), ' NON e'' perfetto']);
13     if(abb == 1)
14         disp([num2str(n), ' e'' abbondante']);
15     else
16         disp([num2str(n), ' e'' difettivo']);
17     end
18 end
19
20 m = inserisciPositivo();
21
22 amici = controllaSeAmici(n,m);
23
24 if(amici)
25     disp([num2str(n), ' e ', num2str(m), ' sono amici'])
26 else
27     disp([num2str(n), ' e ', num2str(m), ' NON sono amici'])
28 end
```

```
1  function n = inserisciPositivo()
2
3  n = -1;
4  while(n < 0)
5      n = input('Inserire un numero positivo ');
6  end
```

```
1  function s = sommaDivisori(n)
2
3  D = 1 : 1 : n / 2;
4  A = mod(n, D);
5  s = sum(D(A == 0));
```

```
1  function s = sommaDivisoriCLike(n)
2
3  v = [];
```

```
for ii = 1 : n /2
    if(mod(n, ii) == 0)
        v = [v, ii];
    end
end
s = sum(v);
```

```
function [perf, abb, dif] = controllaSePerfetto(n)

if(n == sommaDivisori(n))
    perf = 1;
    abb = 0;
    dif = 0;
elseif(n < sommaDivisori(n))
    perf = 0;
    abb = 1;
    dif = 0;
else
    perf = 0;
    abb = 0;
    dif = 1;
end
```

```
function res = controllaSeAmici(n, m)

if (sommaDivisori(n) == m && sommaDivisori(m) == n)
    res = 1;
else
    res = 0;
end
```

Soluzione dell'esercizio 9.4

```
clear
clc


parola1 = input('Inserire la prima parola: ','s');
parola2 = input('Inserire la seconda parola: ','s');

ist1 = istogramma(parola1);
ist2 = istogramma(parola2);

```

```
if all(ist1 == ist2)
%if sum(ist1 == ist2) == length(ist1)
    disp('Le due parole sono una l''anagramma dell''altra');
else
    disp('Le due parole non sono una l''anagramma dell''altra')
       ;
end
```

```
function v = istogramma(parola)

v = zeros(128,1);
for i = parola
    v(i) = v(i) + 1;
end
```

Soluzione dell'esercizio 9.5

```
clear
clc

codaNormale = input('Inserire la coda normale: ');
codaPrioritaria = input('Inserire la coda prioritaria: ');
ultimoEstratto = input('Inserire tipo di ultimo numero estratto
    (0 normale, 1 prioritario): ');

if ultimoEstratto == 0 || calcolaLunghezza(codaPrioritaria) >
   calcolaLunghezza(codaNormale)/3
    [codaPrioritaria, v] = estraiPrimo(codaPrioritaria);
    ultimoEstratto = 1;
else
    [codaNormale, v] = estraiPrimo(codaNormale);
    ultimoEstratto = 0;
end

disp(['Numero estratto: ' num2str(v)]);
```

```
function [nuovaCoda] = accoda(coda, valore)

nuovaCoda = coda;
nuovaCoda(end+1) = valore;
```

```
function [nuovaCoda, valore] = estraiPrimo(coda)

```

```
nuovaCoda = coda;
valore = nuovaCoda(1);
nuovaCoda(1) = [];
```

```
function [lunghezza] = calcolaLunghezza(coda)

lunghezza = length(coda);
```

Soluzione dell'esercizio 9.6

```
clear;
clc;
close all;

vett = input('Inserisci un vettore numerico: ');
n = input('Inserisci un valore intero positivo: ');

med = [];
for k = 1:n
        [vett, med(k)] = partition(vett);
        disp(vett);
end

disp('Sequenza dei valori centrali: ');
disp(med);
```

```
function [T, m] = partition(V)

lung = length(V);
if (lung == 0)
        T = V;
        m = 0;
else
        if mod(lung, 2) == 0
                m = mean([V(lung / 2), V(lung / 2 + 1)]);
        else
                m = V(ceil(lung / 2));
        end
        T = [V(V < m) V(V == m) V(V > m)];
end
```

Soluzione dell'esercizio 9.7

```
clear
clc
close all

%%
load('dati_babbo_natale');
n_rimasti = size(elenco,1);

while n_rimasti > 0

    % Babbo Natale
    regaloBN = selezionaRegalo(buono,1);
    consegnaRegalo(elenco, 1, regaloBN, 'Babbo Natale');
    [elenco, buono] = cancellaBambino(elenco,buono,1);
    n_rimasti = n_rimasti - 1;

    % KwanzaBot
    if n_rimasti > 0
        regaloQB = selezionaRegalo(buono,n_rimasti);
        consegnaRegalo(elenco, n_rimasti, regaloQB, 'KwanzaBot'
            );
        [elenco, buono] = cancellaBambino(elenco, buono,
            n_rimasti);
        n_rimasti = n_rimasti - 1;
    end

    % Superman
    if n_rimasti > 0
        ind = randi(n_rimasti);
        regaloSM = selezionaRegalo(buono,ind);
        consegnaRegalo(elenco, ind, regaloSM, 'Superman');
        [elenco, buono] = cancellaBambino(elenco, buono, ind);
        n_rimasti = n_rimasti - 1;
    end
    pause();
end
```

```
function regalo = selezionaRegalo(buono, ind)


if buono(ind) < 700
    regalo = 1; %regalo piccolo
elseif buono(ind) < 900
```

```
    regalo = 2; %regalo medio
else
    regalo = 3; %regalo grande
end
```

```
function consegnaRegalo(elenco, idx, regalo, identita)

fprintf(['Quest''anno ' elenco(idx,:) ' ha ricevuto un regalo
   di tipo ']);
if regalo == 1
    fprintf('piccolo');
elseif regalo == 2
    fprintf('medio');
else
    fprintf('grande\n');
    fprintf([identita ' ha gradito i biscotti']);
end
fprintf('\n');
```

```
function [elenco, buono] = cancellaBambino(elenco, buono, ind)

buono(ind) = [];
elenco(ind,:) = [];
```