Docente: **Matteo Re**

UNIVERSITÀ DEGLI STUDI DI MILANO

C.d.l. Informatica

Bioinformatica

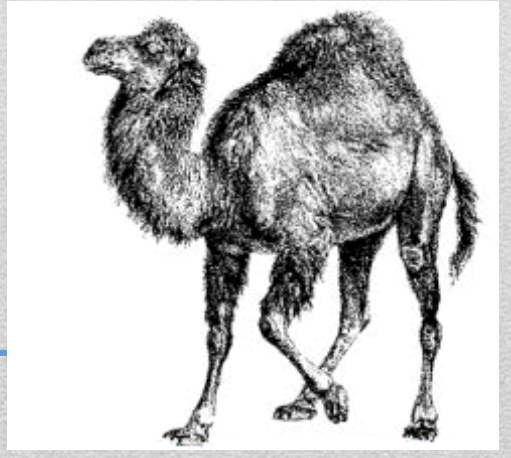
A.A. 2013-2014 semestre I

p4

UPGMA

Obiettivi

- **Clustering gerarchico in PERL**
  - Implementazione di un algoritmo di clustering
  - Utilizzo di matrici
  - Subroutines
- **Biologia computazionale**
  - Costruzione alberi filogenetici
  - Implementazione metodo **UPGMA**



Linee guida

- **Il livello di complessità di questa esercitazione è medio**
  - Cercate di risolvere il problema dopo averlo suddiviso in sottoproblemi
  - Indipendentemente dal fatto che lo script Perl funzioni o meno l'esercizio **NON verrà valutato** se, insieme allo script, non verrà inviato anche lo pseudocodice.

- **Modalità di svolgimento dell'esercitazione:**
  - Scaricare dal sito web del corso il file UPGMA.pl ed il file di input inputmatrix.txt
  - Questo script è **incompleto**
  - La posizione delle parti da aggiungere è evidenziata da questo commento:

########## description # FILL IN ####################

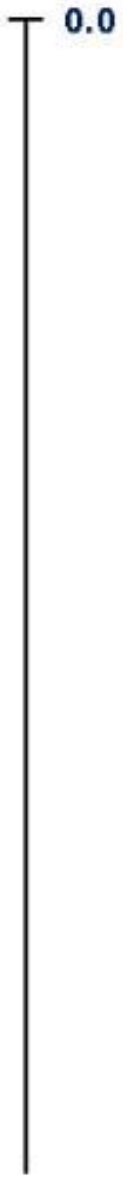
description: descrizione dell'operazione da svolgere

  - Alcune operazioni sono indicate con **OPT** … esse riguardano, principalmente, operazioni che servono per migliorare l'output dello script e che lo rendono più leggibile. Sono parti opzionali. Se le realizzate avrete dei punti in più per la soluzione dell'esercizio.

	A	B	C	D	E	F	G
A							
B	19.00						
C	27.00	31.00					
D	8.00	18.00	26.00				
E	33.00	36.00	41.00	31.00			
F	18.00	1.00	32.00	17.00	35.00		
G	13.00	13.00	29.00	14.00	28.00	12.00	

Input:

Matrice distanze



UPGMA:

Unweighted Pair-Group Method with Arithmetic mean

Unweighted:

Tutte le distanze pairwise danno lo stesso contributo

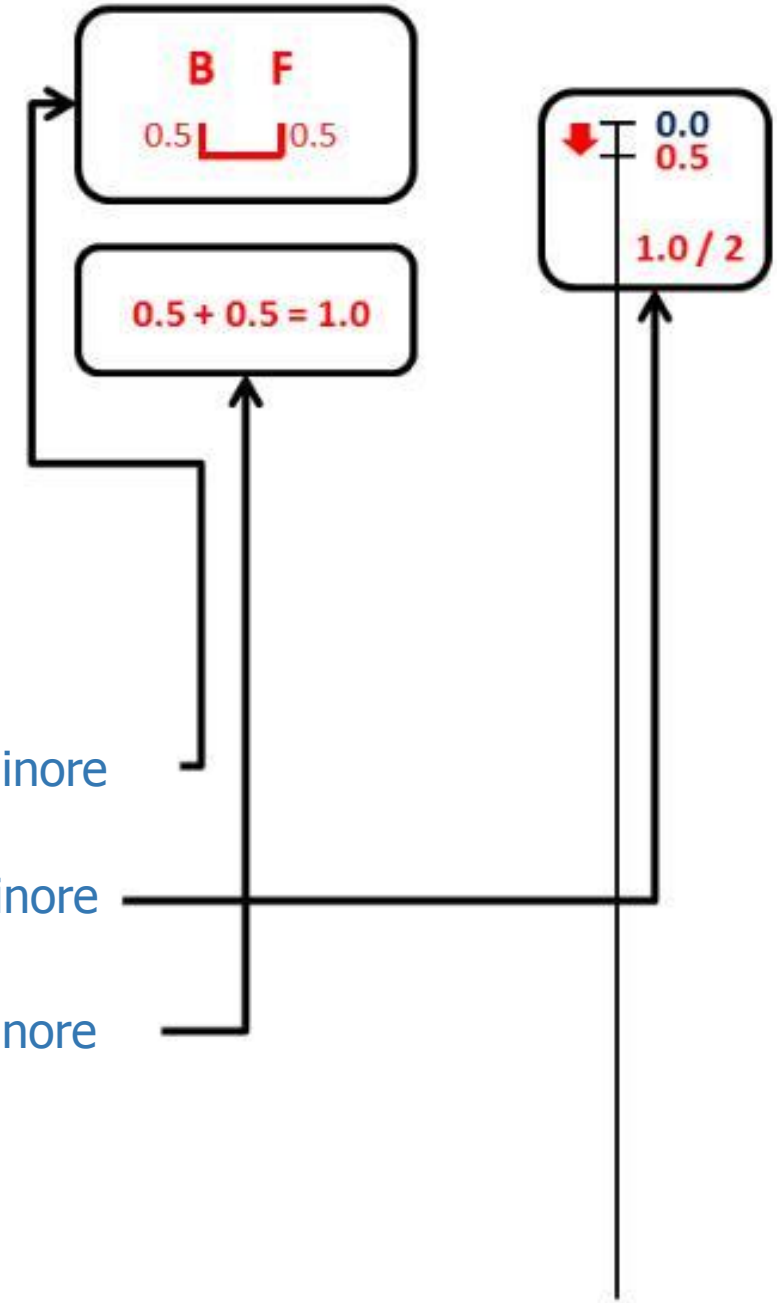
Pair-Group:

Gruppi vengono generati combinati a coppie

Arithmetic mean:

Distanze pairwise di ogni gruppo (cladi) sono distanze medie da tutti membri del gruppo.

	A	B	C	D	E	F	G
A							
B	19.00						
C	27.00	31.00					
D	8.00	18.00	26.00				
E	33.00	36.00	41.00	31.00			
F	18.00	1.00	32.00	17.00	35.00		
G	13.00	13.00	29.00	14.00	28.00	12.00	



1. Trovare la distanza minore
2. Unire le due seq./gruppi con distanza minore
3. Profondità = ½ dist. minore
4. Distanza da foglia a foglia = dist. minore

	A	B	C	D	E	F	G
A							
B	19.00						
C	27.00	31.00					
D	8.00	18.00	26.00				
E	33.00	36.00	41.00	31.00			
F	18.00	13.00	32.00	17.00	35.00		
G	13.00	13.00	29.00	14.00	28.00	12.00	

(19 + 18) / 2 = 18.5

(31 + 32) / 2 = 31.5

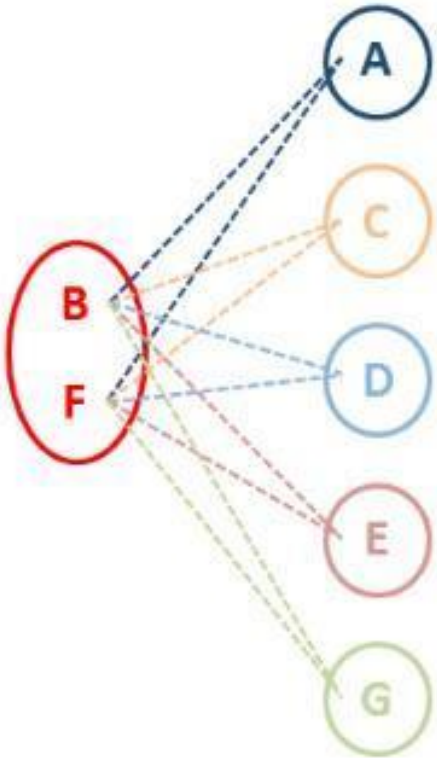
	A	BF	C	D	E	G
A						
BF	18.50					
C	27.00	31.50				
D	8.00	17.50	26.00			
E	33.00	35.50	41.00	31.00		
G	13.00	12.50	29.00	14.00	28.00	

(18 + 17) / 2 = 17.5

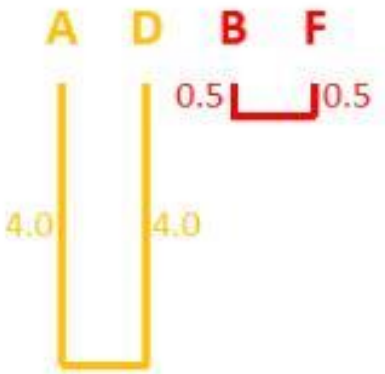
(36 + 35) / 2 = 35.5

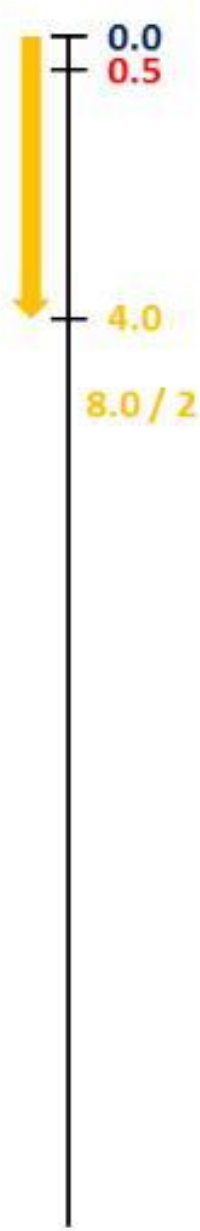
(13 + 12) / 2 = 12.5

5. Calcolare le distanze **medie** tra le due seq. del gruppo e le altre sequenze.



	A	B	C	D	E	F	G
A							
B	19.00						
C	27.00	31.00					
D	8.00	18.00	26.00				
E	33.00	36.00	41.00	31.00			
F	18.00	1.00	32.00	17.00	35.00		
G	13.00	13.00	29.00	14.00	28.00	12.00	





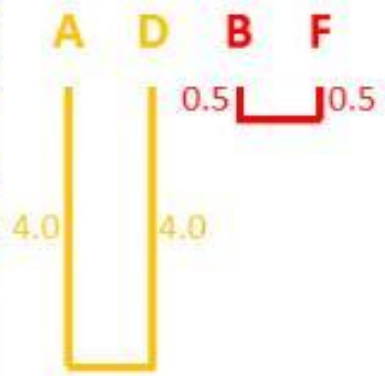
4.0 + 4.0 = 8.0

8.0 / 2

	A	BF	C	D	E	G
A						
BF	18.50					
C	27.00	31.50				
D	8.00	17.50	26.00			
E	33.00	35.50	41.00	31.00		
G	13.00	12.50	29.00	14.00	28.00	

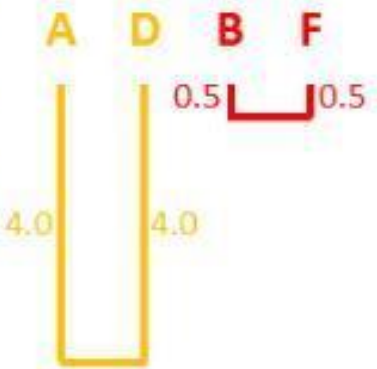
6. Ripetere l'intero ciclo partendo dalla nuova distanza minore

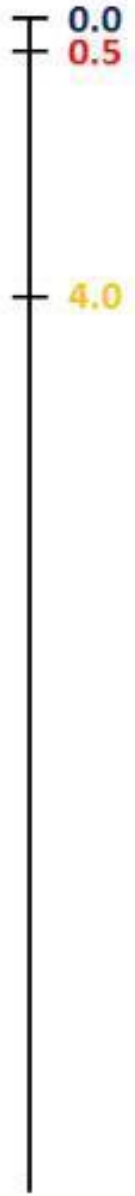
	A	B	C	D	E	F	G
A							
B	19.00						
C	27.00	31.00					
D	8.00	18.00	26.00				
E	33.00	36.00	41.00	31.00			
F	18.00	1.00	32.00	17.00	35.00		
G	13.00	13.00	29.00	14.00	28.00	12.00	



	A	BF	C	D	E	G
A						
BF	18.50					
C	27.00	31.50				
D	8.00	17.50	26.00			
E	33.00	35.50	41.00	31.00		
G	13.00	12.50	29.00	14.00	28.00	

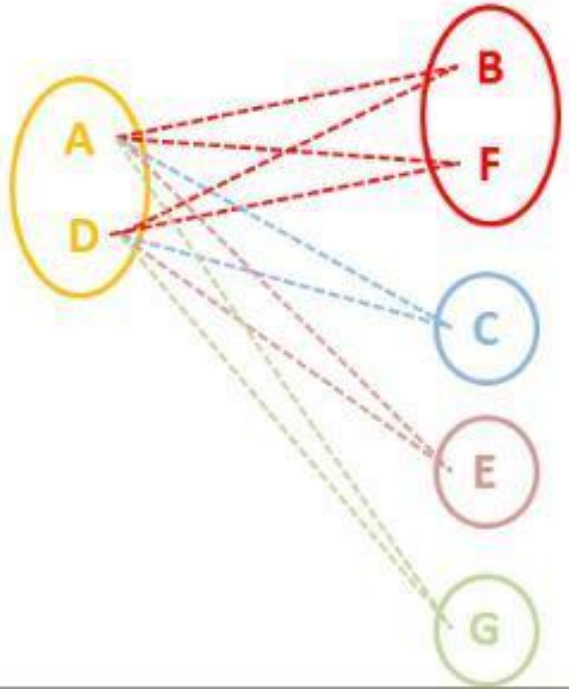
	A	B	C	D	E	F	G
A							
B	19.00						
C	27.00	31.00					
D	8.00	18.00	26.00				
E	33.00	36.00	41.00	31.00			
F	18.00	1.00	32.00	17.00	35.00		
G	13.00	13.00	29.00	14.00	28.00	12.00	





(19 + 18 + 18 + 17) / 4 = 18.0

	AD	BF	C	E	G
AD					
BF	18.00				
C	26.50	31.50			
E	32.00	35.50	41.00		
G	13.50	12.50	29.00	28.00	



(27 + 26) / 2 = 26.5

(33 + 31) / 2 = 32.0

(13 + 14) / 2 = 13.5

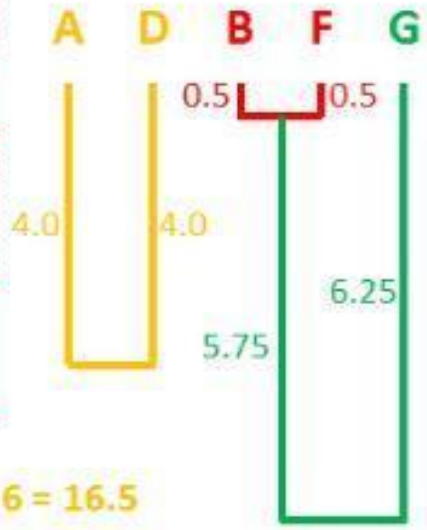
	A	B	C	D	E	F	G
A							
B	19.00						
C	27.00	31.00					
D	8.00	18.00	26.00				
E	33.00	36.00	41.00	31.00			
F	18.00	1.00	32.00	17.00	35.00		
G	13.00	13.00	29.00	14.00	28.00	12.00	

	AD	BF	C	E	G
AD					
BF	18.00				
C	26.50	31.50			
E	32.00	35.50	41.00		
G	13.50	12.50	29.00	28.00	

0.5 + 5.75 + 6.25 = 12.5

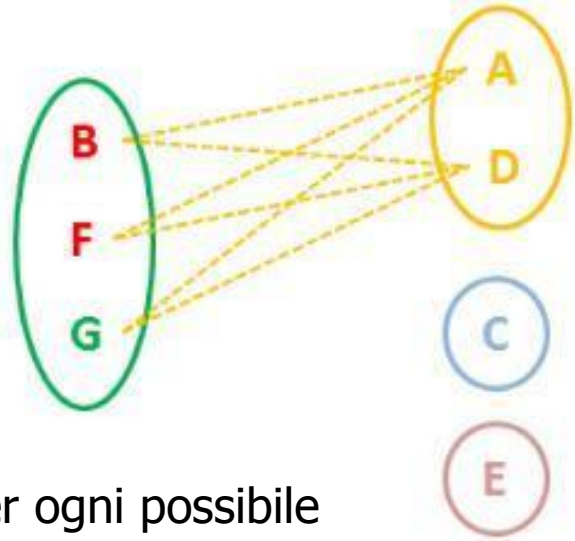
12.5 / 2

	A	B	C	D	E	F	G
A							
B	19.00						
C	27.00	31.00					
D	8.00	18.00	26.00				
E	33.00	36.00	41.00	31.00			
F	18.00	1.00	32.00	17.00	35.00		
G	13.00	13.00	29.00	14.00	28.00	12.00	



(19 + 18 + 13 + 18 + 17 + 14) / 6 = 16.5

	AD	BFG	C	E
AD				
BFG	16.50			
C	26.50	30.67		
E	32.00	33.00	41.00	

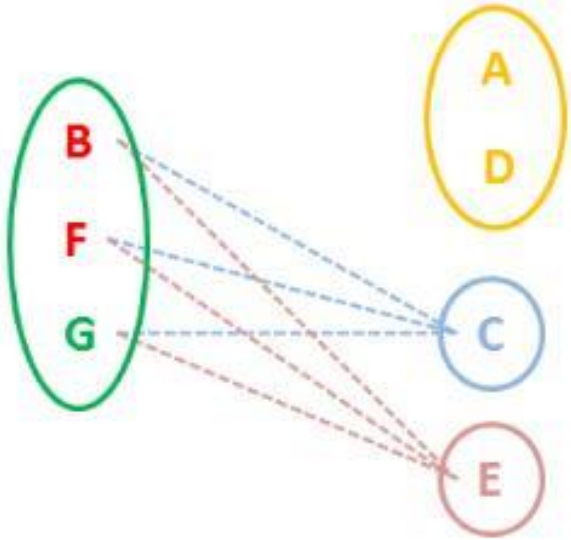


Le nuove distanze sono valori medi per ogni possibile distanza pairwise tra i membri dei gruppi

	A	B	C	D	E	F	G
A							
B	19.00						
C	27.00	31.00					
D	8.00	18.00	26.00				
E	33.00	36.00	41.00	31.00			
F	18.00	1.00	32.00	17.00	35.00		
G	13.00	13.00	29.00	14.00	28.00	12.00	

(19 + 18 + 13 + 18 + 17 + 14) / 6 = 16.5

	AD	BFG	C	E
AD				
BFG	16.50			
C	26.50	30.67		
E	32.00	33.00	41.00	

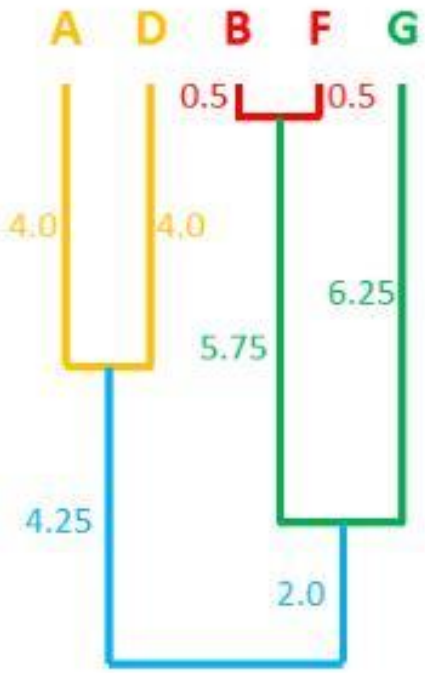


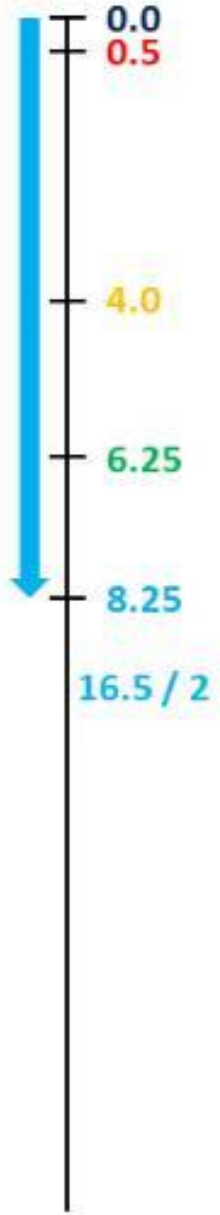
(31 + 32 + 29) / 3 = 30.67

(36 + 35 + 28) / 3 = 33.0

	A	B	C	D	E	F	G
A							
B	19.00						
C	27.00	31.00					
D	8.00	18.00	26.00				
E	33.00	36.00	41.00	31.00			
F	18.00	1.00	32.00	17.00	35.00		
G	13.00	13.00	29.00	14.00	28.00	12.00	

	AD	BFG	C	E
AD				
BFG	16.50			
C	26.50	30.67		
E	32.00	33.00	41.00	





0.5 + 5.75 + 4.25 = 16.5

4.0 + 4.25 +

6.25 + 2.0 = 16.5

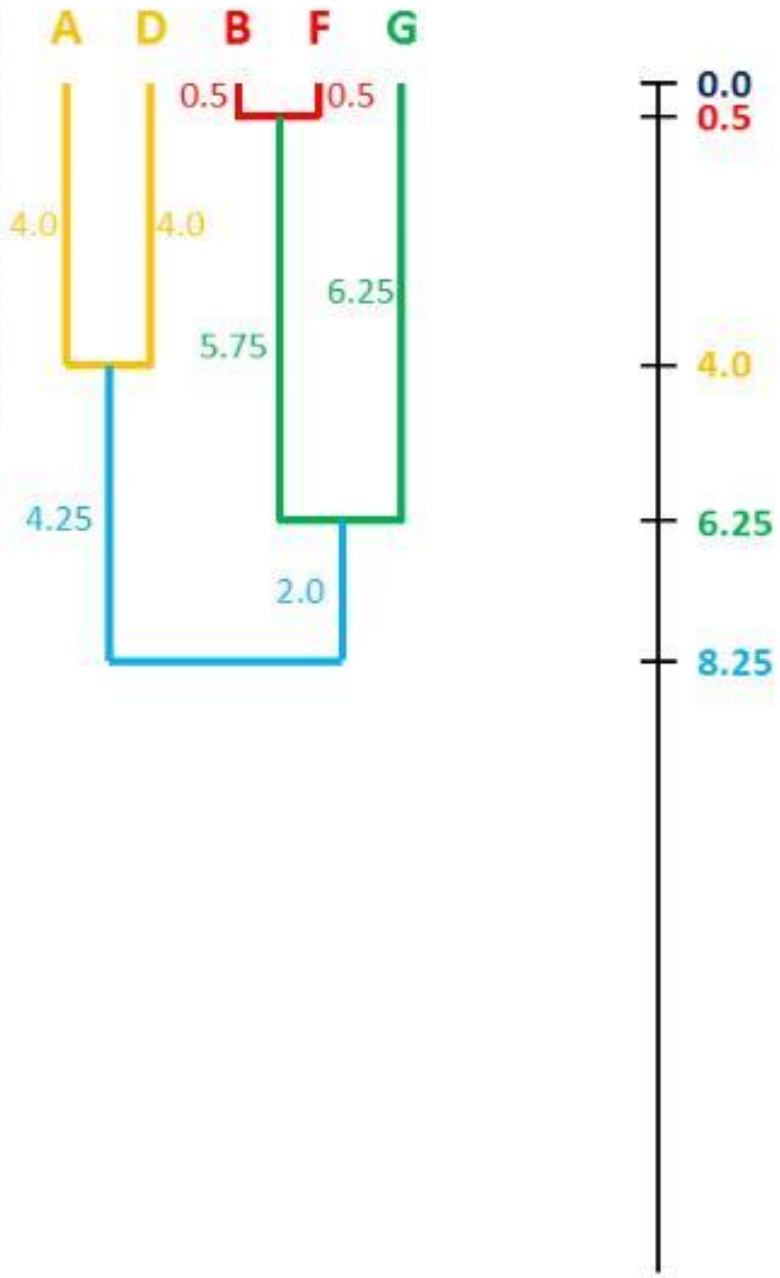
16.5 / 2

	A	B	C	D	E	F	G
A							
B	19.00						
C	27.00	31.00					
D	8.00	18.00	26.00				
E	33.00	36.00	41.00	31.00			
F	18.00	1.00	32.00	17.00	35.00		
G	13.00	13.00	29.00	14.00	28.00	12.00	

(27 + 31 + 26 + 32 + 29) / 5 = 29.00

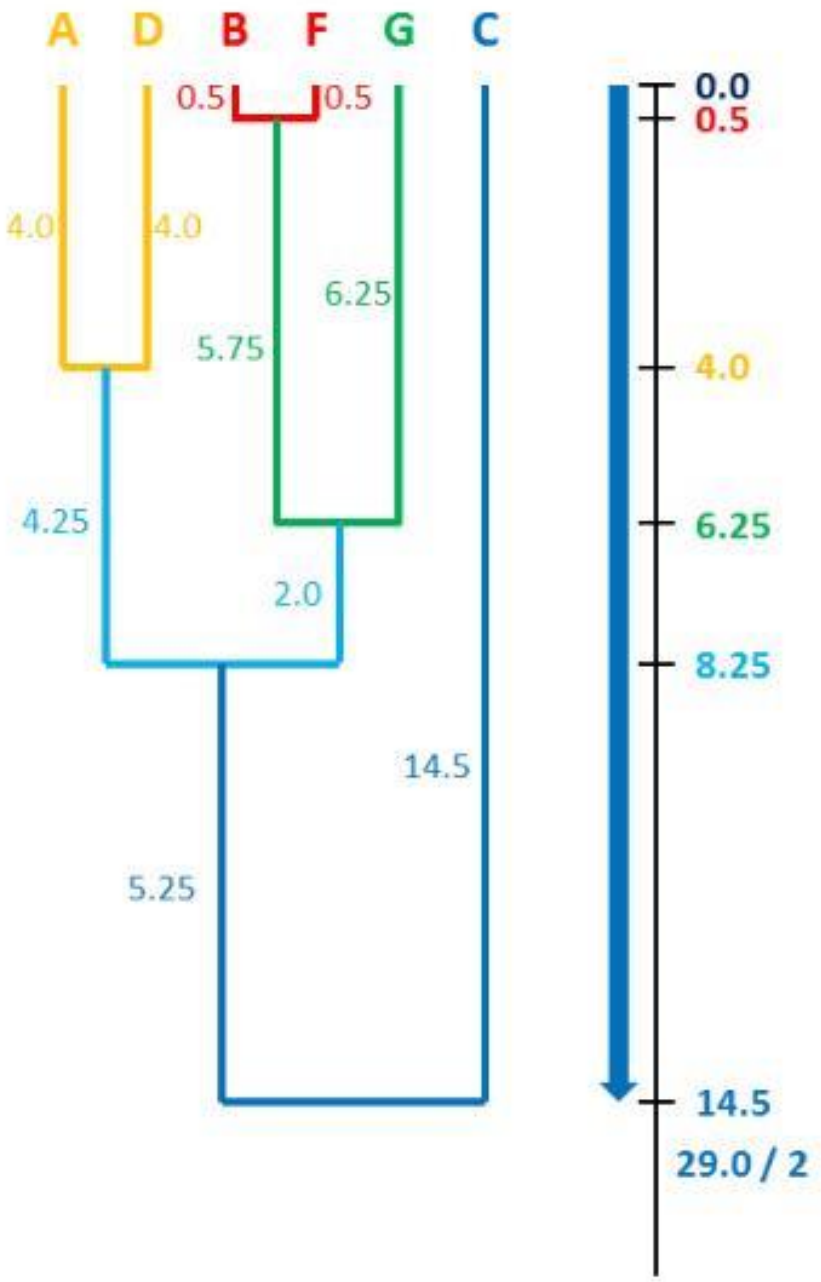
	ADBFG	C	E
ADBFG			
C	29.00		
E	32.60	41.00	

(33 + 36 + 31 + 35 + 28) / 5 = 32.60



	A	B	C	D	E	F	G
A							
B	19.00						
C	27.00	31.00					
D	8.00	18.00	26.00				
E	33.00	36.00	41.00	31.00			
F	18.00	1.00	32.00	17.00	35.00		
G	13.00	13.00	29.00	14.00	28.00	12.00	

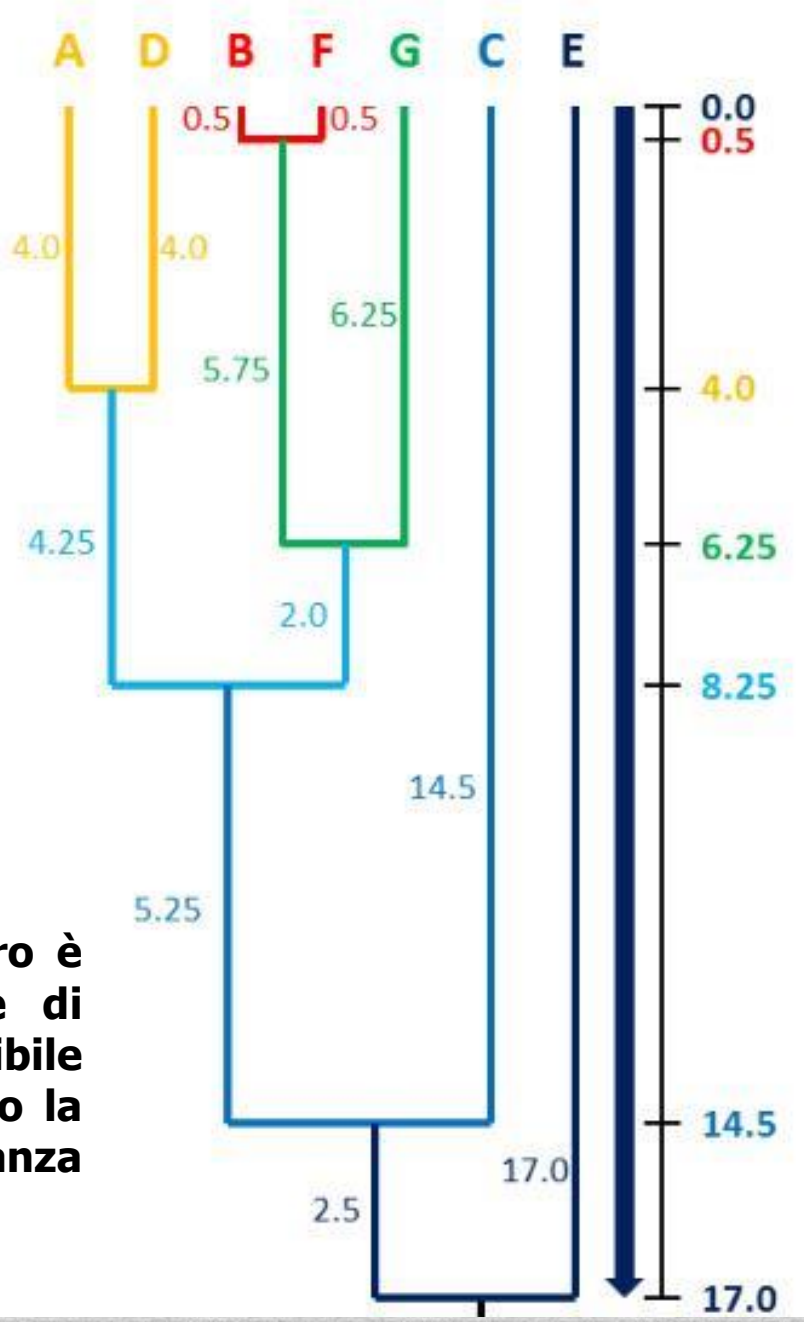
	ADBFG	C	E
ADBFG			
C	**29.00**		
E	32.60	41.00	



	A	B	C	D	E	F	G
A							
B	19.00						
C	27.00	31.00					
D	8.00	18.00	26.00				
E	33.00	36.00	41.00	31.00			
F	18.00	1.00	32.00	17.00	35.00		
G	13.00	13.00	29.00	14.00	28.00	12.00	

(33 + 36 + 41 +31 + 35 + 28) / 6 = 34.00

	ADBFGC	E
ADBFGC		
E	34.00	



UPGMA assume l'orologio molecolare. L'albero è radicato rispetto al raggruppamento finale di cladi. Tutti i percorsi che uniscono ogni possibile coppia di foglie passando per la radice hanno la stessa distanza che corrisponde alla distanza media finale.

	A	B	C	D	E	F	G
A							
B	19.00						
C	27.00	31.00					
D	8.00	18.00	26.00				
E	33.00	36.00	41.00	31.00			
F	18.00	**1.00**	32.00	17.00	35.00		
G	13.00	13.00	29.00	14.00	28.00	12.00	

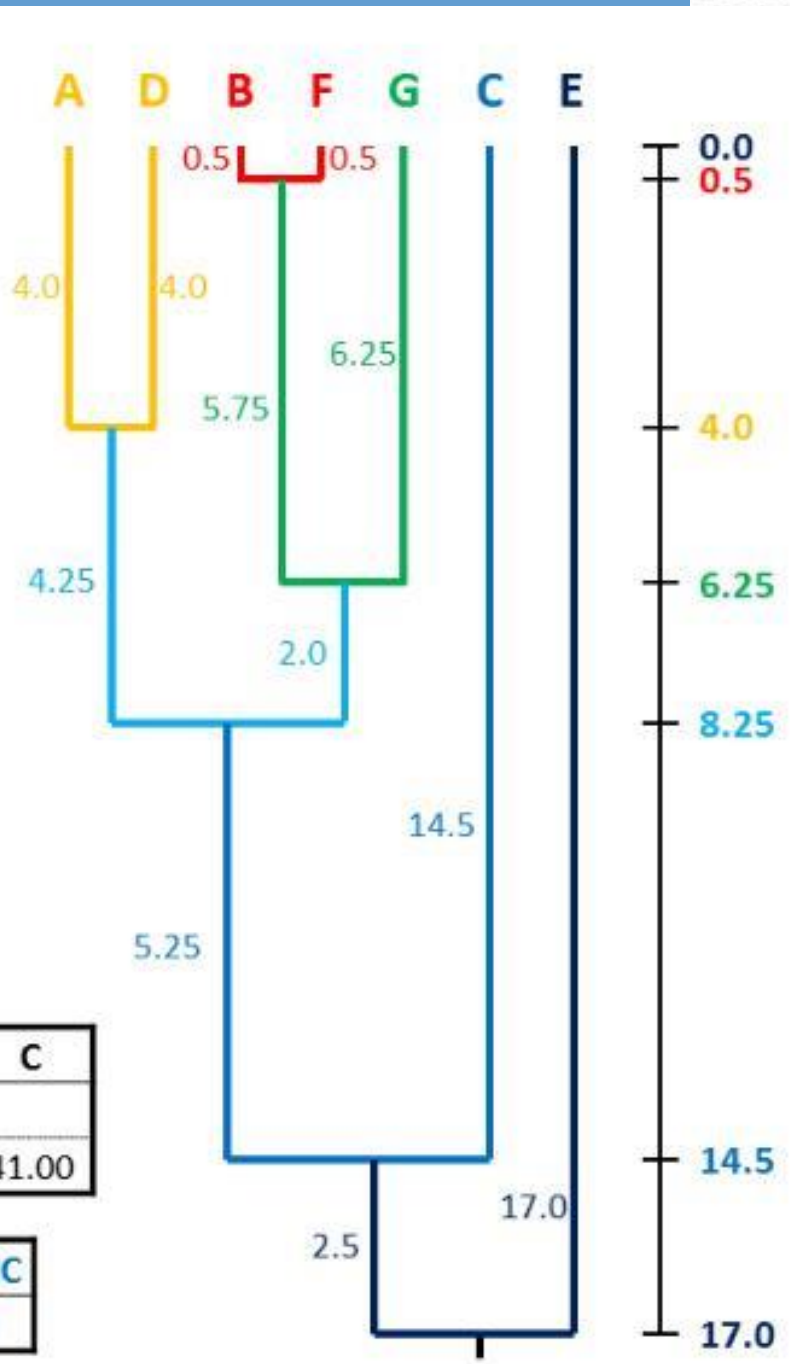
	A	BF	C	D	E
BF	18.50				
C	27.00	31.50			
D	8.00	17.50	26.00		
E	33.00	35.50	41.00	31.00	
G	13.00	12.50	29.00	14.00	28.00

	AD	BF	C	E
BF	18.00			
C	26.50	31.50		
E	32.00	35.50	41.00	
G	13.50	**12.50**	29.00	28.00

	AD	BFG	C
BFG	16.50		
C	26.50	30.67	
E	32.00	33.00	41.00

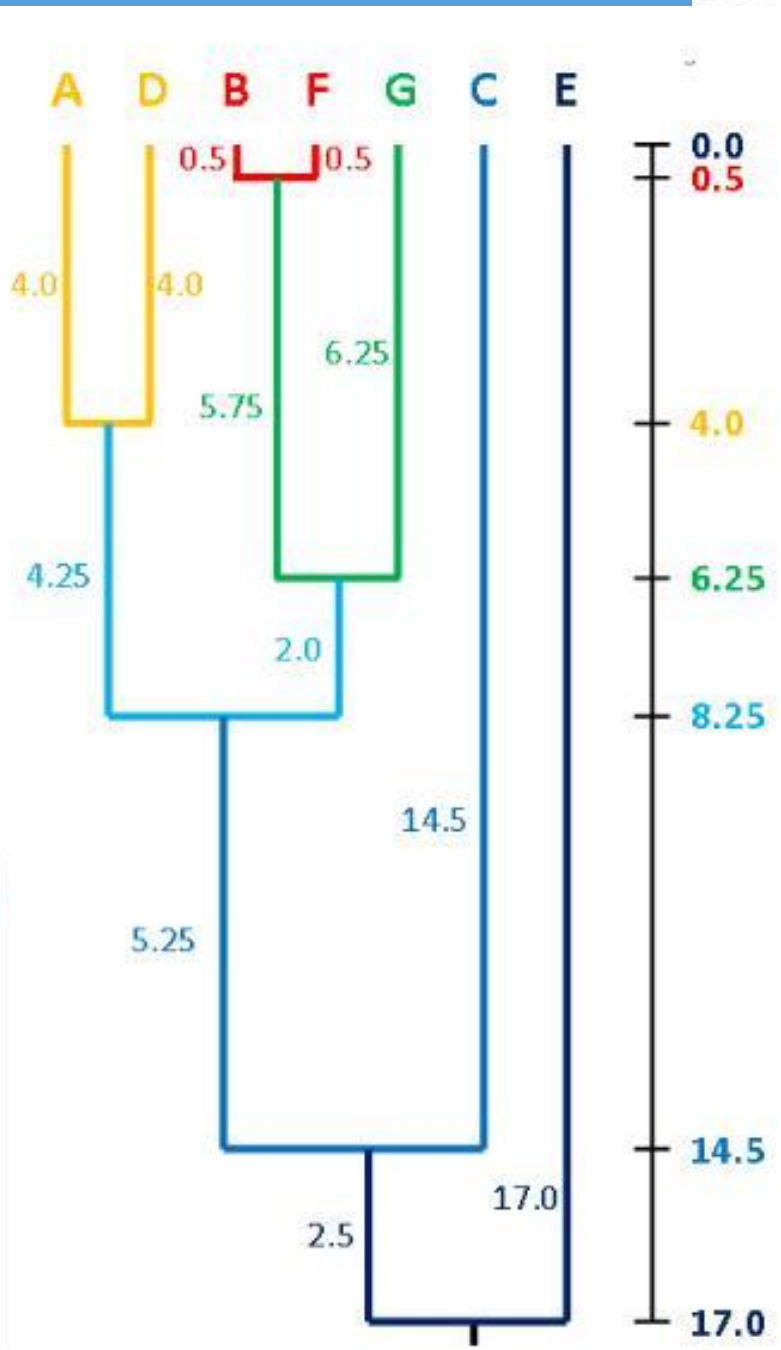
	ADBFG	C
C	**29.00**	
E	32.60	41.00

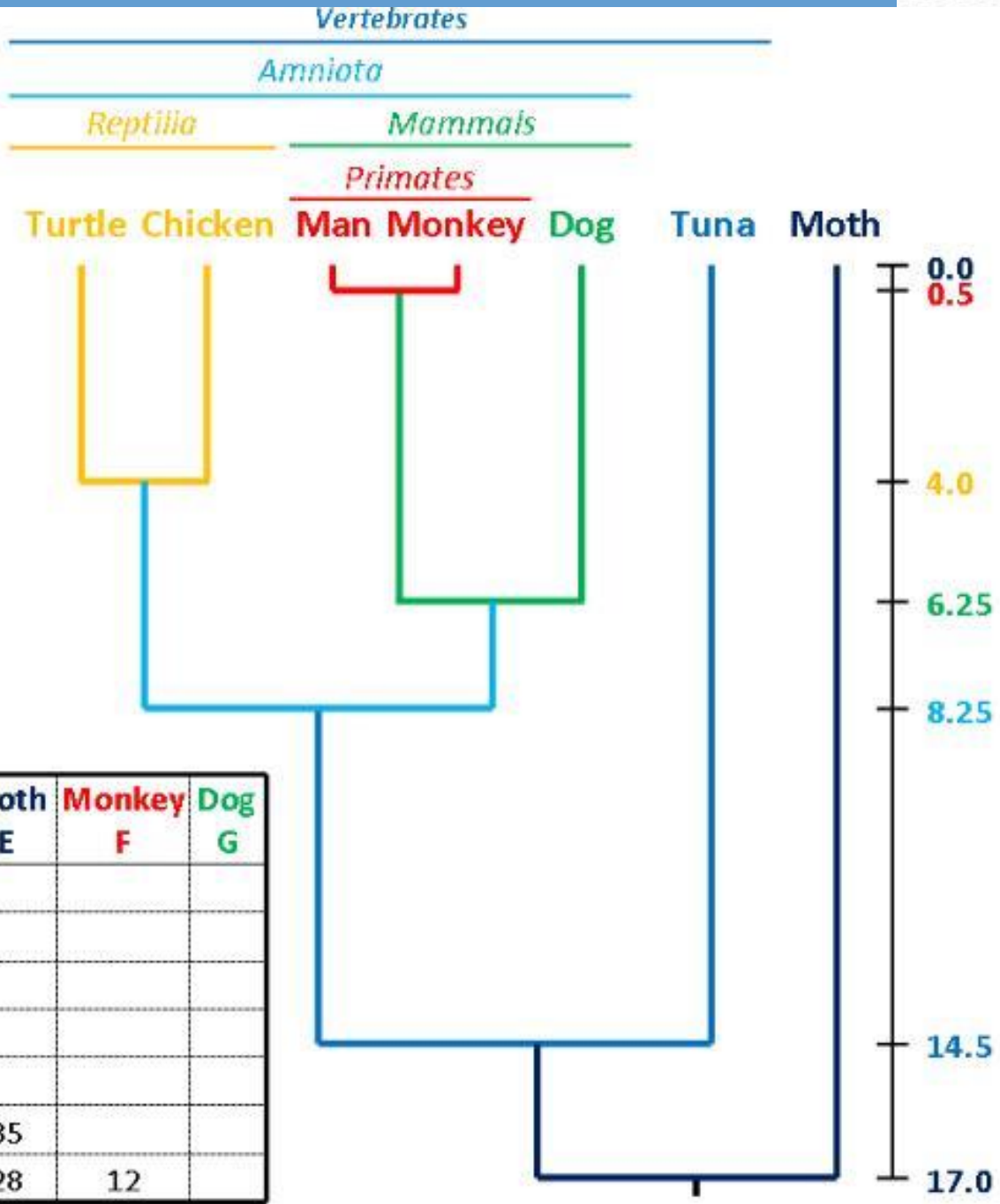
	ADBFGC
E	**34.00**



	A	B	C	D	E	F	G
A							
B	19.00						
C	27.00	31.00					
D	8.00	18.00	26.00				
E	33.00	36.00	41.00	31.00			
F	18.00	1.00	32.00	17.00	35.00		
G	13.00	13.00	29.00	14.00	28.00	12.00	

	Turtle A	Man B	Tuna C	Chicken D	Moth E	Monkey F	Dog G
Turtle							
Man	19						
Tuna	27	31					
Chicken	8	18	26				
Moth	33	36	41	31			
Monkey	18	1	32	17	35		
Dog	13	13	29	14	28	12	





	Turtle A	Man B	Tuna C	Chicken D	Moth E	Monkey F	Dog G
Turtle							
Man	19						
Tuna	27	31					
Chicken	8	18	26				
Moth	33	36	41	31			
Monkey	18	1	32	17	35		
Dog	13	13	29	14	28	12	

Funzioni in Perl

- Molto semplici da relizzare
- Si basano sull'utilizzo della chiave **sub**
- Tutto il corpo della funzione è racchiuso tra parentesi **graffe**
- Ci sono due modi di "restituire" il risultato/i:
  - 1) Scrivere il risultato in una variabile **globale** dichiarata all'esterno della funzione con il modificatore di visibilità **my**
  - 2) Restituire al chiamante il risultato mediante la chiave **return**
- I parametri della funzione si acquisiscono in vari modi … il più semplice è l'utilizzo della chiave **shift** esattamente come quando recuperiamo gli argomenti dello script (oppure possiamo lavorare su variabili globali).

```
sub somma{
    $a = shift;
    $b = shift;
    $c = $a + $b;
    return $c;
}
```

NB: la funzione va scritta nello script **PRIMA** del suo utilizzo !

```
$test = somma(2,3);
```

Realizzazione UPGMA

1) Acquisizione dati della matrice di distanze (leggi da file o da tastiera)
2) Trova distanza minima (definisce coppia di elementi!)
3) Raggruppamento dei due elementi
4) Calcolo delle distanze **tra** la coppia di elementi **e tutti gli elementi rimanenti**
5) Creazione di una nuova matrice in cui I due elementi appena raggruppati appaiono come unica entità (alternativa: **modifica** matrice esistente)
6) Salvataggio in una stringa dell'evento di raggruppamento appena avvenuto. Ad esempio: (elemento_x,elemento_y)
7) **FINCHE' numero di elementi > 1 RIPARTI DA 2**

Risultato finale: stringa in questo formato

(((elemento_A,elemento_C), elemento_D),elemento_B)

Funzione leggi matrice:

@dist

```
#       1000    1000    1000    1000    1000    1000    1000    1000
#       1000    1000    1000    1000    1000    1000    1000    1000
#       1000    19.00   1000    1000    1000    1000    1000    1000
#       1000    27.00   31.00   1000    1000    1000    1000    1000
#       1000     8.00   18.00   26.00   1000    1000    1000    1000
#       1000    33.00   36.00   41.00   31.00   1000    1000    1000
#       1000    18.00    1.00   32.00   17.00   35.00   1000    1000
#       1000    13.00   13.00   29.00   14.00   28.00   12.00   1000
```

Funzione mini:

Per ogni cella $dist[$i][$j] se valore_cella<$min

```
$min=$dist[$i][$j];
$p=$i;
$q=$j;
```

Funzione newmatrix:

Chiama mini
Update matrice (raggruppamento e calcolo)
Update variabili di clustering
Stampa matrice (ed altre info)

Variabili globali (clustering) :

```
for ($i=1;$i<=$n;$i++)
   {
   $clu[$i]=1;
   $clustr[$i]=$i;
   }
```

Variabili globali :

```
$p (indice riga)
$q (indice colonna)
$min (dist. Minima)
@dist (mat. Dist.)
```

Funzione leggi matrice:

@dist

```
#       1000    1000    1000    1000    1000    1000    1000    1000
#       1000    1000    1000    1000    1000    1000    1000    1000
#       1000    19.00   1000    1000    1000    1000    1000    1000
#       1000    27.00   31.00   1000    1000    1000    1000    1000
#       1000     8.00   18.00   26.00   1000    1000    1000    1000
#       1000    33.00   36.00   41.00   31.00   1000    1000    1000
#       1000    18.00    1.00   32.00   17.00   35.00   1000    1000
#       1000    13.00   13.00   29.00   14.00   28.00   12.00   1000
```

Funzione mini:

```
Per ogni cella
$dist[$i][$j] se
valore_cella<$min
```

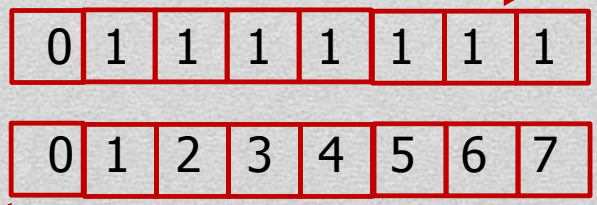
```
$min=$dist[$i][$j];
$p=$i;
$q=$j;
```

Qui terremo il conto di quanti elementi sono stati raggruppati in ogni cluster

Variabili globali (clustering) :

```
for ($i=1;$i<=$n;$i++)
   {
   $clu[$i]=1;
   $clustr[$i]=$i;
   }
```

Sono entrambi array…



Qui salveremo stringhe soluzione

Funzione leggi matrice:

@dist

$j

$i

1000	1000	1000	1000	1000	1000	1000	1000
1000	1000	1000	1000	1000	1000	1000	1000
1000	19.00	1000	1000	1000	1000	1000	1000
1000	27.00	31.00	1000	1000	1000	1000	1000
1000	8.00	18.00	26.00	1000	1000	1000	1000
1000	33.00	36.00	41.00	31.00	1000	1000	1000
1000	18.00	1.00	32.00	17.00	35.00	1000	1000
1000	13.00	13.00	29.00	14.00	28.00	12.00	1000

Funzione mini:

Per ogni cella $dist[$i][$j] se valore_cella<$min

```
$min=$dist[$i][$j];
$p=$i;
$q=$j;
```

Attraversamento della matrice: (NB: solo mezza matrice)

```
for ($i=2;$i<=$n;$i++){

        for ($j=1;$j<=$i-1;$j++){
        # operazioni
        }

}
```

Variabili globali :

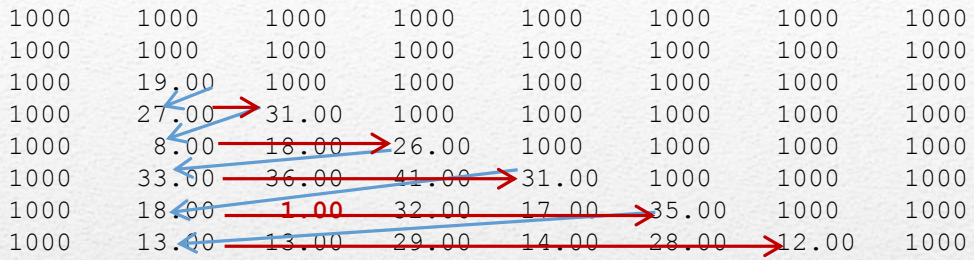
$p (indice riga)
$q (indice colonna)
$min (dist. Minima)
@dist (mat. Dist.)

Funzione leggi matrice:

@dist

$j

$i



Funzione mini:

Per ogni cella $dist[$i][$j] se valore_cella<$min

```
$min=$dist[$i][$j];
$p=$i;
$q=$j;
```

Operatori utili per test logici:

< (minore) <= (minore o uguale)
> (maggiore) >= (maggiore o uguale)
== (uguale)
!= (diverso)

Concatenamento: && (and) || (or)

```
if ($a>2 || $a <0){}
```

E' falso se $a==1 …

Variabili globali :

$p (indice riga)
$q (indice colonna)
$min (dist. Minima)
@dist (mat. Dist.)

Funzione leggi matrice:

@dist

$j

$i								
	1000	1000	1000	1000	1000	1000	1000	1000
	1000	1000	1000	1000	1000	1000	1000	1000
	1000	19.00	1000	1000	1000	1000	1000	1000
	1000	27.00	31.00	1000	1000	1000	1000	1000
	1000	8.00	18.00	26.00	1000	1000	1000	1000
	1000	33.00	36.00	41.00	31.00	1000	1000	1000
	1000	18.00	**1.00**	32.00	17.00	35.00	1000	1000
	1000	13.00	13.00	29.00	14.00	28.00	12.00	1000

Funzione mini:

```
Per ogni cella
$dist[$i][$j] se
valore_cella<$min
```

```
$min=$dist[$i][$j];
$p=$i;
$q=$j;
```

$p=6

$q=2

5 casi possibili

(mentre $i e $j variano e $p e $q sono costanti):

1) $i==$p || $j==$p
2) $i!=$q && $i!=$p && $j!=$q && $j!=$p
3) $i==$q && $j<$q
4) $j==$q && $i>$q && $i<$p
5) $j==$q && $i>$p

Variabili globali :

```
$p (indice riga)
$q (indice colonna)
$min (dist. Minima)
@dist (mat. Dist.)
```

NB: alcuni valori restano neri …
ma non vengono mai raggiunti durante l'attraversamento

5 casi possibili

(mentre $i e $j variano e $p e $q sono costanti):

1) $i==$p || $j==$p

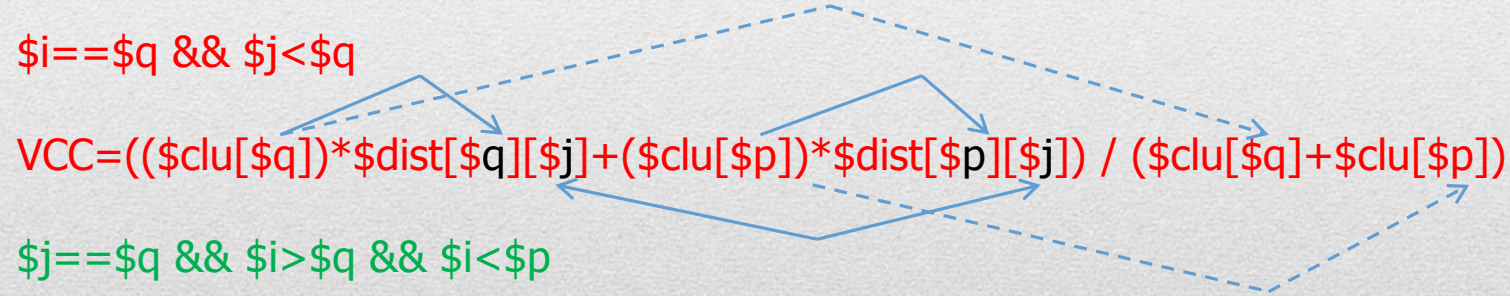
imposta valore cella corrente = 1000

2) $i!=$q && $i!=$p && $j!=$q && $j!=$p

imposta valore cella corrente (VCC) a contenuto cella corrente

3) $i==$q && $j<$q

VCC=(($clu[$q])*$dist[$q][$j]+($clu[$p])*$dist[$p][$j]) / ($clu[$q]+$clu[$p])



4) $j==$q && $i>$q && $i<$p

VCC= (clust_q*cella_iq + clust_p*cella_pi) / (clust_q+clust_p)

5) $j==$q && $i>$p

attenzione …

VCC= (clust_q*cella_iq + clust_p*cella_ip) / (clust_q+clust_p)

Dopo aver attraversato la matrice (ed aver effettuato le operazioni)
La matrice iniziale:

Gli oggetti 2 e 6 sono stati raggruppati

Matrix snapshot:

1000	1000	1000	1000	1000	1000	1000	1000
1000	1000	1000	1000	1000	1000	1000	1000
1000	19.00	1000	1000	1000	1000	1000	1000
1000	27.00	31.00	1000	1000	1000	1000	1000
1000	8.00	18.00	26.00	1000	1000	1000	1000
1000	33.00	36.00	41.00	31.00	1000	1000	1000
1000	18.00	1.00	32.00	17.00	35.00	1000	1000
1000	13.00	13.00	29.00	14.00	28.00	12.00	1000

COSA E' CAMBIATO ?

Diventa:

Matrix snapshot :

1000	1000	1000	1000	1000	1000	1000	1000
1000	1000	1000	1000	1000	1000	1000	1000
1000	18.5	1000	1000	1000	1000	1000	1000
1000	27.00	31.5	1000	1000	1000	1000	1000
1000	8.00	17.5	26.00	1000	1000	1000	1000
1000	33.00	35.5	41.00	31.00	1000	1000	1000
1000	1000	1000	1000	1000	1000	1000	1000
1000	13.00	12.5	29.00	14.00	28.00	1000	1000

Valore alto … min non li vede più

Le distanze degli altri oggetti dal gruppo sono state aggiornate,
es. dist(3, (2,6)) = (31.00+1)/2 = 31.5 (opzione 4 tra i 5 casi possibili)

Prima di passare al prossimo ciclo di raggruppamento dobbiamo fare alcune operazioni:

1) Aggiornare la variablie che tiene conto del numero di oggetti in ogni raggruppamento:

$clu[2]=$clu[2]+$clu[6];

0	1	1	1	1	1	1	1

$clu

D'ora in poi il secondo elemento è un gruppo di 2 elementi …

0	1	2	1	1	1	1	1

2) Costruire la parte di soluzione corrispondente a questo gruppo (salviamola in $clustr):

join(",", "($clustr[$q]","$clustr[$p])");

Separatore stringa1 stringa2

0	1	"(2,6)"	3	4	5	6	7

$clustr

Le distanze degli altri oggetti dal gruppo sono state aggiornate,
es. dist(1, (2,6)) = (19.00+1)/2 = 18.5

Ora il ciclo può ripartire…

Oltre alle funzioni che abbiamo descritto lo script ne include una per stampare il contenuto della matrice in un formato «pulito» … nel senso che evita di stampare i valori **1000** che utilizziamo per neutralizzare (rispetto alla funzione mini) i valori originali della matrice delle distanze.

Dopo alcuni cicli di raggruppamento si verifica un evento:

Non esistono più due gruppi separati (tutti i taxa sono in un unico gruppo). Il programma si ferma.

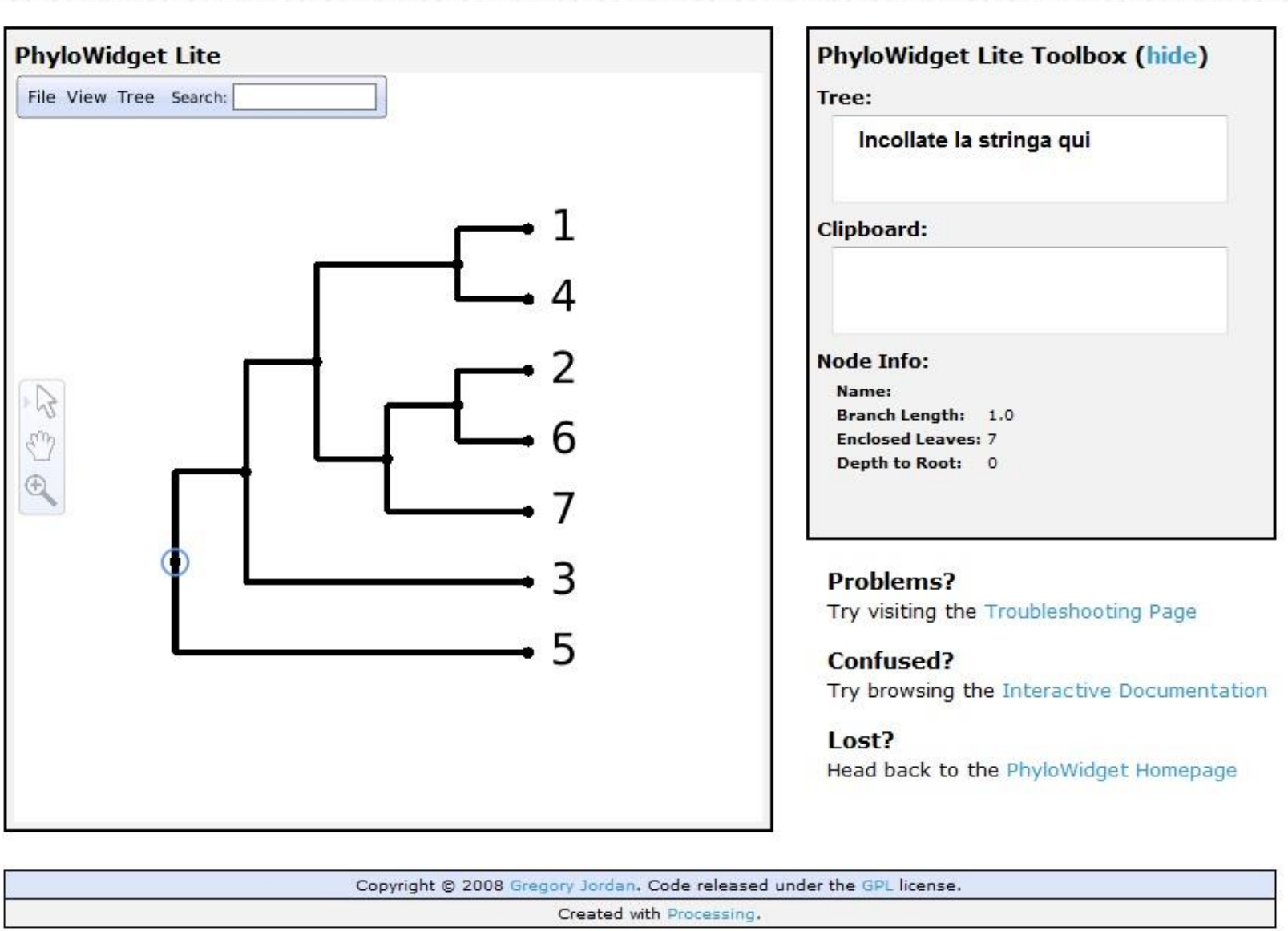
All'ultimo ciclo utile il programma stampa la soluzione:

New Tree => ((((1,4),((2,6),7)),3),5) => distance=34

Copiate questa stringa ed utilizzatela per stampare un albero da questo sito:

http://www.phylowidget.org/ (utilizzate phylowidget lite)

Dovreste ottenere questo risultato



Esercizi

- Rendere lo script funzionante (**5 pt**) (COMMENTARE IN MANIERA DETTAGLIATA)
- Modificare lo script in modo da produrre una stringa di descrizione dell'albero con questo formato(i numeri dell'esempio non hanno a che fare con il programma appena visto … è solo un esempio di formato) :
- (((1,4:dist),3:dist),2:dist) (**6 pt**)
- Stampare la matrice nella metà inferiore invece che nella metà superiore (**6 pt**)