Laboratorio di Algoritmi e Strutture Dati I Docenti: Murano / Cutugno

-- -- -- 5 LUGLIO 2006 -- -- --

Laurea in Informatica Università degli Studi di Napoli "Federico II"

Nome e Cognome Numero di Matricola:

Spazio riservato alla correzione

1	2	3	Totale
/8	/10	/12	/30

Non utilizzate altri fogli. Utilizzate soltanto lo spazio sottostante. Fogli differenti non saranno presi in considerazione per la correzione. Non scrivere a matita

Gli studenti che hanno frequentato e consegnato i progetti devono svolgere solamente gli esercizi 1, 2 e 3. Tutti gli altri devono anche svolgere anche l'esercizio 4 e l'esercizio 5.

1. Si considerino due code **C1 e C2**, implementate con array. Supponendo che C1 e C2 abbiano inizialmente entrambi n elementi, si implementi un algoritmo **ricorsivo** che prende in input **C1 e C2** come array di dimensione **MAX** (cioè, C1[MAX] e C2[MAX]) e restituisce una lista puntata definita come

```
struct elemento {
        int inf;
        struct elemento *next;}

struct elemento *Lista;
```

scegliendo solo le chiavi dispari da C1 e le chiavi pari da C2 e riordini le chiavi al volo durante l'inserimento.

Possibilmente, si operi in modo che alla fine dell'operazione le due code risultino invariate.

Si implementi l'algoritmo tenendo conto che le code sono strutture dati il cui **accesso a nodi interni è proibito**. In particolare, si richiede che l'accesso alle code nell'algoritmo avvenga **solo tramite funzioni indipendenti dalla implementazione** delle code stesse. Tutte **le funzioni devono essere opportunamente implementate**.

Valutare la complessità di tempo e di spazio dell'intero algoritmo.

2. Si consideri una lista di numeri interi **Lista1** implementate come lista doppiamente puntata e non circolare, utilizzando la seguente struttura

```
struct elemento {
        int inf;
        struct elemento *next;
        struct elemento *succ;}

struct elemento *Lista;
```

a) Si implementi la funzione ricorsiva **void gira(struct elemento *L1)** che elimina dalla lista **Lista1** in successione tutti gli elementi eliminando alternativamente la chiave minore e quella maggiore ad ogni giro. La funzione deve terminare se la lista si svuota completamente o se in un qualsiasi momento risultano presenti 3 chiavi consecutive uguali. Dal momento che la lista è circolare il concetto di consecutività si estende agli estremi.

b) Si consideri la seguente variante alla funzione **gira():** si considerino due liste circolari L1 ed L2 definite come sopra, un processo attraversa L1, i passaggi dei puntatori sono monitorati con messaggi a schermo, ogni volta che in L1 viene identificata una chiave **int Start_Forward** data, il processo viene sospeso, la chiave Start_Forward viene incrementata di 1 e viene dato il via ad un processo identico su L2, allo stesso modo, ogni volta che in L2 viene identificata una chiave **int Start_Back** il processo viene sospeso, la chiave Start_back viene decrementata di 1 e si continua al rimbalzo. Discutere quali sono le possibili condizioni di terminazione dell'algoritmo, individuare il caso peggiore e discuterne la complessità.

3. Sia **G** un grafo non orientato pesato con pesi positivi, di **n** vertici 0, 1,…, n-1 e rappresentato con lista di adiacenza utilizzando la seguente struttura:

```
typedef struct graph {
        int   nv;
        edge   **adj; } graph;

        graph *G;
```

```
typedef struct edge  {
        int   key;
        int   peso;
        struct  edge   *next; } edge;
```

scrivere in linguaggio C una funzione che preso in input il grafo **G**,

a. data una coppia di nodi (x,y), identifichi ogni possibile percorso (senza ripetizioni) di nodi che porta da x a y, ne indichi il costo totale di ogni percorso, e ne rimuova l'ultimo arco incontrato nel percorso più costoso;
b. dato il percorso meno costoso fra quelli precedentemente riportati, aggiunga un nuovo nodo w nel grafo e un nuovo arco fra il nodo y il nuovo nodo w.
c. Studiare la complessità della funzione implementata al punto a e b.

Gli studenti che non hanno consegnato il progetto devono risolvere i seguenti esercizi aggiuntivi:

Spazio riservato alla correzione

1	2	3	4	5	Totale
5/	5/	8/	7/	5	30/

4. Dato un albero binari di ricerca T implementato con la seguente struttura a puntatori:

```
struct nodo {
        int inforadice;
        struct nodo *left;
        struct nodo *right;}

struct nodo *T;
```

   implementare una funzione in linguaggio C ricorsiva che, dato un intero **n** di valore massimo pari a quello della profondità dell'albero, restituisca l'elenco orizzontale di tutti i nodi di livello n indicando raggruppandoli per nonni.
   Descrivere la complessità della funzione implementata.

5. Si descriva un algoritmo di visita di grafo a piacere e se ne discuta la complessità.