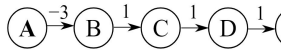
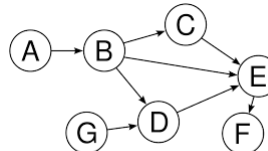
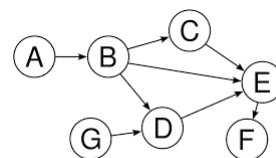
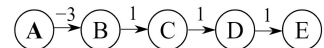


Scrivi i tuoi dati $\Rightarrow$	Cognome: .....	Nome: .....	Matricola: .....	PUNTI
ESERCIZIO 1	Risposte Esatte:	Risposte Omesse:	Risposte Errate:	

### ESERCIZIO 1: Domande a risposta multipla

**Premessa:** Questa parte è costituita da 10 domande a risposta multipla. Per ciascuna domanda vengono fornite 4 risposte, di cui soltanto una è corretta. Per rispondere utilizzare la griglia annessa, barrando con una  $\times$  la casella corrispondente alla risposta prescelta. È consentito omettere la risposta. In caso di errore, contornare con un cerchietto la  $\times$  erroneamente apposta (ovvero, in questo modo  $\otimes$ ) e rifare la  $\times$  sulla nuova risposta prescelta. Se una domanda presenta più di una risposta, verrà considerata omessa. Per tutti i quesiti verrà attribuito un identico punteggio, e cioè: risposta esatta 3 punti, risposta omessa 0 punti, risposta sbagliata -1 punto. Il voto relativo a questa parte è ottenuto sommando i punti ottenuti e normalizzando su base 30. Se tale somma è negativa, verrà assegnato 0.

- L'algoritmo INSERTION SORT, nel caso medio costa:
  - $O(n)$
  - $o(n^2)$
  - $\Omega(n^2)$
  - $\Theta(n \log n)$
- Sia dato in input l'array  $A = [3, 5, 2, 1, 8]$ ; quanti confronti tra elementi esegue l'algoritmo INSERTIONSORT2 per ordinare in ordine non decrescente  $A$ ?
  - 10
  - 5
  - 25
  - 7
- Un algoritmo ha una complessità temporale  $O(f(n))$  se:
  - Il tempo di esecuzione  $T(n)$  dell'algoritmo su uno specifico input di dimensione  $n$  verifica  $T(n) = O(f(n))$
  - Il tempo di esecuzione  $T(n)$  dell'algoritmo su ogni input di dimensione  $n$  verifica  $T(n) = O(f(n))$
  - Il tempo di esecuzione medio  $T(n)$  dell'algoritmo su un input di dimensione  $n$  verifica  $T(n) = \Theta(f(n))$
  - Nel caso migliore, il tempo di esecuzione  $T(n)$  dell'algoritmo su un input di dimensione  $n$  verifica  $T(n) = o(f(n))$
- Dato un min-heap binario di  $n$  elementi rappresentato mediante un albero binario, quale delle seguenti affermazioni è vera:
  - La procedura di *fixHeap* su un nodo foglia può essere eseguita in  $O(1)$
  - L'altezza dell'albero è  $\lceil \log n \rceil - 1$
  - Le foglie sull'ultimo livello sono  $\Theta(\log n)$
  - Il numero di nodi interni è sempre minore del numero di foglie
- Quali sono, rispettivamente, i costi per implementare le operazioni di *Insert*, *Delete*, e *Search*, in un dizionario di  $n$  elementi implementato utilizzando una lista ordinata?
  - $O(n), O(1), \Theta(n)$
  - $\Theta(n), O(1), O(n)$
  - $O(n), O(n), O(n)$
  - $O(n), O(1), O(n)$
- Dato il grafo in figura, si applichi su di esso l'algoritmo di Bellman&Ford, con nodo sorgente  $A$ . Se gli archi vengono letti ad ogni iterazione in ordine da destra a sinistra, quante iterazione sono necessarie per trovare tutte le distanze da  $A$ ?
 
  - 1
  - 2
  - 3
  - 4
- Dato il grafo in figura, quale dei seguenti non è un suo ordinamento topologico?
  - $\{A, B, G, C, D, E, F\}$
  - $\{A, G, D, B, C, E, F\}$
  - $\{A, G, B, C, D, E, F\}$
  - $\{A, B, G, D, C, E, F\}$
- Dato il grafo in figura 7, quale delle seguenti affermazioni è falsa?
  - Il grafo è aciclico
  - Il grado uscente del grafo è 3
  - Il grafo è fortemente connesso rispetto ad  $A$
  - La distanza tra  $A$  e  $G$  è  $+\infty$
- Dato un grafo connesso di  $n$  nodi ed  $m$  archi, per quali valori (asintotici) di  $m$  si ha che l'implementazione di Prim con heap di Fibonacci non è strettamente più efficiente dell'implementazione di Kruskal con alberi QuickUnion con euristica di bilanciamento *union by size*?
  - $m = \Theta(n \log n)$
  - tutti
  - nessuno
  - $m = \Theta(n)$
- Dato un grafo pesato con  $n$  vertici ed  $m$  archi, l'algoritmo di Borůvka alla prima passata aggiunge alla soluzione un numero massimo di archi pari a:
  - $n - 1$
  - $\log n$
  - 1
  - $\lceil n/2 \rceil$



## Griglia Risposte

[illegible]