Fondamenti di Informatica, A.A. 2012-2013

10/07/2013 – fila A

Esercizio 1

È dato il frammento di codice Matlab

```
mg =[  16   2   3  13;
        5  11  10   8;
        9   7   6  12;
        4  14  15   1;];
[mx,imx]=max(mg);
[mn,imn]=min(mx);
disp( mg(imx(imn),imn));
```

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

Svolgimento

L'operatore `max` applicato ad una matrice bidimensionale lavora colonna per colonna, quindi restituirà in `mx` e `imx` due vettori riga contenenti i massimi lungo ciascuna colonna e la loro posizione (indice di riga). L'operatore di `min` applicato a `mx` restituirà il più piccolo tra i massimi di colonna ed il suo indice di colonna, che corrisponde peraltro all'indice di colonna nella matrice originaria `mg`. Pertanto il codice stamperà a schermo il minimo tra i massimi di colonna, ovvero il valore 13.

Esercizio 2

Discutere dei *cell array* in Matlab e del loro utilizzo.

Svolgimento

I cell array in Matlab sono delle estensioni dei normali array in quanto consentono di memorizzare in una unica struttura dati tra loro disomogenei, che poi vengono indirizzati con un indice numerico. Il contenuto di ciascun elemento di un cell array è completamente arbitrario, quindi può essere uno scalare, un array o un altro cell array. Si distinguono dai normali array per l'uso delle parentesi graffe nella indicizzazione. In particolare, i cell array sono lo strumento necessario per memorizzare un array di stringhe; infatti, in Matlab una stringa è un array di caratteri, quindi un array di stringhe dovrebbe essere un array di caratteri bidimensionale e questo imporrebbe che tutte le stringhe fossero della stessa lunghezza, restrizione che invece non si applica al caso dei cell array.

Esercizio 3

In una prova di esame di informatica viene richiesto di scrivere una routine di ricerca sequenziale su un array. Uno studente propone il seguente codice

```
function idx=search(key,v)
  for k=1:length(v)
    if (key==v(k))
```

```
            idx=k;
            return
        end
    end
end
```

Si identifichi il problema presente nel codice e si proponga una correzione adeguata.

Svolgimento

Il problema sorge nel momento in cui la chiave cercata `key` non sia presente nel vettore `v`; in tal caso infatti l'indice di uscita `idx` non viene assegnato. Per risolvere il problema si può inizializzare

```
idx=0;
```

immediatamente prima del ciclo `for`, avendo assunto per convenzione che un valore di uscita pari a 0 (che *non* è un indice valido in Matlab) rappresenti la situazione di chiave non trovata.

Esercizio 4

È data la seguente funzione Matlab

```
function [x]=mistery(a,b,x,k)
  for i=1:k
    z = b-a*x;
    y = a*z;
    alpha = x'*y;
    x = x + alpha*z;
  end
end
```

Stimare il numero di operazioni aritmetiche ipotizzando che a sia una matrice $n \times n$ e b e x siano dei vettori $n \times 1$.

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. Le prime due istruzioni sono dei prodotti matrice-vettore, e possono essere assunte entrambe costare $2n^2$; le altre due operazioni sono un prodotto scalare e una somma di vettori, ed entrambe posso essere assunte costare $2n$. In definitiva il costo sarà

$$k(2(2n^2) + 2(2n)) \approx 4kn^2 = O(kn^2).$$

Esercizio 5

È dato il frammento di codice Matlab

```
k=n;
m=inf;
while (m>thresh)&&(k>0)
   if (v(k) <m)
      m=v(k);
   end
   k=k-3;
end
```

Riscrivere il codice facendo uso del ciclo `for`.

Svolgimento

```
m=inf;
for k=n:-3:1
   if (m>thresh)
      if (v(k) <m)
        m=v(k);
      end
   end
end
```

In alternativa si può usare la istruzione `break`

```
m=inf;
for k=n:-3:1
   if (m<=thresh)
      break;
   end
   if (v(k) <m)
     m=v(k);
   end
end
```

10/07/2013 – fila B

Esercizio 1

È dato il frammento di codice Matlab

```
mg =[  16   2   3  13;
        5  11  10   8;
        9   7   6  12;
        4  14  15   1;];
[mn,imn]=min(mg);
[mx,imx]=max(mn);
disp( mg(imn(imx),imx));
```

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

Svolgimento

L'operatore `min` applicato ad una matrice bidimensionale lavora colonna per colonna, quindi restituirà in `mn` e `imn` due vettori riga contenenti i minimi lungo ciascuna colonna e la loro posizione (indice di riga). L'operatore di `max` applicato a `mn` restituirà il più grande tra i minimi di colonna ed il suo indice di colonna, che corrisponde peraltro all'indice di colonna nella matrice originaria `mg`. Pertanto il codice stamperà a schermo il massimo tra i minimi di colonna, ovvero il valore 4.

Esercizio 2

Discutere delle *struct* in Matlab e del loro utilizzo.

Svolgimento

Le strutture in Matlab consentono di memorizzare in una unica variabile dati tra loro disomogenei, che poi vengono indirizzati con un nome simbolico (stringa). Il contenuto di ciascun elemento di una struttura è completamente arbitrario, quindi può essere uno scalare, un array o un'altra struttura. Il nome con cui si indirizzano i vari campi può essere anche costruito dinamicamente; nell'uso pratico è opportuno che l'operazione di costituzione di una struttura venga delegata ad una funzione in modo da ridurre le possibilità di errori nella identificazione dei nomi delle singole componenti.

Esercizio 3

In una prova di esame di informatica viene richiesto di scrivere una routine di ricerca binaria su un array ordinato. Uno studente propone il seguente codice

```
function res=bsearch(key,v)
  % Binary search
  k=1;
  n=length(v);
  while ((k<=n)&&(~found))
    m = floor((k+n)/2)
     if (v(m)==key)
```

```
      found = true;
      res=m
    elseif (key < v(m))
      n=m-1
    elseif (key > v(m))
      k=m+1
    end
  end
```

Si identifichi il problema presente nel codice e si proponga una correzione adeguata.

Svolgimento

Il codice presenta due problemi: la variabile `found` non viene inizializzata a `false`, e qualora la chiave cercata `key` non sia presente nel vettore `v`, l'indice di uscita `idx` non viene assegnato. In definitiva occorre aggiungere le due inizializzazioni

```
idx=0;
found=0;
```

immediatamente prima del ciclo `while`, avendo assunto per convenzione che un valore di uscita pari a 0 (che *non* è un indice valido in Matlab) rappresenti la situazione di chiave non trovata.

Esercizio 4

È data la seguente funzione Matlab

```
function [x]=mistery(a,x,y)
  n=size(a,1);
  k=length(x);
  for i=1:k
    alpha = max(max(a(1:k,1:k)));
    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 k^2 confronti, mentre la seconda comporta $2k^2$ moltiplicazioni nella formazione del prodotto `alpha*x*y'` e k^2 addizioni nell'aggiornamento della sottomatrice `a(i:i+k-1,i:i+k-1)`. Pertanto il costo sarà

$$k(k^2 + k^2 + 2k^2)) = 4k^3 = O(k^3);$$

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

```
mx=0;
p=0; q=0;
for i = 1:4:n
  for j=n:-1:1
      if (mx > thresh)
          break;
      end
      alpha=abs(a(i,j));
      if (alpha>mx)
          p=i; q=j;
          mx=alpha;
      end
  end
end
```

Riscriverlo usando dei cicli `while` ed evitando la istruzione `break`.

Svolgimento

```
mx=0;
p=0; q=0;
i=1;
while (i<=n)
  j=n;
  while (mx<=thresh)&&(j>=1)
      alpha=abs(a(i,j));
      if (alpha>mx)
          p=i; q=j;
          mx=alpha;
      end
      j=j-1;
  end
  i=i+4;
end
```