1. Asintotica

a) Indicare quali delle seguenti affermazioni sono vere e quali sono false.
   1. $(\log \log n)n^{1/8} = \Omega((\log n)^2)$ vera
   2. $2^{n/2} + 2^{n/2} = \Theta(2^n)$ falso
   3. $n^4-100n^2-30 = \Omega(n^4)$ vero
   4. $\log(n^n) = \Theta(\log n)$ falso
   5. $(2n)^n = \Theta((3n)^n)$ falso

b) Dimostrare che la seguente affermazione e` vera giustificando la risposta. Occorre fornire le costanti c ed n_0.

c) Dimostrare che la seguente affermazione e` falsa giustificando **matematicamente** la risposta.
$n^3\log n = \Omega(n^4)$

d) Si dimostri che se $1<f(n)=O(g(n))$ allora $(f(n))^2=O((g(n))^2)$. **Occorre utilizzare solo la definizione di O e nessuna altra proprieta`.**

e) Si dimostri che se $0<f(n)=O(g(n))$ e $1<p(n)= \Omega(q(n))$ allora $f(n)/p(n)=O(g(n)/q(n))$. **Occorre utilizzare solo la definizione di O e Ω e nessuna altra proprieta`.**

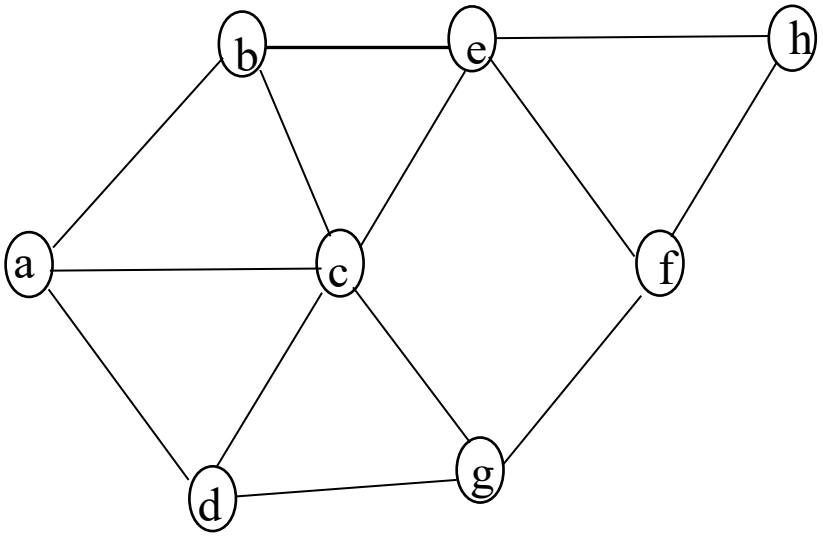
f) Si analizzi il tempo di esecuzione nel caso pessimo del seguente segmento di codice fornendo una stima asintotica **quanto migliore e` possibile** per esso. **Si giustifichi in modo chiaro la risposta.**

```
i=1, j=1
WHILE( i≤n and j≤2^m){
        i=i+1
        j=j*2

  }
```

```
FOR(i=1; i≤n; i=i+1){
   FOR(j=0; j<2*i; j=j+2) {
      print(i)
  }
}
```

2. Grafi
   a)
   I. si dica di che colore viene colorato ciascun nodo del seguente grafo dall'algoritmo che verifica se il grafo è bipartito.
   II. si dica se il grafo è bipartito o meno e nel caso non lo sia si indichino gli archi che devono essere rimossi affinché il grafo diventi bipartito **giustificando la risposta.**
   III. si fornisca una partizione (X,Y) dei nodi da cui si evinca che il grafo così ottenuto (o il grafo di partenza a seconda di quello che avete risposto al punto b) è bipartito.



   b) Si consideri l'esecuzione dell'algoritmo DFS sul grafo **non direzionato** contenente i seguenti 11 archi: (a,b) (a,c) (a,d) (a,e) (b,c) (b,d) (c,d) (c,e) (c,f) (d,f) (e,g). Si disegni l'albero DFS e si indichi per ciascun nodo un intero in modo gli interi cosi` assegnati siano consecutivi e indichino l'ordine in cui i nodi vengono esplorati. **Si assuma che nelle liste di adiacenza i nodi siano disposti in base all'ordine alfabetico delle loro etichette.**

c) Nell'algoritmo di Dijkstra con coda a priorita` ciascun vertice e` associato ad una chiave. Rispondere alle seguenti domande:
   I. cosa rappresenta la chiave di un vertice v che si trova nella coda a priorita`?
   II. in quale caso deve essere aggiornata la chiave di un vertice v che si trova nella coda a priorita` e come deve essere aggiornata?
   III. al piu` quante chiavi potrebbero essere aggiornate ogni volta che si estrae il minimo dalla coda a priorita`? (giustificare la risposta)

d) Si consideri l'algoritmo che prende in input un grafo G=(V,E) non direzionato e restituisce **true** se il grafo G e` bipartito e **false** altrimenti. Si risponda alle seguenti domande **giustificando in modo chiaro entrambe le risposte.**

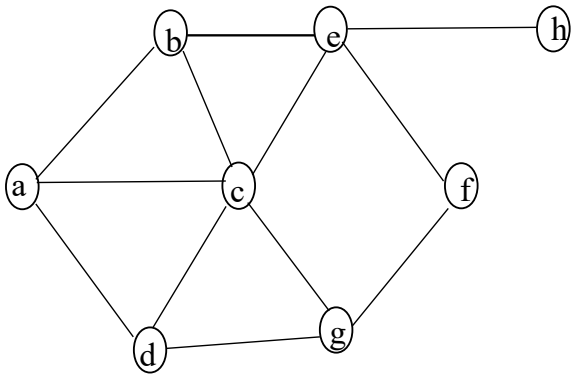
   I. Se G è bipartito, una volta eseguito il suddetto algoritmo, come si possono ottenere due insiemi disgiunti X e Y tali che $X \cup Y = V$ e ogni arco di G collega un nodo di X ad uno di Y?
   II. Siano u e v due nodi di G adiacenti tra loro, colorati dal suddetto algoritmo con lo stesso colore. Se l(u) è il numero di nodi che separano u dalla radice nell'albero BFS, cosa possiamo dire su l(v), cioe` sul numero di nodi che separano v dalla radice.

e) Scrivere lo pseudocodice di un algoritmo **ricorsivo** con tempo di esecuzione $\mathbf{O(n^2)}$ che prende in input un grafo direzionato e restituisce
   III. una sequenza dei nodi ordinati in base all'ordinamento topologico, se il grafo è un DAG
   IV. un sequenza contenente solo un nodo fittizio **q**, se il grafo non è un DAG.

   Se non si sa scrivere lo pseudocodice per il punto 2 si implementi solo lo pseudocodice per il punto 1. In questo caso il punteggio dell'esercizio sara` inferiore.

f)
   i. Scrivere lo pseudocodice dell'algoritmo di Prim con le linee per costruire il minimo albero ricoprente.
   ii. Spiegare a parole cosa rappresenta la chiave associata ad un nodo nella coda` a priorita`.
   iii. Spiegare perche' l'arco inserito ad un generico passo dall'algoritmo fa sicuramente parte di un minimo albero ricoprente.

g) Si disegni l'albero BFS generato da una visita BFS del seguente grafo a partire dal nodo sorgente **a**. Si assuma che i nodi siano disposti nelle liste di adiacenza in base all'ordine crescente delle proprie etichette.



Greedy

a) Fornire un'istanza di Interval Scheduling con n=7 e valore della soluzione ottima uguale a 4.

b) Si dimostri che l'algoritmo greedy per Interval Scheduling produce la soluzione ottima usando il seguente fatto: *per ogni p, il p-esimo job selezionato dall'algoritmo greedy termina non piu` tardi del job con il p-esimo tempo di fine piu` piccolo tra quelli selezionati dall'algoritmo ottimo*. Non occorre dimostrare il suddetto fatto.

c) Si esegua l'algoritmo greedy per la minimizzazione dei ritardi sulla seguente istanza del problema: t_1=8, t_2=4, t_3=2, t_4=9, t_5=6, t_6=5, d_1=4, d_2=11, d_3=2, d_4=23, d_5=6, d_6=7, dove i valori t_i sono le durate e i valori d_i le scadenze. Si mostrino gli intervalli di tempo assegnati ai job dall'algoritmo e si indichino il ritardo di ciascun job e il ritardo massimo

d) Si consideri la seguente istanza di interval scheduling: [7,9],[1,4],[3,6],[1,5],[4,9],[5,6],[4,6]. Fornire tutte le possibili soluzioni ottime per questa istanza **ottenibili con la strategia greedy.**

e) Si dimostri che la strategia greedy per la minimizzazione dei ritardi produce uno schedule ottimo usando i seguenti due fatti:
1. Tutti gli scheduling senza inversioni e privi di idle time hanno lo stesso ritardo massimo dello schedule prodotto dalla strategia greedy.
2. Ogni scheduling può essere trasformato in uno scheduling senza inversioni e privo di idle time senza che aumenti il suo ritardo massimo.
NB: non occorre dimostrare i fatti 1 e 2.

f) Si scriva lo pseudocodice dell'algoritmo per la minimizzazione dei ritardi.

g) Si scriva lo pseudocodice dell'algoritmo greedy ottimo per partizionamento di intervalli e si analizzi il tempo di esecuzione nel caso pessimo **giustificando la risposta**. **Il codice deve essere scritto in italiano, cccezion fatta per le parole chiave while, if, else , for, ecc.**

h) Si spieghi come implementare l'algoritmo al punto precedente in modo che abbia tempo totale di esecuzione O(nlogn)