Fondamenti di Informatica, A.A. 2013-2014

22/09/2014

Esercizio 1

È dato il frammento di codice Matlab

```
v = [1 3 2 4];
n=4
m = 1;
x=4;
for i=1:n
   v(i) = v(i)^3;
   if x(i)<v(i)
        x=[x max(v)];
   else
        x=[x x(i)+1];
   end
end
disp(x);
```

Si chiede cosa viene visualizzato dall'interprete Matlab.

Soluzione

Il vettore x viene creato nel ciclo for ed il risultato sarà:

```
>> x =
     4     5
>> x =
     4     5    27
>> x =
     4     5    27    28
>> x =
     4     5    27    28    64
```

Esercizio 2

Commentare le righe del blocco di codice Matlab/Octave che segue. Dare una spiegazione esaustiva sia dei comandi e funzioni utilizzate che degli output prodotti. Nello specifico, se si volesse impostare un range fisso per l'asse delle ordinate mediante il comando `axis([XMIN XMAX YMIN YMAX])` quale è il valore più opportuno per `YMIN` e `YMAX`.

```
        subplot(2, 3, 4)
        N = 16;
        xr = linspace(0, 1, N);
        yr = xr + rand(1,N) - 0.5;
        plot(xr, yr, 'r+')
        title('Symbols for Real Data')
```

Soluzione

```
        %Crea una finestra che conterrà 6
        %grafici distribuiti su 2 righe e 3 colonne. Il terzo
        %parametro di valore 4 indica che il grafico
        %successivamente creato verrà visualizzato nel quarto
        %quadrante, ovvero nel quadrante in posizione seconda
        %riga prima colonna.
        subplot(2, 3, 4)
        N = 16;
        %Crea un  vettore xr di 16
        %elementi il cui valore è compreso tra 0 e 1
        xr = linspace(0, 1, N);
        %Crea un vettore yr di 16 elementi il cui valore è
        %compreso tra -0.5 ed 1.5
        yr = xr + rand(1,N) - 0.5;
        %yr può' essere visto come un vettore contenente 16 valori
        %di una funzione di xr ovvero yr=f(xr). Il comando plot
        %grafica il vettore yr rispetto ad xr, ovvero interpreta
        %yr come i valori di una funzione di xr
        plot(xr, yr, 'r+')
        %assegna un titolo al grafico
        title('Symbols for Real Data')
```

I valori di `YMIN` ed saranno gli estremi dell'intervallo di valori assunti da `yr`. Essendo `xr` compreso tra 0 e 1, `yr` sarà compreso tra `YMIN=-0.5` e `YMAX=1.5`

Esercizio 3

Il codice riportato ha un difetto. Identificarlo e proporre una soluzione.

```
function result = rfunct(N)
    if (N==1)||(N==2)
        result = 1;
    else
        result = rfunct(N-1) + rfunct(N-2);
    end
```

Soluzione

Il problema sta nel mancato controllo dell'input; nel caso in cui l'utente passi un valore intero `N<=0`, ovvero un valore non intero, non viene mai trovata la condizione di terminazione della ricorsione, e l'interprete va in errore. Una soluzione possibile è

```
function result = rfunct(N)

    if (N<1)
        result=0;
    elseif (N==1)||(N==2)
        result = 1;
```

```
else
    result = rfunct(N-1) + rfunct(N-2);
end
```

Esercizio 4

È data la seguente funzione Matlab, dove `H` è una matrice quadrata:

```
function [H]=hesslsq(H)
  n=size(H,1);
  for j=1:n-1
     Q = givens(H(j,j),H(j+1,j));
     H(j:j+1,j:end) = Q*H(j:j+1,j:end);
 end
```

Si stimi il numero di operazioni aritmetiche eseguite, sapendo che la funzione `givens` restituisce una opportuna matrice 2×2; si assuma che il costo di ciascuna invocazione della funzione `givens` sia una costante C_g.

Soluzione

A ciascun passo viene eseguita la funzione `givens` che come da testo ha un costo fisso, e successivamente viene effettuato il prodotto di una matrice 2×2 per la matrice `H(j:j+1,j:end)`, la quale ha dimensione $2 \times (n - j + 1)$ essendo H quadrata. Il prodotto di due matrici $m \times k$ e $k \times n$ costa $2mnk$ operazioni; pertanto il costo sarà

$$\begin{aligned}\sum_{j=1}^{n-1}(C_g + 2 \cdot (2 \cdot 2 \cdot (n-j+1)) &= (n-1)C_g + 8 \cdot \sum_{j=2}^{n} j = (n-1)C_g + 8 \cdot ((\sum_{j=1}^{n} j) - 1) \\ &= (n-1)C_g + 8\frac{n(n+1)}{2} - 8 = 4n^2 + (C_g + 4)n - 8\end{aligned}$$

avendo osservato che quando j varia da 1 a $n-1$, allora $n - j + 1$ varia tra 2 e n.

Esercizio 5

È dato il frammento di codice Matlab

```
v=[2  4  6  7];
  n=length(v);
  i=n; j=1
  while j <= n
    v(i)=v(i)+v(j);
    i = i - 1;
    j=n-i+1
  end
```

Riscrivere il codice usando un ciclo `for` ed una sola variabile.

Soluzione

```
v=[2  4  6  7];
for  i=length(v):-1:1
          v(i)=v(i)+v(end -i+1);
end
```

Fondamenti di Informatica, A.A. 2013-2014

22/09/2014

Esercizio 1

È dato il frammento di codice Matlab

```
n=4
v = [1 3 2 4];
m = 1;
x=4;
for i=n:-1:1
   v(i) = v(i)^3;
   if x(n-i+1)>v(i)
        x=[x min(v)];
   else
        x=[x x(n-i+1)+1];
   end
end
disp(x);
```

Si chiede cosa viene visualizzato dall'interprete Matlab.

Soluzione

Il vettore x viene creato nel ciclo for ed il risultato sarà:

```
>> x =
     4     5
>> x =
     4     5     6
>> x =
     4     5     6     7
>> x =
     4     5     6     7     1
```

Esercizio 2

Commentare le righe del blocco di codice Matlab/Octave che segue. Dare una spiegazione esaustiva sia dei comandi e funzioni utilizzate che degli output prodotti.

```
        subplot(2, 3, 5)
        x = 0:0.01:20;
        y1 = 20*exp(-0.05*x).*sin(x);
        y2 = 0.8*exp(-0.5*x).*sin(3*x);
        plotyy(x,y1,x,y2,'plot');
        xlabel('Zero to 20 usec.');
________title('plotting on the Second Axis');
```

Soluzione

```
%Crea una finestra che conterrà 6
%grafici distribuiti su 2 righe e 3 colonne. Il terzo
%parametro di valore 5 indica che il grafico
%successivamente creato verrà visualizzato nel quarto
%quadrante, ovvero nel quadrante in posizione seconda
%riga seconda colonna.
subplot(2, 3, 5)
%genera un vettore x contenente 2001 elementi (20/0.01 +1 )
%il cui valore è compreso tra 0 e 20.
x = 0:0.01:20;
%Genera un vettore y1=f1(x). size(y1)=size(x).
y1 = 20*exp(-0.05*x).*sin(x);
%Genera un vettore y2=f2(x). size(y2)=size(x)
y2 = 0.8*exp(-0.5*x).*sin(3*x);
%come si può notare dalle funzioni f1 e f2 y1 ed y2 assumono
%valori che differiscono per 2 ordini di grandezza. La funzione
%plotyy permette di visualizzare una doppia scala sull'asse
%delle y. In particolare sull'asse di sinistra visualizzarà i
%valori di y1 e sull'asse di destra quelli di y2.
plotyy(x,y1,x,y2,'plot');
%assegna un etichetta all'asse delle x
xlabel('Zero to 20 \musec.');
%assegna un titolo al grafico
title('plotting on the Second Axis');
```

Esercizio 3

Il codice seguente dovrebbe implementare una funzione di ricerca lineare di un valore `x` in un vettore `V` di lunghezza `n`. La versione proposta contiene un errore. Identificarlo e proporre una soluzione corretta.

```
function [ r ] = SeqSearch( V,i,j,x )

if i>=j
    r=0;
    return;
elseif V(i) == x
    r=i;
    return;
else
    r=SeqSearch(V, i+1, j, x);
    return;
end
end
```

Soluzione

La funzione così come proposta non riconosce correttamente la situazione in cui la chiave `x` si trovi nell'ultima posizione del vettore `V`. La correzione consiste nel modificare la prima condizione di uscita come segue

```
function [ r ] = SeqSearch( V,i,j,x )

if i>j
    r=0;
    return;
elseif V(i) == x
    r=i;
    return;
else
    r=SeqSearch(V, i+1, j, x);
    return;
end
end
```

Esercizio 4

È data la seguente funzione Matlab, dove `beta` è un vettore e `V` una matrice rettangolare:

```
function [W,Y]=block(beta, V)
  [n,r]=size(V);
  if (n ~= length(beta))
    error("Sizes must match ");
  end
  Y=V(:,1);
  W=-beta(1)*V(:,1);
  for j=2:r
      z = -beta(j)*((W*Y')*V(:,j));
      W = [W,z];
      Y = [Y, V(:,j)];
 end
```

Si stimi il numero di operazioni aritmetiche eseguite.

Soluzione

All'interno del ciclo l'operazione `z = -beta(j)*((W*Y')*V(:,j));` comporta (considerando l'ordine di precedenza imposto con le parentesi) per prima cosa il prodotto di `W` e `Y'`. Al primo passo ($j = 2$) questi sono vettori di dimensione n; durante la prima iterazione a ciascuna delle variabili viene aggiunta una colonna, così che al secondo passo sono matrici $n \times 2$, al terzo passo $n \times 3$ e così via. Dato che `Y` viene trasposta, si ha sempre coerenza nelle dimensioni del prodotto di matrici; ricordiamo che il prodotto di due matrici di dimensioni $m \times k$ e $k \times n$ ha un costo $2mnk$. Nel caso in questione le dimensioni sono $n \times (j-1)$ e

$(j-1) \times n$, quindi il risultato è una matrice $n \times n$. Pertanto si ha un costo relativo al solo prodotto `W*Y'` di

$$\sum_{j=2}^{r} 2 \cdot n \cdot (j-1) \cdot n = 2n^2 \sum_{j=1}^{r-1} j = 2n^2 \frac{r(r-1)}{2} = n^2 r(r-1)$$

A seguito poi si ha il prodotto della matrice risultante con il vettore `V(:,j)` del costo fisso $2n^2$ ed un prodotto per uno scalare `beta(j)` dal costo n, entrambi ripetuti $(r-1)$ volte; il prodotto per lo scalare viene effettuato anche una volta fuori dal ciclo. In totale il costo è quindi

$$opcnt = n^2 r(r-1) + 2(r-1)n^2 + (r-1)n + n \approx r^2 n^2$$

Esercizio 5

È dato il frammento di codice Matlab

```
if length(s1)~=length(s2)
    r=false;
else
    r=true;
    for i=1:length(s1)
        if s1(i)~=s2(i)
            r=false;
            break;
        end
    end
end
```

Proporre una soluzione che non faccia uso ne di cicli `for` ne di cicli `while`.

Soluzione

```
if length(s1)~=length(s2)
    r=false;
else
    r=sum(s1==s2)==length(s1);
end
```