Fondamenti di Informatica, A.A. 2012-2013

22/02/2014

Esercizio 1

È dato il frammento di codice Matlab

```
n = 3;
v = zeros(1,n);
w = zeros(n,1);
for i=1:n
   v(i) = i;
   w(i) = i^2;
end
a = v*w;
disp(a);
```

Si chiede cosa viene visualizzato dall'interprete Matlab, giustificando la risposta.

Svolgimento

Il vettore `v` va a contenere gli interi da 1 a 3; il vettore colonna `w` contiene i valori `[1,4,9]'`, e `a` conterrà il loro prodotto scalare che vale 36.

Esercizio 2

Descrivere le regole di visibilità delle variabili in Matlab, con particolare riguardo all'utilizzo delle funzioni.

Soluzione

Le variabili di Matlab sono normalmente visibili nel workspace in cui sono state create. Ciascuna invocazione di funzione (comprese quelle ricorsive) implica la creazinoe di un nuovo workspace che inizialmente contiene solo le variabili presenti nella lista degli argomenti ed a cui è stato associato un valore dalla funzione chiamante; pertanto ogni accesso ad una variabile si riferisce puramente a quella locale, senza altra relazione con le variabili del workspace da cui la funzione è stata invocata.

Gli argomenti delle funzioni vengono normalmente passati per valore, ossia il corpo della funzione accede (ed eventualmente modifica) una copia dei valori delle variabili nel workspace del chiamante; eventuali modifiche rimangono quindi confinate alla funzione stessa.

La parola chiave `global` può essere usata per specificare esplicitamente una associazione diretta tra una variabile di una funzione ed una del workspace di partenza; questa associazione deve essere resa esplicita sia nello spazio di partenza che nella funzione.

Esercizio 3

Un esercizio richiede di riconoscere in un array la più lunga sottosequenza di numeri non decrescenti. Il seguente codice contiene due errori:

```
maxl=0;
l=1;
for i=2:length(v)
    if (v(i-1)<v(i))
      l=l+1;
    else
      if (l>maxl)
        maxl=l;
        l=1;
      end
    end
end
```

Lo si modifichi in modo da realizzare correttamente la funzione richiesta.

Svolgimento

Il primo errore del codice è che viene richiesto di riconoscere sequenze non decrescenti, ma si controlla che gli elementi siano in ordine strettamente crescente; pertanto il confronto deve essere fatto per `(v(i-1)<=v(i))`. Il secondo problema è che non viene tenuto conto dell'ultima sottosequenza all'uscita dal vettore; al limite, se il vettore è già ordinato, il risultato viene 0 invece che `length(v)`. Occorre però anche controllare che il vettore non sia vuoto, altrimenti il risultato corretto è effettivamente 0. Il codice corretto è pertanto

```
maxl=0;
l=1;
for i=2:length(v)
    if (v(i-1)<=v(i))
      l=l+1;
    else
      if (l>maxl)
        maxl=l;
        l=1;
      end
    end
end
if (lenvgth(v)>0)&&(l>maxl)
  maxl=l;
end
```

Esercizio 4

È data la seguente funzione Matlab

```
function [x]=mistery(a,x,y)
  n=size(a,1);
  k=length(y);
  for i=1:k
    alpha = a(i,1:k)*a(1:k,i);
```

```
    a(i:i+k-1,i:i+k-1) = a(i:i+k-1,i:i+k-1) - alpha*x*y';
  end
end
```

Stimare il numero di operazioni aritmetiche ipotizzando che a sia una matrice $n \times n$ e x, y siano dei vettori $k \times 1$. Quale ipotesi è necessario porre sulle dimensioni affinché il codice non dia errore?

Svolgimento

Il codice si compone di un ciclo ripetuto k volte, e le varie iterazioni sono tutte uguali tra loro per quanto riguarda il numero di operazioni aritmetiche effettuate; tutte le iterazioni operano su array di dimensione $k \times k$. La prima istruzione comporta un prodotto scalare (riga per colonna) quindi di costo $2k$. La seconda operazione comporta la formazione del prodotto `alpha*x*y'` e k^2 addizioni nell'aggiornamento della sottomatrice `a(i:i+k-1,i:i+k-1)`. Per quanto riguarda il prodotto `alpha*x*y'`, il suo costo dipende dall'ordine con cui viene effettuato: se infatti si fa `(alpha*x)*y'` si scala un vettore per uno scalare (k moltiplicazioni) e poi si fa il prodotto esterno di due vettori (k^2 moltiplicazioni); se invece si effettua prima il prodotto dei due vettori, allora il costo sarà di $2k^2$ operazioni. Usando l'alternativa più economica, avremo:

$$k(2k + k + k^2)) = k(3k + k^2) = O(k^3);$$

affinché il codice possa funzionare correttamente occorre che nella indicizzazione della matrice `a` non ci siano mai errori, e quindi occorre che il massimo valore possibile per gli indici di riga e colonna sia non superiore ad n; in definitiva

$$k + k - 1 \leq n \Rightarrow 2k \leq n + 1.$$

Esercizio 5

È dato il frammento di codice Matlab

```
n=length(v)
i=1;
j=n
while (i<=j)
   ti=v(i)^2;
   tj=v(j)^2;
   v(i)=tj;
   v(j)=ti
   i=i+1;
   j=j-1;
end
```

Riscrivere il codice facendo uso del ciclo `for` ovvero di operazioni su array.

Svolgimento

Il ciclo eleva al quadrato gli elementi del vettore e li dispone in ordine inverso. La soluzione più economica usa il linguaggio degli array come segue

```
v = v(end:-1:1).^2;
```

utilizzando un ciclo `for` invece si può scrivere

```
n=length(v);
n2=floor(n/2);
for j=1:n2
    ti=v(n-j+1)^2;
    tj=v(j)^2;
    v(n-j+1)=tj;
    v(j)=ti;
end
if (2*n2<n)
   v(n2+1) = v(n2+1)^2;
end
```