Informatica Generale II - Prova teorica

A.A. 20052006

Esame: 27 aprile 2007
Codice: SVHO

1. In quale ordine partendo dal nodo C vengono visitati i nodi del grafo in figura 1 da un algoritmo di visita in ampiezza?

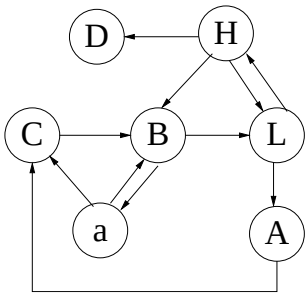

Figura 1:

   (a) CBLaADH
   (b) CBLaHAD
   (c) CBLHDAa
   (d) Nessuna risposta è accettabile;
   (e) non rispondo

2. Quale tipo di attraversamento d'albero implementa il seguente codice?

```
void foo-order(Node* l, void visit(Node*))
  StackPtr s = new Stack();

  Push(s, i);
  while(! StackIsEmpty(s)) {
    Node * h = Pop(s);

    visit(h);
    if (h->right != NULL) Push(s, h->right);
    if (h->left != NULL) Push(s, h->left);
  }
  delete s;
}
```

   (a) level-order
   (b) inorder
   (c) preorder
   (d) postorder
   (e) non rispondo

3. Le operazioni di base di stack e code sono:

(a) push e pop per stack, put e get per code
(b) push e pop per code, put e get per stack
(c) put e get
(d) push e pop
(e) non rispondo

4. Quale tra le seguenti affermazioni relative all'ordinamento a bolle (bubblesort) è *falsa*?

(a) considerato che la prima iterazione dell'algoritmo termina avendo posto in cima all'array la più piccola componente dell'array medesimo, dopo (n-1) iterazioni (n essendo la dimensione dei dati d'ingresso) l'array è sicuramente ordinato sebbene non sempre tutte le (n-1) iterazioni siano necessarie;
(b) ha complessità asintotica $\mathcal{O}(n^2)$;
(c) non presenta in alcun caso costo d'esecuzione pari a $\mathcal{O}(n)$;
(d) la complessità dell'algoritmo dipende dai valori dei dati d'ingresso;
(e) non rispondo

5. Quali degli alberi in figura 2 *non sono* alberi binari di ricerca, mentre gli altri lo sono?

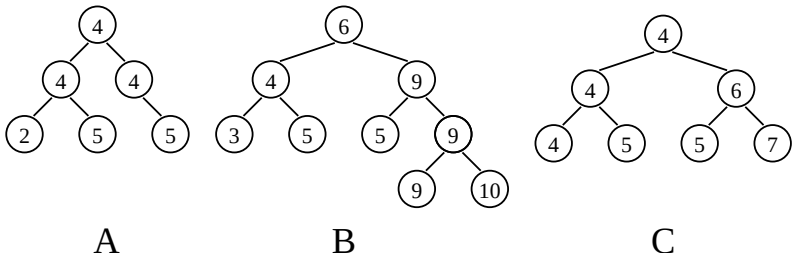

Figura 2:

(a) A,B,C
(b) A,B
(c) sono tutti alberi binari di ricerca.
(d) A,C
(e) non rispondo

6. Il costruttore è una funzione speciale che:

(a) viene definita all'interno di una struttura, ha come identificatore lo stesso nome della struttura, non ha tipo di ritorno e serve per inizializzare i campi della struttura;
(b) viene definita all'interno di una struttura e ha lo scopo di allocare dinamicamente memoria nello heap per i campi della struttura;
(c) viene definita all'interno di una struttura, ha come identificatore lo stesso della struttura, non ha tipo di ritorno e deve inizializzare tutti i campi della struttura;
(d) viene definita all'interno di una struttura, può essere identificata tramite un qualunque identificatore valido e deve ritornare un tipo identico al tipo della struttura;
(e) non rispondo

7. Qual'è l'altezza dell'albero rappresentato in figura 3?

(a) 2
(b) 5
(c) 3
(d) 1
(e) non rispondo

8. Supponendo di aver definito una struttura `Punto` e funzione `funzEsame` come di seguito:

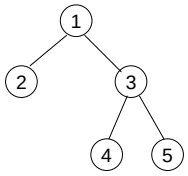

Figura 3: Albero

```
struct Punto {
  int x, y;
  Punto() { x = 0; y = 0; }
  Punto(int _x, int _y) { x = _x; y = _y; }
};

void funzEsame (Punto *i, Punto *j);
```

L'invocazione della suddetta funzione `funzEsame` avrà la forma:

(a)
```
Punto x(1,0), y(0,5);
funzEsame();
```
(b)
```
Punto x(1,0), y(0,5);
funzEsame(*x, *y);
```
(c)
```
Punto x(1,0), y(0,5);
funzEsame(x, y);
```
(d)
```
Punto x(1,0), y(0,5);
funzEsame(&x, &y);
```
(e) non rispondo

9. Si consideri un algoritmo di visita di una grafo:

   (a) usare una *marca* che segni i nodi già visitati può essere utile per migliorare l'efficienza dell'algoritmo, ma non è necessario sempre;
   (b) nessuna delle risposte è accettabile;
   (c) è preferibile non usare una *marca* che segni i nodi già visitati perché l'algoritmo potrebbe risultare appesantito da una tale operazione;
   (d) occorre usare una *marca* che segni i nodi già visitati per essere sicuri che l'algoritmo non entri in un ciclo infinito;
   (e) non rispondo

10. Si supponga di avere la sequenza di numeri *3,8,6,4,5*. Dopo *3* iterazioni soltanto di Bubblesort su tale sequenza, il risultato sarà:

   (a) 3,4,5,8,6
   (b) 3,4,8,5,6
   (c) 3,8,4,5,6
   (d) 3,4,8,6,5
   (e) non rispondo