Progettazione di Algoritmi (9 CFU)
12/7/2018

Cognome e Nome:
Numero di Matricola:

Spazio riservato alla correzione

1	2	3	4	5	6	Totale
/20	/8	/25	20	/15	/12	/100

1.

a) Indicare quali delle seguenti affermazioni sono vere e quali sono false.
   1. $(\log n)n^{1/2} = O(n)$
   2. $n^{1/2} = \Omega(n^{1/4})$
   3. $1000\,n = O(n)$
   4. $n^4 - 1000n^2 + 8 = O(n^3)$
   5. $\frac{1}{4}\,n = O(n)$

b) Si dimostri che se $f(n)=O(n)$ e $g(n)=O(h(n))$ allora $f(n)g(n)=O(nh(n))$. **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.**

```
FOR(i=1; i≤n; i=i+1){

     FOR(k=0; k<m; k=k+10) {
      print(k);
     }

     j=0;
     while(j<m) {
     if(j%2==0)
            j=j+4;

     print(j);
   }


          }
```

2

Si scriva lo pseudocodice di un algoritmo ricorsivo che prende in input un nodo u di un albero binario di interi e restituisce la somma degli elementi dei nodi del sottoalbero avente come radice u.

NB: L'elemento di un generico nodo v è v.dato.

3.

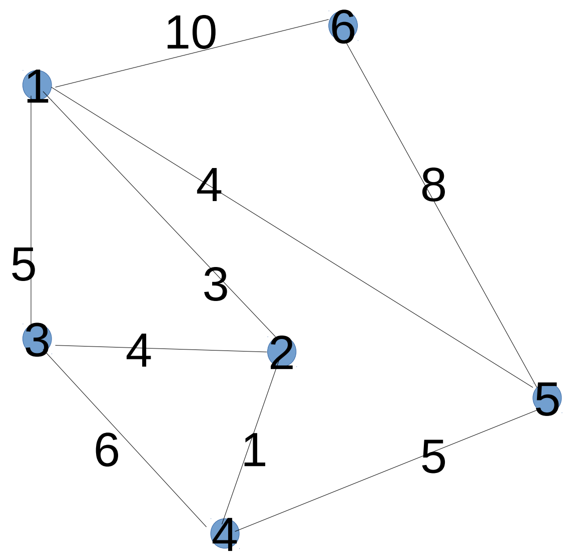
a) Scrivere lo pseudocodice dell'algoritmo BFS che fa uso di una coda FIFO e si analizzi la complessita` dell'algoritmo nel caso pessimo. Analizzare il tempo di esecuzione di un algoritmo significa fornire una stima asintotica quanto migliore e` possibile del suo tempo di esecuzione **giustificando in modo chiaro la risposta.**

b) Si forniscano le istruzioni che permettono all'algoritmo BFS al punto a) di costruire l'albero BFS e si indichi in che punti dell'algoritmo vanno inserite queste istruzioni. A tale scopo si numerino le linee di codice dell'algoritmo BFS.

c) Si descriva in modo chiaro e schematico come modificare o come utilizzare l'algoritmo BFS per disegnare un algoritmo che dato in input un grafo non direzionato restituisce true se il grafo non contiene cicli e false altrimenti. **Non e` necessario scrivere lo pseudocodice.**

d) Si modifichi l’algoritmo al punto c) in modo che, nel caso in cui il grafo input contenga un ciclo, l’algoritmo restituisca la sequenza dei nodi che fanno parte del ciclo e in caso di assenza di cicli restituisca una sequenza vuota. **Non e` necessario scrivere lo pseudocodice.**

e) Mostrare l'esecuzione dell'algoritmo di Kruskal sul seguente grafo**. Occorre mostrare per ogni passo l'arco esaminato dall'algoritmo e disegnare alla fine l'albero generato.**



4.

a) Si spieghi in che cosa consiste un'istanza (input) del problema della minimizzazione dei ritardi e in cosa consiste una soluzione (output) del problema definendo in modo chiaro e schematico la nozione di ritardo di un job e di ritardo massimo.

b) Si scriva lo pseudocodice dell’algoritmo greedy che da` in output la soluzione per il problema della minimizzazione dei ritardi 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.**

c) Si spieghi che cosa e` un’inversione e si dimostri che uno scheduling privo di idle time e di inversioni e` possibile scambiare due job con la stessa scadenza senza che il ritardo massimo aumenti.

5.

Si consideri il seguente problema. Per il saggio di fine anno gli alunni di una scuola saranno disposti in fila secondo un ordine prestabilito e **non modificabile**. La fila sara` suddivisa in gruppi contigui e ciascun gruppo dovra` intonare una parte dell'inno della scuola. Il maestro di canto ha inventato un dispositivo che permette di valutare come si fondono le voci di un gruppo tra di loro. Piu` **basso** e` il punteggio assegnato dal dispositivo ad un gruppo, migliore e` l'armonia delle voci. Il maestro vuole ripartire la fila di alunni in gruppi contigui in modo da ottenere entrambi i seguenti obiettivi

I. la somma dei punteggi dei gruppi sia la piu` piccola possibile
II. il numero totale di gruppi non sia troppo grande per evitare che l'inno debba essere suddiviso in parti troppo piccole. Ogni gruppo fa aumentare il costo della soluzione di un valore costante $g>0$.

NB: Il dispositivo computa per ogni coppia di posizioni i e j con in $i \leq j$, il valore $f(i,j)$ dove **f(i,j)** = punteggio assegnato al gruppo che parte dall'alunno in posizione i e termina con l'alunno in posizione j.

a) Si formuli il suddetto problema sotto forma di problema computazionale specificando in cosa consistono un'istanza del problema (input) e una soluzione del problema (output). Occorre definire una funzione costo di cui occorre ottimizzare il valore.

b) Si fornisca una formula ricorsiva per il calcolo del valore della soluzione ottima del problema basata sul principio della programmazione dinamica. **Si spieghi in modo chiaro come si ottiene la suddetta formula.**

c) Si scriva lo pseudocodice dell’algoritmo che trova il valore della soluzione ottima per il problema.

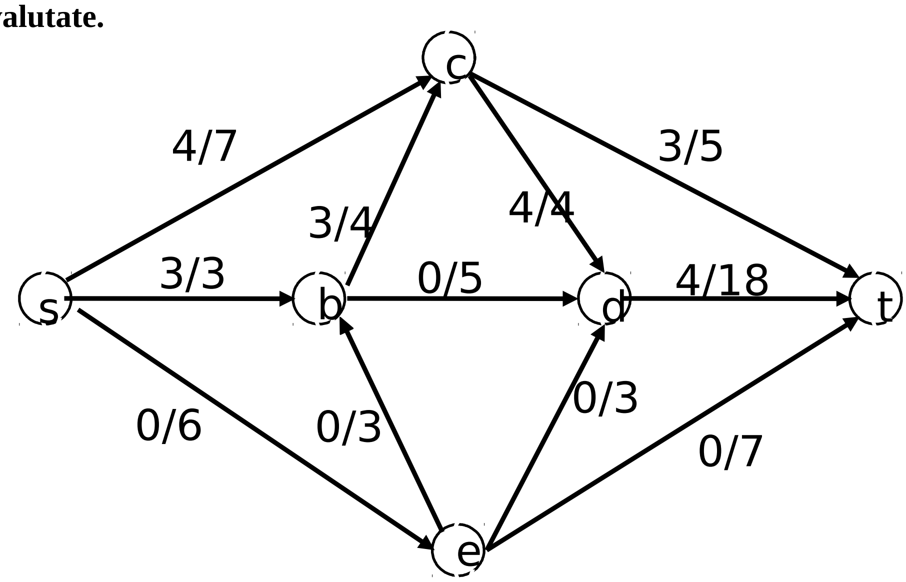
6.

a) Si consideri la seguente rete di flusso e la funzione di flusso i cui valori sono indicati a sinistra delle capacita` degli archi. Si disegni la rete residua rispetto alla funzione flusso indicata e si dica se questa funzione ha valore massimo. Nel caso in cui la funzione non abbia valore massimo, si fornisca **la funzione flusso con valore massimo e il taglio di capacita` minima.** A tal fine si eseguano una o piu` iterazioni dell'algoritmo di Ford-Fulkerson **a partire dalla funzione di flusso data.**

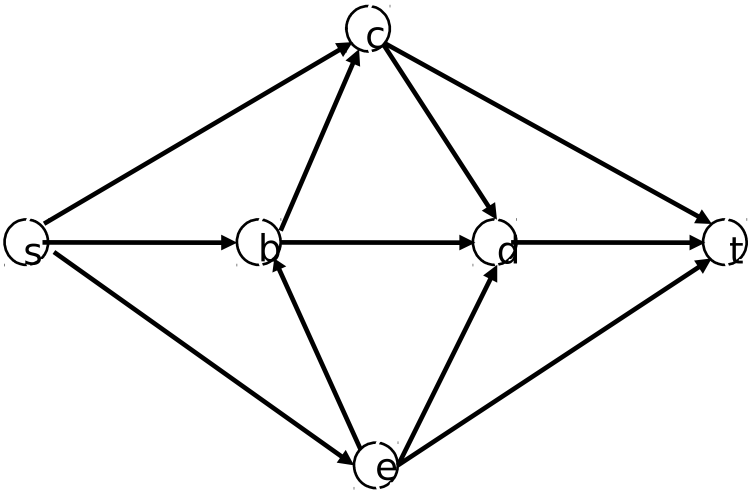
Per ogni iterazione dell'algoritmo, occorre

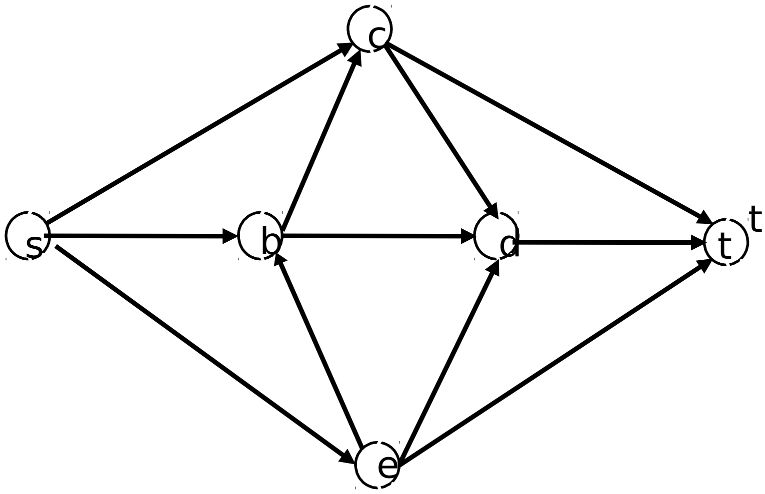
- **disegnare la rete residua all’inizio di quell’iterazione**
- **indicare il cammino aumentante da voi scelto**
- **mostrare il valore associato ad ogni arco del grafo al termine di quella iterazione**

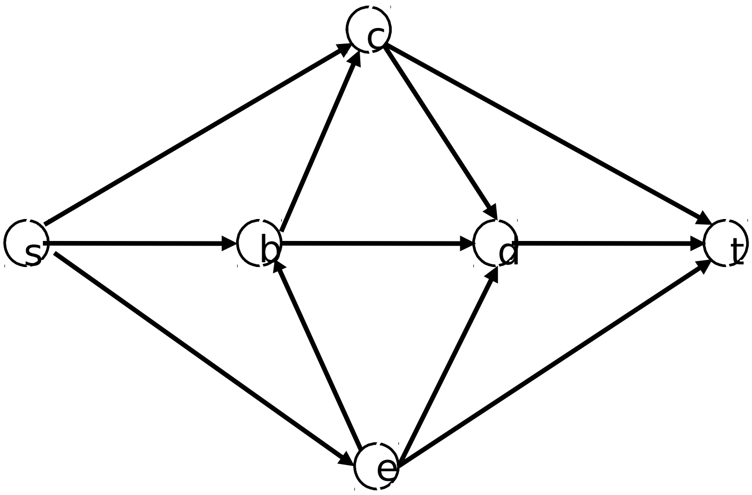
N.B.: le risposte che non sono ottenute a partire dalla funzione di flusso data non saranno valutate.

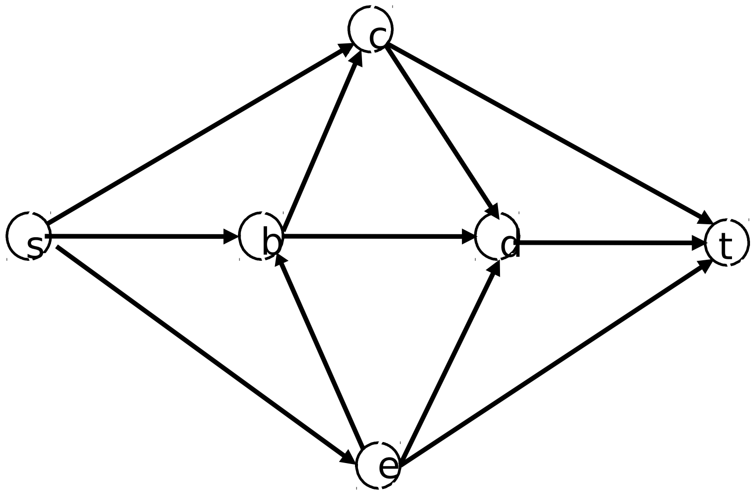


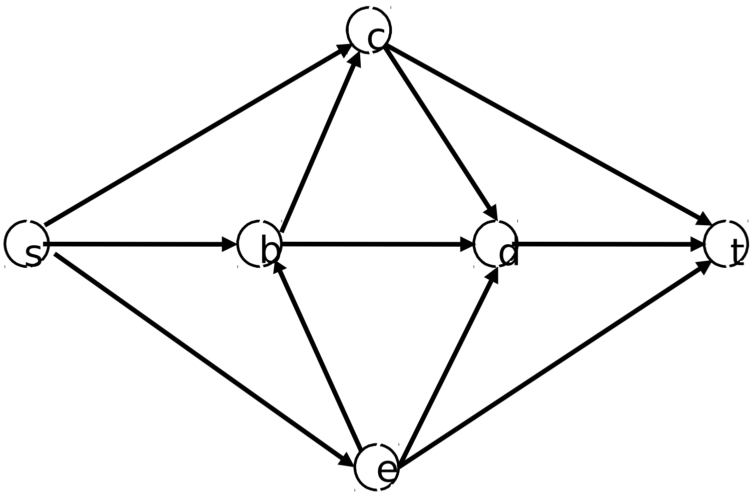
Per vostra comodita`, di seguito sono riportate diverse copie della rete di flusso, suddivise a coppie. **A partire dalla funzione di flusso data, usate l’immagine di sinistra di ciascuna coppia per disegnare la rete residua e l’immagine di destra per riportare i valori della funzione flusso assegnati a ciascun arco.** Ovviamente potrebbe essere necessario aggiungere e/o cancellare (con una x) degli archi nelle immagini di sinistra. Il numero di coppie non e` indicativo del numero di iterazioni effettuate dall’algoritmo di Ford-Fulkerson. Procedete dall’alto verso il basso utilizzando solo le coppie di grafi che vi servono per illustrare l’intera esecuzione dell’algoritmo. **N.B.: Se non saranno rispettate queste indicazioni per lo svolgimento dell’esercizio, l’esercizio non sara` valutato.**

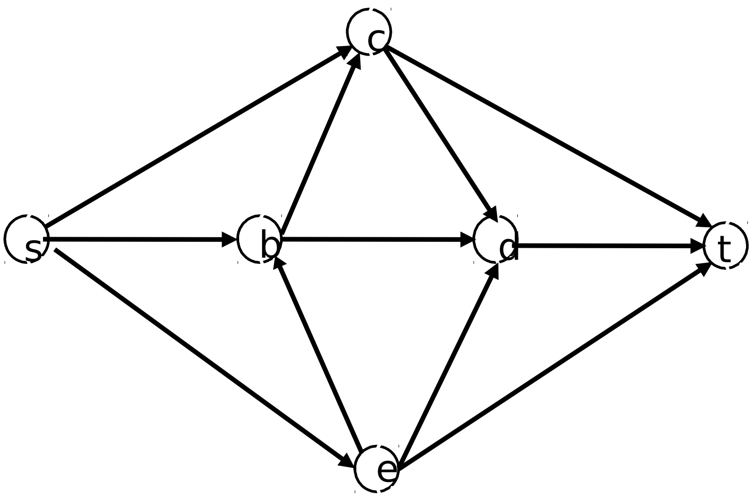

c
s
b
d
t
e

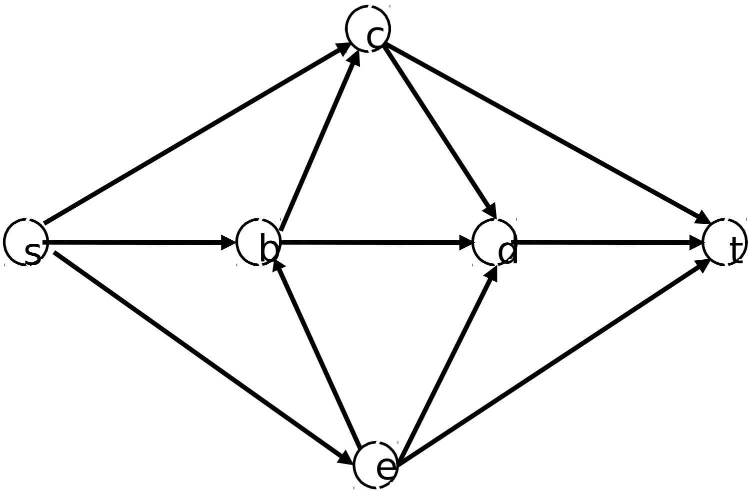

c
t
s
b
d
t
e

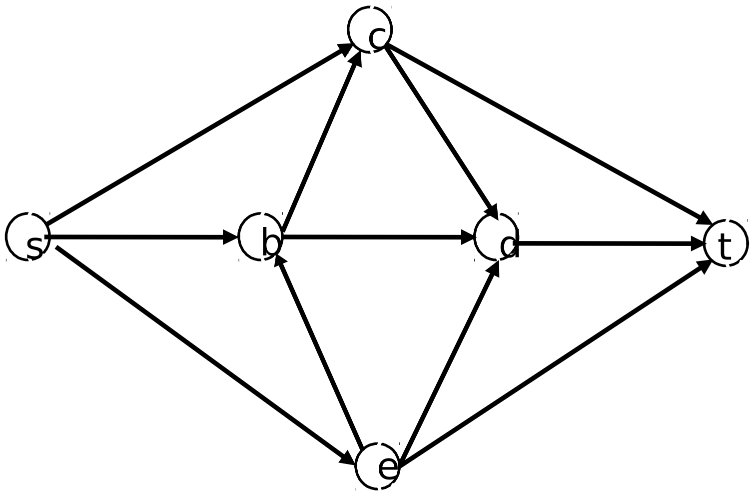

c
s
b
d
t
e

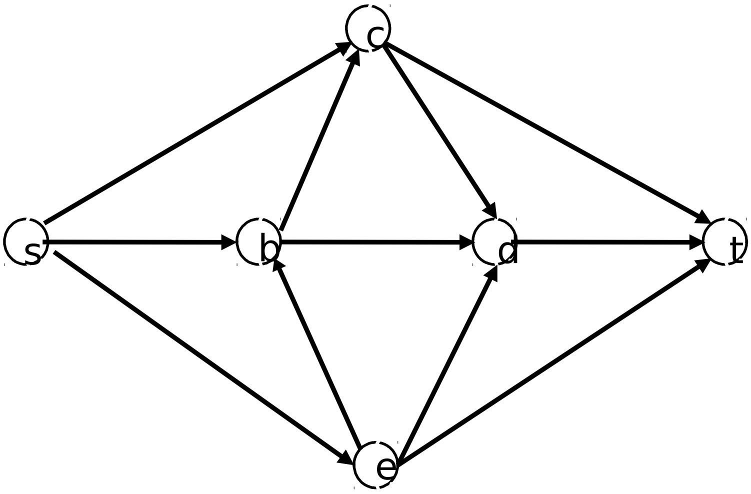

c
s
b
d
t
e

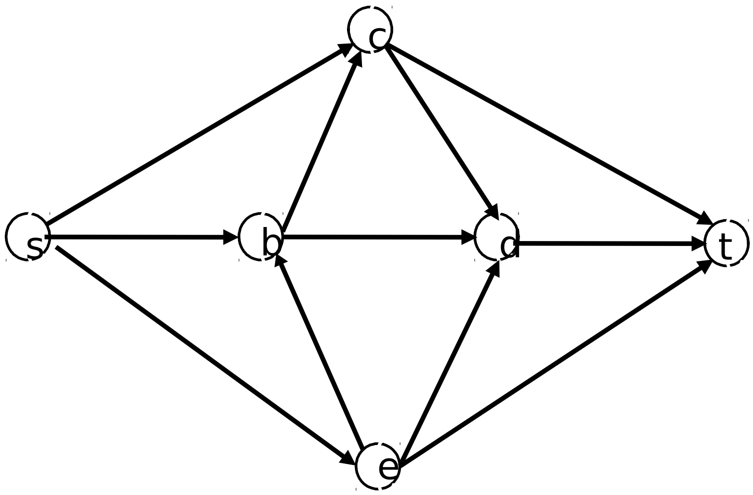

c
s
b
d
t
e

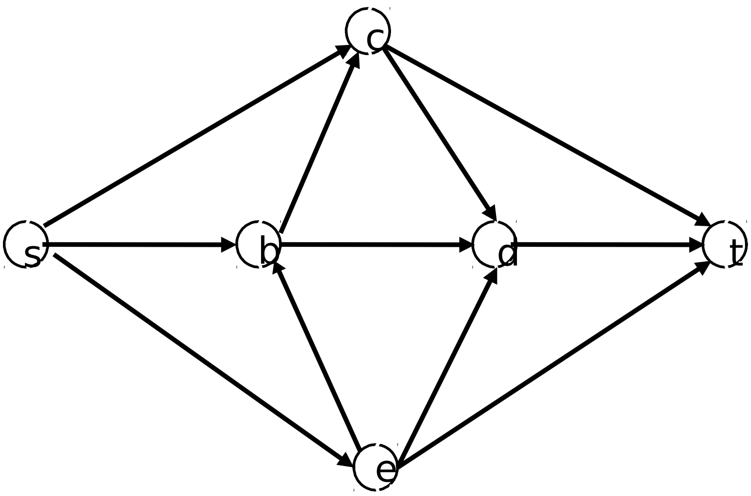

c
s
b
d
t
e

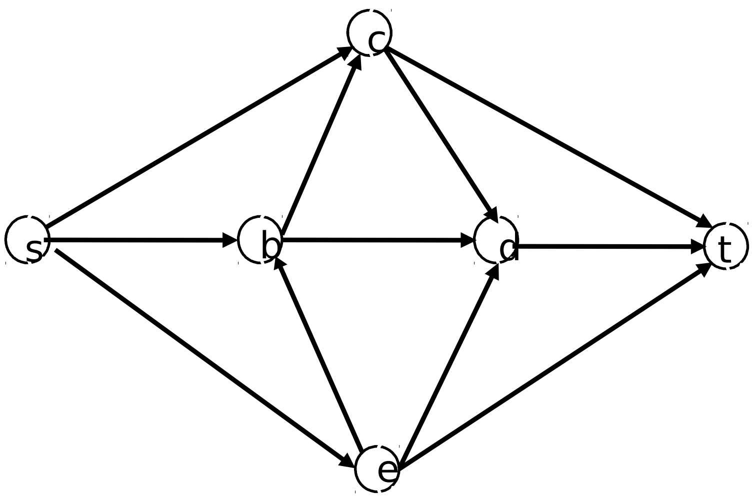

c
s
b
d
t
e


c
s
b
d
t
e


c
s
b
d
t
e


c
s
b
d
t
e


c
s
b
d
t
e


c
s
b
d
t
e

b) Si consideri il seguente grafo bipartito $G=(V,E)$ dove $V=\{u_1,u_2,\ldots,u_n,v_1,v_2,\ldots,v_n\}$, **con n dispari**, ed E contiene archi della forma (u_i , v_j) dove i e` pari e j e` dispari e archi della forma (u_k , v_p), dove k e dispari e p e` pari. Dire se G ha o meno un matching perfetto. **Giustificare la risposta.**