Laboratorio di algoritmi e strutture dati

Docente: Violetta Lonati

Prova di laboratorio - Appello del 16 giugno 2021 (in presenza)

Note importanti

- Si legga attentamente il testo degli esercizi e le indicazioni su come svolgerli. Se ci sono dubbi sul significato delle richieste, è opportuno chiedere chiarimenti!
- I file utili allo svolgimento degli esercizi, e indicati nel testo, sono contenuti nell'archivio zip, nella cartella `allegati`.
- Si leggano attentamente anche le indicazioni su come preparare le risposte. Per ogni esercizio è richiesto di preparare un file: in alcuni casi si tratta di un file di testo, in altri casi di un programma in C. Per ogni esercizio viene indicato il nome con cui salvare il file; è importante rispettare questa indicazione.
- Nella prima riga di tutti i file consegnati è necessario **scrivere nome, cognome e matricola**.
- Dopo essersi autenticati, si carichino sul sito `upload.di.unimi.it` i file contenenti le risposte. I nomi dei file devono essere i seguenti:
  `es1-piuPiccolo.c`
  `es2-tree.txt`
  `es3-dipendenti.c`

1 Più piccolo

Scrivete una funzione C con prototipo

```
int h( int A[], int B[], int n, int m)
```

che, dato un vettore A di lunghezza `n` e un vettore B di lunghezza `m`, decide se ogni valore in A è strettamente più piccolo di ogni valore in B.

Ad esempio se `A` $= \{1,5,5\}$ e `B` $= \{6,5\}$ allora `h(A,B)` restituisce 0, ma per `C` = $\{7,6\}$ allora `h(A,C)` restituisce 1.

Indicate il tempo di esecuzione; si richiede una soluzione efficiente!

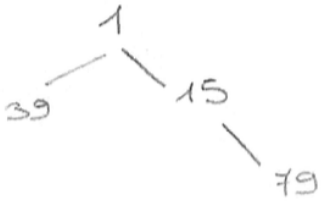
Note per la consegna. Scrivete la funzione in un file di nome `es1-piuPiccolo.c`.

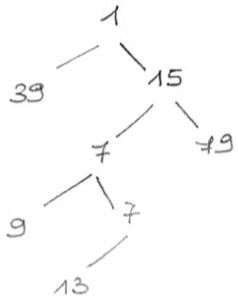
2 Funzioni misteriose - comprensione di codice, strutture dati

Considerate la porzione di codice qui sotto (contenuta anche nel file di nome `tree.c`), in cui il tipo `Bit_node` è usato per implementare i nodi di un albero binario:

```c
struct bit_node {
  int item;
  struct bit_node *l, *r;
};

typedef struct bit_node *Bit_node;


void printArray( int *a, int n ) {
        for ( int i = 0; i < n; i++ )
                printf( "%d ", a[i] );
        printf( "\n" );
}

void f_r( Bit_node root, int *path, int len ) {
        if (root == NULL )
                return;

        if ( root -> item % 2 ) {
                path[len] = root -> item;
                len++;
        }

        if ( root -> r == NULL && root -> l == NULL ) {
                printArray( path, len );
                return;
        }

        f_r( root -> l, path, len );
        f_r( root -> r, path, len );
}

void f( Bit_node root ) {
        int *path = malloc( 1000 * sizeof( int ) );
        f_r( root, path, 0 );
}
```

Analizzate le funzioni `f` e `f_r`, considerate i due alberi disegnati qui sotto, e rispondete alle domande seguenti:





1. Cosa stampa la funzione `f` se viene invocata sulla radice dell'albero di sinistra?
2. Cosa stampa la funzione `f` se viene invocata sulla radice dell'albero di destra?
3. Che altezza raggiunge lo stack se la funzione `f` viene invocata sulla radice dell'albero di destra? (si considerino solo le chiamate di funzione effettuate durante l'esecuzione di `f`).

4. In generale, che altezza raggiunge lo stack se la funzione viene invocata sulla radice di un albero binario qualunque?
5. In generale, quante righe stampa la funzione se invocata sulla radice di un albero binario qualunque?
6. Cosa stampa la funzione se invocata sulla radice di un qualunque albero binario che contiene solo numeri pari?
7. Completate la frase seguente:

   Se `root` è il puntatore alla radice di un albero binario, allora l'invocazione della funzione `f(root)` produce in output ...

Note per la consegna. Scrivete le vostre risposte in un file di testo e salvatelo con il nome `es2-tree.txt`.

3 Dipendenti

Nella multinazionale Algoré il lavoro è organizzato in maniera gerarchica. Ogni dipendente è inquadrato in un certo *livello di impiego*. Tranne i dipendenti di *massimo livello*, ogni dipendente ha un *supervisore*, di cui è detto *subordinato*. Un dipendente può avere 0, 1, o più subordinati.

Si considerino i seguenti compiti.

(a) Dato un certo dipendente, stampare l'elenco dei suoi subordinati.

(b) Contare quanti sono i dipendenti che non hanno alcun subordinato.

(c) Dato un certo dipendente, individuare chi è il suo supervisore.

(d) Dato un certo dipendente, stampare la lista dei dipendenti che si trovano sopra di lui gerarchicamente, partendo dal suo supervisore e risalendo la gerarchia fino a un dipendente di massimo livello.

(e) Stampare l'elenco di tutti i dipendenti –non importa l'ordine–, indicando per ciascuno chi è il suo supervisore (tranne che nel caso di dipendenti di massimo livello).

(f) Stampare l'elenco di tutti i dipendenti, in ordine di livello (prima tutti quelli di livello massimo, poi tutti quelli subordinati a quelli di livello massimo, ecc); non importa l'ordine tra i dipendenti di pari livello.

Esempio Anna è supervisore di Bruno, Carlo e Daniela. Bruno è supervisore di Enrico e Francesco. Gianni è supervisore di Harry. Francesco è supervisore di Irene. Il numero di dipendenti senza subordinati è 4 (Carlo, Daniela, Enrico, Harry). La lista di dipendenti che si trovano sopra Irene è: Francesco, Bruno, Anna. Questo è l'elenco dei dipendenti in ordine di livello: A, G (massimo livello), B, D, H, C (subordinati di dipendenti di massimo livello, non importa il loro ordine), F, E, I.

Modellazione e progettazione Assumendo che gli N dipendenti di Algoré siano indicati con un numero progressivo da 0 a $N-1$, svolgete i seguenti punti.

1. Modellate la situazione con una struttura dati opportuna:
   - descrivete come si possono rappresentare i dipendenti e le loro relazioni con la struttura dati scelta;
   - riformulate, nei termini della struttura dati scelta, ciascuno dei compiti enunciati sopra.
2. Descrivete come è opportuno implementare la struttura dati scelta.
3. Per ciascun compito, progettate e descrivete un algoritmo che consente di svolgere il compito, sfruttando le scelte di progettazione e implementazione fatte precedentemente. Gli algoritmi possono essere descritti a parole o in pseudocodice; può essere opportuno fare riferimento ad algoritmi noti.

4. Spiegate come modifichereste le risposte ai punti procedenti se i dipendenti fossero identificati da un nome (dunque da una stringa) invece che un numero progressivo.

Implementazione Assumendo ancora che gli N dipendenti di Algoré siano indicati con un numero progressivo da 0 a $N-1$. definite, in linguaggio C, uno o più tipi di dati utili a rappresentare i dipendenti e le loro relazioni, in base alle scelte fatte ai punti precedenti. Scrivete quindi una funzione C per ciascuno dei compiti enunciati sopra. Le funzioni devono avere questi nomi:

- `stampaSubordinati` per il compito (a)
- `supervisore` per il compito (c).
- `quantiSenzaSubordinati` per il compito (b)
- `stampaImpiegatiSopra` per il compito (d)
- `stampaImpiegatiConSupervisore` per il compito (e)
- `stampaPerLivello` per il compito (f) **[richiesta facoltativa]**

Note per la consegna. Raccogliete le definizioni di tipo e le funzioni in un file con nome `es3-dipendenti.c`. Mettete le risposte alle domande di modellazione e progettazione come commenti all'inizio del file.