Esercizi Analisi asintotica:

a) Indicare quali delle seguenti affermazioni sono vere e quali sono false.

1. $4n^4 = O(n^3)$
2. $n^{\log n} = \Omega((\log n)^n)$
3. $n (\log n)^{1/2} + n^{1/3} = O(\log n^n)$
4. $n^3+10 = \Omega(n^3+10n)$
5. $2^{\log_4 n} = \Theta(n)$

1. $n^4 2^n = O(4^n)$
2. $n^n = \Omega(n^{3n/2} n^{n/2})$
3. $n (\log n) = \Theta(\log n^{2n})$
4. $n^3+1000n^2+100 = \Theta(n^3+n)$
5. $\log(\log n) + n = O((\log n)^{1/2} + n^{1/2})$

1. $n (\log n)^4 + 1000n^2 = O(n^4)$
2. $n^n = \Omega(n^{3/4} n^{1/4})$
3. $n^{1/2} = O(\log n)$
4. $n^3+1000n^2+100 = \Theta(n^3)$
5. $\log(\log n) = \Omega((\log n)^{1/2})$

1. $(\log n) n^{1/2} = O((\log n)^2)$
2. $n^{1/4} = \Omega(2^{30})$
3. $\log_4 n = \mathbf{\Theta}(\log_2 n)$
4. $(n/4)^n = \mathbf{\Theta}(n^n)$
5. $¼\ n + n^{100} = O(n^{1/4n})$

1. $(\log n) n^{1/8} = O((\log n)^2)$
2. $n^2\ 2^{n/2} = \Omega(2^n)$
3. $10n+n^k = O(n)$, k e` una costante minore o uguale di 1
4. $1000n^4 - 100n^2 = \Omega(n^3)$
5. $¼\ n\ + n^{1/2} \log n = O(n)$

1. $\log(n^n) = \Theta((\log n)^n)$
2. $n^{1/2} = \Omega(n^{1/4})$
3. $1000\ n+n^3 = O(n)$
4. $1000n^4 - 100n^2 = \Omega(n^3)$

1. $n^2 \log^2 n + 100n^2 = O(n^4)$
2. $n^{1/8} = \Omega(n^{1/4})$
3. $n^{1/3} = O(\log n)$
4. $n^3 - 1000n^2 + 8 = \Omega(n^3)$
5. $\log(\log n) = O((\log n)(\log n))$

$4n^4 = O(n^3) \Rightarrow \exists\ c>0,\ n_0 \geq 0 \mid 4n^4 \leq cn^3 \quad \forall n \geq n_0$

$4n^4 \leq cn^3 \Rightarrow \quad cn^3 \geq 4n^4 \Rightarrow \quad c \geq \frac{4n^4}{n^3} \Rightarrow \quad c \geq 4n$

ma essendo c una costante abbiamo che qualsiasi valore di c noi prendiamo possiamo trovare un n grande tale che $c < 4n$ quindi $4n^4 \neq O(n^3)$

$n \log n = \Omega((\log n)^n) \Leftrightarrow \exists\ c>0,\ n_0 \geq 0 \mid n\log n \geq c(\log n)^n \quad \forall n \geq n_0$

$c \log^n n \leq n \log n \Rightarrow \quad c \leq \frac{n \log n}{\log^n n} \Rightarrow \quad c \leq \frac{n}{\log^n n}$

al tendere di n all'infinito

b) Si dimostri che se $1<f(n)=O(h(n))$ allora $(f(n))^a = O(h(n)^a)$, dove a è una costante positiva. Occorre utilizzare solo la definizione di O e nessuna altra proprieta`.

b) Si dimostri che se $0<f(n)=O(h(n))$ allora $a(f(n)) = O(h(n))$, dove a è una costante positiva. Occorre utilizzare solo la definizione di O e nessuna altra proprieta`.

b) Si dimostri che se $f(n)= \Omega (g(n))$ e $g(n)= \Omega(p(n))$ allora $f(n)= \Omega(p(n))$. **Occorre utilizzare solo la definizione di Ω e nessuna altra proprieta` (fornire le costanti c ed n_0).**

b) Si dimostri che se $f(n)=O(g(n))$ allora $nf(n)=O(ng(n))$. **Occorre utilizzare solo la definizione di O e nessuna altra proprieta`.**

b) Si dimostri che se $f(n)=O(g(n))$ e $g(n)=O(h(n))$ allora $a(f(n))=O(h(n))$, dove a e` una costante. **Occorre utilizzare solo la definizione di O e nessuna altra proprieta`.**

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

b) Si dimostri che se $0<f(n)=O(h(n))$ e $0<g(n)= O(p(n))$ allora $f(n)g(n)= O(h(n)\, p(n))$. Occorre utilizzare solo la definizione di O e nessuna altra proprieta`.

---

c) 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<m)
  if (i <= j)
    i=i+1
  else
    j=j+1
```

```
FOR(i=1; i<n; i=i+1){
  FOR(j=1; j<3^i; j=j*3) {
    print(j);
  }
}
```

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

```
FOR(i=1; i≤n; i=i+1){

  FOR(k=i; k<2i; k=k+1) {
    print(k);
  }

}
```

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

}
```

```
FOR(i=n; i>0; i=i/3){

  FOR(k=2; k<m; k=k^2) {
    print(k);
  }

  print(i);
}
```

```
FOR(i=1; i<n; i=5*i){
  FOR(j=0; j<10; j=j+1) {
     print(j);
  }
}
```

Esercizi Grafi:

a) Si scriva lo pseudocodice **dell'algoritmo BFS** che fa uso di una **coda FIFO** aggiungendo anche le linee di codice per la costruzione dell'albero BFS. Si analizzi il tempo di esecuzione dell'algoritmo proposto.

a) Si scriva lo pseudocodice **dell'algoritmo di Dijkstra** che fa uso di una **coda a priorita`** aggiungendo anche le linee di codice per la costruzione dell'albero dei cammini minimi. Si analizzi il tempo di esecuzione dell'algoritmo proposto quando la coda a priorita` è implementata con un heap binario.

a) Si scriva lo pseudocodice **dell'algoritmo di Prim** per il minimo albero ricoprente aggiungendo anche le linee di codice per la costruzione dell'albero. Si analizzi il tempo di esecuzione dell'algoritmo proposto quando la coda a priorita` è implementata con un heap binario.

b) Scrivere lo pseudocodice dell'algoritmo per ottenere l'ordinamento topologico di un DAG e si analizzi il tempo di esecuzione dell'algoritmo proposto. **Analizzare il tempo di esecuzione significa fornire un limite superiore asintotico quanto migliore e` possibile al tempo di esecuzione dell'algoritmo giustificando la risposta.**

b) Scrivere lo **pseudocodice** dell'algoritmo che dato un grafo direzionato aciclico restituisce l'ordinamento topologico del grafo. **L'algoritmo deve avere tempo O(n+m).** Il voto dipendera` da quanto dettagliato sara` lo pseudocodice.

a) Si scriva lo pseudocodice di un algoritmo BFS **senza** coda FIFO che abbia tempo di esecuzione O(n+m) .

b) Si scriva lo pseudocodice di un algoritmo che prende in input un grafo G non orientato e restituisce true se il grafo G e` bipartito e false altrimenti. L'algoritmo deve avere tempo di esecuzione O(n+m). Il voto dipendera`, oltre che dalla correttezza dell'algoritmo, anche da quanto dettagliato e` lo pseudocodice.

b) Scrivere lo **pseudocodice** dell'algoritmo che dato un grafo direzionato aciclico restituisce l'ordinamento topologico del grafo. **L'algoritmo deve avere tempo O(n+m).** Il voto dipendera` da quanto dettagliato sara` lo pseudocodice.

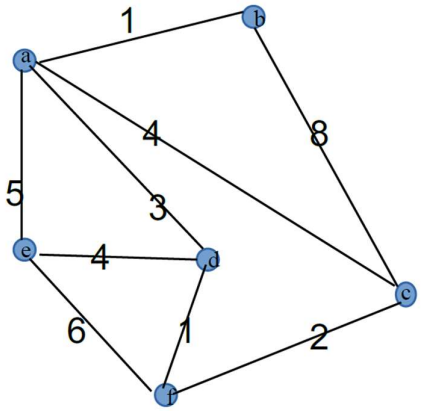
a) Si definisca formalmente il concetto di ordine topologico di un grafo direzionato aciclico. In assenza di questa definizione i punti successivi dell'esercizio non saranno valutati.

c) Si consideri l'algoritmo che determina se un grafo è bipartito. Si dimostri che se l'algoritmo colora due nodi adiacenti dello stesso colore allora il grafo non è bipartito.

c) Si descriva un algoritmo che prende in input un grafo non direzionato e, nel caso in cui il grafo contenga un ciclo, restituisce una lista contenente i nodi che formano il ciclo e in caso contrario restituisce una lista vuota. L'algoritmo deve avere tempo di esecuzione O(n+m) nel caso pessimo.

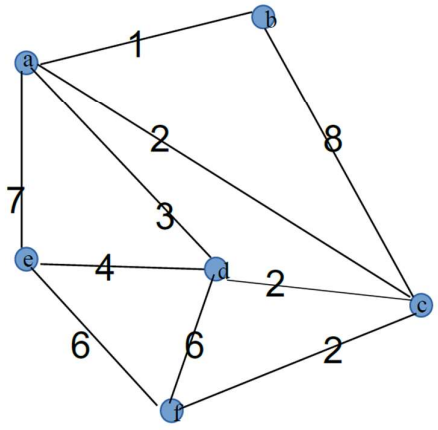
b) Si definisca il concetto di ordinamento topologico di un grafo direzionato e si dimostri che se un grafo direzionato G è un DAG allora G ha un ordinamento topologico.

a) Si mostri l'esecuzione dell'algoritmo di Prim sul seguente grafo. Si assuma che l'algoritmo scelga come radice il nodo **a. Occorre mostrare per ogni passo il contenuto della coda a priorita` e l'albero ottenuto fino a quel passo. Disegnare alla fine l'albero generato.**

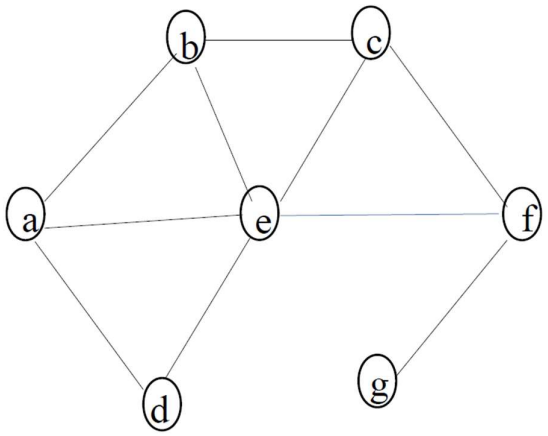


Si mostri l'esecuzione dell'algoritmo di Dijkstra sul seguente grafo a partire dal nodo sorgente a. Si assuma che l'algoritmo scelga come radice il nodo **a. Occorre mostrare per ogni passo il contenuto della coda a priorita` e l'arco aggiunto all'albero dei percorsi minimi (arco formato dal nodo v aggiunto ad S in quel passo e dal nodo in S che precede v lungo il cammino minimo da s a v). Disegnare alla fine l'albero dei percorsi minimi generato.**

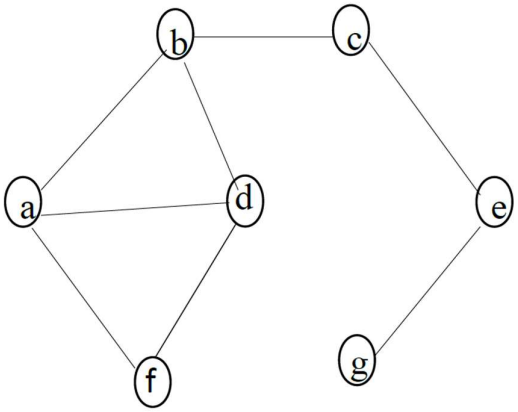
a) Si mostri l'esecuzione dell'algoritmo di Krukal sul seguente grafo. **Occorre mostrare per ogni passo la foresta di alberi ottenuta fino a quel passo.**



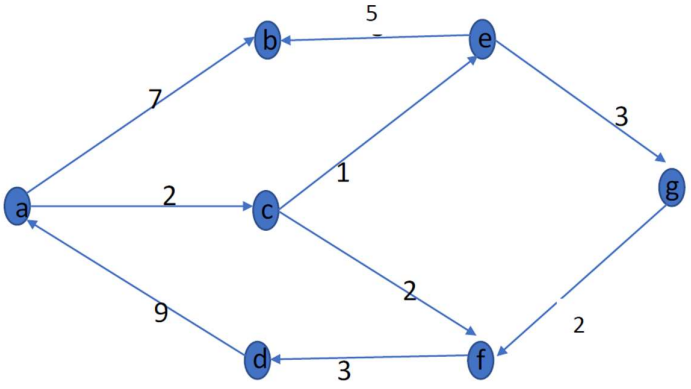
a) Si disegni l'albero DFS generato da una visita DFS del seguente grafo a partire dal nodo sorgente **a**. Si assuma che i nodi siano disposti nelle liste di adiacenza in base all'ordine crescente delle proprie etichette.



b) Si disegni l'albero DFS generato da una visita DFS del seguente grafo a partire dal nodo sorgente **a** numerando i nodi dell'albero con interi consecutivi che indicano in che ordine i nodi sono stati visitati. Si assuma che i nodi siano disposti nelle liste di adiacenza in base all'ordine crescente delle proprie etichette.



b) Si mostri l'esecuzione dell'algoritmo di Dijkstra sul seguente grafo a partire dal nodo sorgente **a. Per ogni passo si mostri il contenuto della coda a priorita`, incluse le chiavi degli elementi, e l'albero dei percorsi minimi costruito fino a quel passo.**



Esercizi Greedy:

a) Si spieghi in che cosa consiste un'istanza (input) del problema della minimizzazione dei ritardi e qual è l'obiettivo del problema (output). Devono essere definite anche tutte le quantita` utilizzate per spiegare l'obiettivo del problema . Se dalla risposta a questo punto si evincera` che lo studente non sa in cosa consiste il problema della minimizzazione dei ritardi, i punti successivi dell'esercizio non saranno valutati.

b) Si fornisca un'istanza del problema della minimizzazione dei ritardi con n=6 per cui il valore della soluzione ottima è 5 e al piu` due attivita` hanno ritardo 0 nella soluzione ottima. Si mostri chiaramente perche' il valore della soluzione per l'istanza da voi fornita è 5 e qual è il ritardo di tutte le attivita`.

c) [6 cfu tutti e 9 cfu nuovo] Si scriva lo pseudocodice dell'algoritmo greedy che restituisce la soluzione ottima per il problema della minimizzazione dei ritardi.

a) Si spieghi in che cosa consiste un'istanza (input) del problema dell'interval scheduling e in cosa consiste una soluzione (output) del problema. Se dalla risposta a questo punto si evincera` che lo studente non sa in cosa consiste il problema dell'interval sheduling, i punti successivi dell'esercizio non saranno valutati.

b) Si fornisca un'istanza del problema dell'interval scheduling con n=6 per cui il valore della soluzione ottima è 3. Si specifichino i valori numerici che descrivono l'input. **Non è sufficiente fornire un disegno che descriva l'input.**

c) Si scriva lo pseudocodice dell'algoritmo greedy che restituisce la soluzione ottima per il problema dell'interval scheduling.

a) Si spieghi in che cosa consiste un'istanza (input) del problema del partizionamento di intervalli e qual è l'obiettivo del problema (output). Se dalla risposta a questo punto si evincera` che lo studente non sa in cosa consiste il problema, i punti successivi dell'esercizio non saranno valutati.

b) Si scriva lo pseudocodice dell'algoritmo greedy che restituisce il valore della soluzione ottima per il problema del partizionamento di intervalli. SI SCRIVA LO PSEUDOCODICE IN ITALIANO: le uniche parole inglesi consentite sono le parole chiave if, for, ecc. .

c) [6 cfu tutti e 9 cfu nuovo] Si fornisca un'istanza del problema **dell'Interval Scheduling** per la quale la strategia greedy "Fewest Conflicts" non fornisce una soluzione ottima. Occorre indicare i tempi di inizio e di fine di ogni job.

a) Si descriva in modo chiaro e schematico in che cosa consiste un'istanza del problema della minimizzazione dei ritardi (input) e qual e` l'obiettivo del problema (output). **Definire in modo preciso le quantita` che intervengono nella descrizione dell'output del problema.** Se dalla risposta a questo punto si evincera` che lo studente non sa in cosa consiste il problema della minimizzazione dei ritardi, i punti successivi dell'esercizio non saranno valutati.

b) Si fornisca un controesempio di 4 job che dimostra che la strategia shortest processing time first non sempre fornisce la soluzione ottima. **Giustificare in modo chiaro la risposta.**

c) Si scriva lo pseudocodice di un algoritmo greedy che trova la soluzione ottima per il problema della minimizzazione dei ritardi descrivendo il significato di tutte le variabili che compaiono nel codice. **Nel caso in cui non venga fornita questa descrizione**

d) Si descriva la strategia greedy che permette di ottenere la soluzione ottima per il problema del **Partizionamento di Intervalli** e si dica a cosa e` uguale il valore della soluzione ottima. Fornire una definizione chiara di eventuali concetti utilizzati per rispondere al quesito.

c) Si spieghi in che cosa consiste un'istanza (input) del problema del **Partizionamento di Intervalli** e in cosa consiste una soluzione (output) del problema. **Se dalla risposta a questo punto si evincera` che lo studente non sa in cosa consiste il problema del Partizionamento di Intervalli, il punto successivo del quesito non sara` valutato.**

b) Si spieghi in che cosa consiste un'istanza (input) del problema dell'Interval Scheduling e in cosa consiste una soluzione (output) del problema. **Se dalla risposta a questo punto si evincera` che lo studente non sa in cosa consiste il problema dell'Interval Scheduling, i punti successivi del quesito non saranno valutati.**

c) Si fornisca un controesempio costituito da un'istanza **di almeno 5 attivita`** che dimostri che la strategia Fewest Conflicts non sempre fornisce la soluzione ottima al problema dell'Interval Scheduling. Si spieghi per quale motivo l'istanza da voi proposta rappresenta un controesempio al fatto che Fewest Conflicts produca una soluzione ottima al problema.

d) Si scriva lo pseudocodice dell'algoritmo greedy per il problema dell'Interval Scheduling spiegando **il significato di tutti i parametri e variabili che compaiono nello pseudocodice.** Si analizzi il tempo di esecuzione dell'algoritmo nel caso pessimo. Analizzare il tempo di esecuzione di un algoritmo significa fornire una stima asintotica quanto migliore e` possibile del suo tempo di esecuzione **giustificando in modo chiaro la risposta.**

b) Si spieghi che cosa e` un eviction scheduling ridotto (se non lo si e` gia spiegato al punto precedente) e si dimostri che e` sempre possibile trasformare un eviction schedule in un eviction schedule ridotto senza aumentare il numero totale di inserimenti nella cache.

a) Si spieghi in che cosa consiste un'istanza (input) del problema del caching offline e in cosa consiste una soluzione (output) del problema. Se dalla risposta a questo punto si evincera` che lo studente non sa in cosa consiste il problema del caching offline, i punti successivi dell'esercizio non saranno valutati.