Algoritmi e Laboratorio
a.a. 2004/2005

Prova scritta del 1 aprile 2005

COGNOME ……. NOME ……..……

Matr. n. ……..

1. **(PUNTI 2 + 1)** Completare i seguenti due algoritmi con gli invarianti di ciclo e (quando richiesto) le asserzioni finali che permettono di dimostrarne la correttezza.

ALGORITMO 1
{A.I.: $m,n >= 0$}
$a \leftarrow n$
$b \leftarrow m$
$z \leftarrow 0$

{I.C. : ….……}
while $a > 0$ **do**
    if not (a **mod** 2 = 0) **then** $z \leftarrow z + b$
    $a \leftarrow a$ **div** 2
    $b \leftarrow b + b$
return z
{A.F.: $z = m*n$}

ALGORITMO 2
{A.I.: $n >= 0$}
$c \leftarrow n$
$r \leftarrow 1$

{I.C.: ….……}
while $c \geq 1$ **do**
    $r \leftarrow r * c$
    $c \leftarrow c - 1$

return r
{A.F.: r = ….……}

2. **(PUNTI 2)**

Fornire la definizione di “algoritmo di ordinamento stabile”.

3. **(PUNTI 2 + 1)** Risolvere la seguente equazione di ricorrenza:

$T(1) = 1$
$T(n) = 4T(\lfloor n/2 \rfloor) + \lfloor n/2 \rfloor$ per $n > 1$

- mediante il metodo iterativo
- mediante applicazione del teorema principale

4. (PUNTI 5)

Scrivere un algoritmo "Divide et Impera" che risolva il seguente problema:
Dati in input un array A[1..n] di interi e un intero v, calcolare (restituendo una tripla *r* di valori interi *r.minUgu*, *r.nonUgu*, e *r.magUgu*) il numero degli elementi di A che risultano rispettivamente minori o uguali, diversi, maggiori o uguali di v.

Esempio: v = 5 A

5	9	5	12	5	7	12	11

r.minUgu = 3**,** *r.nonUgu* = 5**,** *r.magUgu* = 8

5. (PUNTI 2)

Il seguente codice (per l'alfabeto: A,B,C,D,E) puo' essere un codice di Huffman (MOTIVARE LA RISPOSTA) ?

A: 1
B: 01
C: 000
D: 001
E: 010

6. (PUNTI 2 + 2 + 2 + 2)

a) Definire che cos'è un "ordinamento topologico" di un grafo orientato.

……...

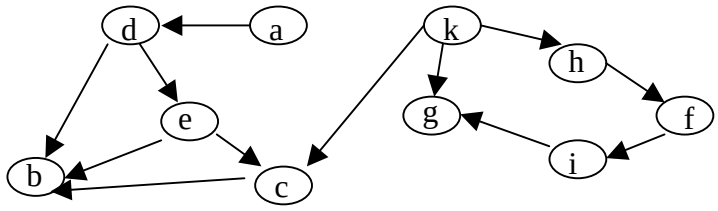
……...

……...

b) Scrivere l'algoritmo "Topological_Sort".

c) Completarlo con le asserzioni di input e di output.

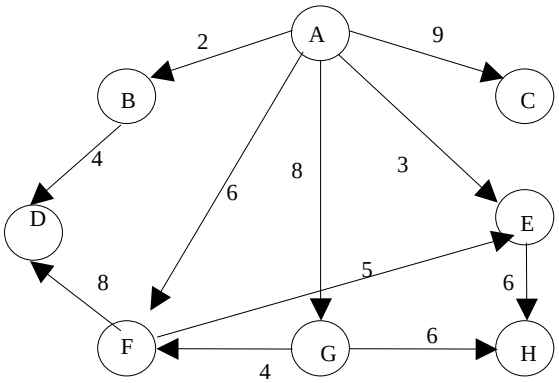
d) Individuare un ordinamento topologico del seguente grafo, applicando l'algoritmo Topological_Sort, nell'ipotesi che gli adiacenti siano memorizzati nelle liste in ordine alfabetico.



……

7. (PUNTI 5 + 1)

a) Applicando l'algoritmo di Dijkstra si determinino i pesi dei cammini minimi che collegano il vertice A con tutti gli altri vertici. Nel seguito ci riferiremo all'albero dei cammini minimi restituito dall'algoritmo con il termine "*soluzione*".



Per svolgere correttamente l'esercizio occorre:

- compilare le seguenti tabelle (la riga 0 è già compilata), e
- disegnare (un disegno per ogni riga delle tabelle) l'albero mantenuto dall'algoritmo (contenente SIA gli archi che fanno parte della *soluzione* CHE gli archi candidati) al termine di ogni iterazione (cioe' DOPO che sono state aggiornate le appetibilita' dei vertici in coda).

La prima tabella indica, per ogni iterazione del ciclo esterno, per ogni vertice v che non appartiene alla *soluzione*, la stima d[v] della distanza dal vertice A **(all'inizio il valore di tale stima e' ∞)**. Il valore di d non deve essere più riportato quando il vertice e' ormai parte della *soluzione* **(si metta il simbolo -)**.

La seconda tabella indica, per ogni iterazione del ciclo esterno, l'insieme dei vertici v che sono entrati a far parte della *soluzione* (per i quali d[v] è la distanza dal vertice A).

La riga 0 corrisponde al temine dell'inizializzazione (prima di entrare nel ciclo).

Quando nella coda con priorità ci sono vertici con lo stesso valore minimo di d si scelga quello che viene prima secondo l'ordine alfabetico.

d	A	B	C	D	E	F	G	H
0	0	∞	∞	∞	∞	∞	∞	∞
1								
2								
3								
4								
5								
6								
7								
8								

	Insieme dei vertici t inclusi nella soluzione
0	{ }
1	
2	
3	
4	
5	
6	
7	
8	

b) L'albero ottenuto e' l'unico albero dei camini minimi dal vertice A per il grafo dato (MOTIVARE LA RISPOSTA) ?

8. (PUNTI 4) Sia G un grafo non orientato e non pesato. Ogni visita in ampiezza di G individua una foresta di visita BFS con nodi interni e foglie.

Scrivere un algoritmo che, realizzando una visita in ampiezza di G, restituisca nella variabile MIN_ALTEZZA l'altezza dell'albero di minore altezza e nella variabile MAX_ALTEZZA l'altezza dell'albero di maggiore altezza tra gli alberi della foresta di visita BFS.