02MNO Algoritmi e Programmazione 01JKE APA I / 01JKF APA II

Appello del 27/06/2014 - Prova di teoria (12 punti)

1. (1 punto)

Si ordini in maniera ascendente mediante merge-sort il seguente vettore di interi:

11 9 12 1 13 3 39 23 29 17 21 10 18 16 8 100

Si indichino i passaggi rilevanti.

2. (2 punti)

Si determini mediante un algoritmo greedy un codice di Huffman ottimo per i seguenti caratteri con le frequenze specificate:

a: 6 b: 16 c: 4 d: 3 e: 22 f: 11 g: 14 h: 9 i: 8 l: 10

3. (1 punto)

Si esprima in notazione prefissa e postfissa la seguente espressione aritmetica mediante visita dell'albero binario corrispondente: A + ((B – C) / D) * E

4. (2 punti)

Sia data la sequenza di chiavi T con eventuale $T_1HELA_1S_1T_2S_2T_3A_2ND$, dove ciascun carattere è individuato dal suo ordine progressivo nell'alfabeto (A=1, ..., Z=26) con eventuale pedice. Si riporti la struttura di una tabella di hash di dimensione 19, inizialmente supposta vuota, in cui avvenga l'inserimento della sequenza indicata. Si supponga di utilizzare il linear chaining con la funzione di hash $h(k) = k \bmod 19$.

5. (1 punto)

Si effettuino, secondo l'ordine specificato, le seguenti operazioni su un BST supposto inizialmente vuoto (+ indica una inserzione in foglia, − una cancellazione):

+10 +4 +3 +16 +21 +7 +11 +1 +8 +19 +12 +15 -4 -10 -7

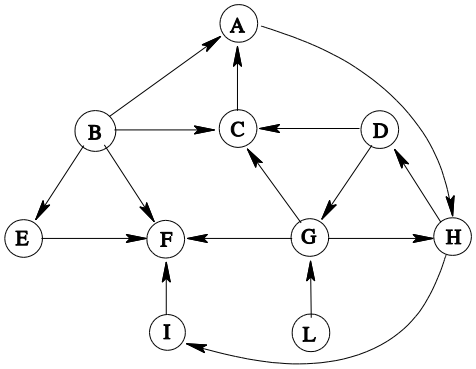
6. (2 punti)

Si inseriscano in sequenza nella radice di un BST, inizialmente supposto vuoto, le chiavi:

10 4 3 16 21 7 11 1

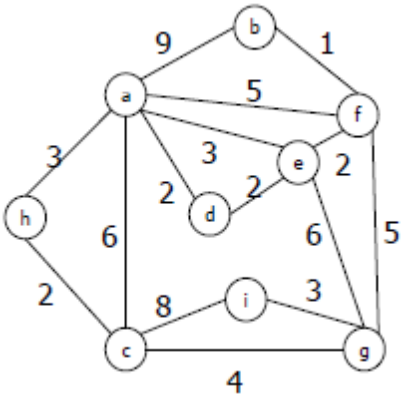
7. (2 x 0.5 punti)

Si rappresenti il seguente grafo orientato come lista delle adiacenze (**0.5 punti**) e come matrice delle adiacenze (**0.5 punti**).



8. (2 punti)

Sia dato il seguente grafo non orientato pesato:



se ne determini un minimum spanning tree applicando l'algoritmo di Prim a partire dal vertice **h**, disegnando l'albero e ritornando come risultato il valore del peso minimo. Si esplicitino i passi intermedi.

02MNO Algoritmi e Programmazione 01JKE APA I / 01JKF APA II

Appello del 27/06/2014 - Prova di programmazione (12 punti)

1. (2 punti)

Siano dati 2 vettori di interi `vet1` e `vet2` di dimensioni note `d1` e `d2`, rispettivamente. Si realizzi in C la funzione di prototipo

```
int ricerca(int *vet1, int *vet2, int d1, int d2);
```

che ricerchi se nel vettore `vet1` compare almeno una volta il vettore `vet2` come sottovettore. In caso affermativo, la funzione ritorni l'indice della prima cella della prima occorrenza di `vet2`, in caso negativo ritorni -1.

Esempio

Se `vet1` è

0	15	12	21	7	25	32	1

e `vet 2` è

21	7	25

la ricerca ha esito positivo e la funzione ritorna l'intero 3.

2. (4 punti)

Si implementi come ADT di I categoria una lista bi-linkata di interi con le seguenti 2 funzioni:

- `list_insert (list_t *l, int chiave, int estremo)` dove se il parametro `estremo` vale 0 l'inserzione avviene in testa, se vale 1 in coda
- `list_display (list_t *l, int modo)` dove se il parametro `modo` vale 0 la visualizzazione a video è fatta dalla testa alla coda, se vale 1 dalla coda alla testa.

dove

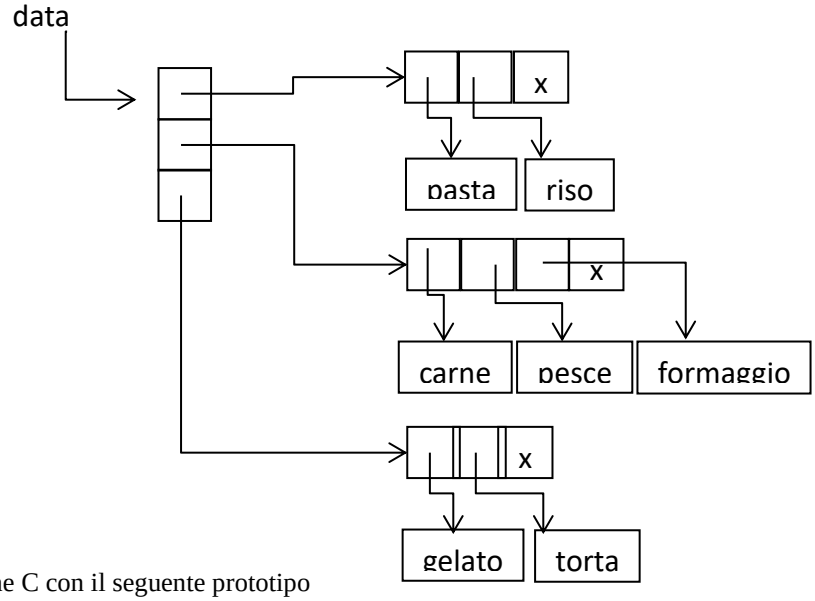
```
typedef struct {nodo_t *head; nodo_t *tail; } list_t;
```

mentre `nodo_t` è una `struct` la cui definizione è richiesta al candidato.

3. (6 punti)

Un ristorante serve un menù a prezzo fisso composto da `n` portate. Per ciascuna portata, il cliente sceglie obbligatoriamente un piatto tra un certo numero di piatti disponibili. Le informazioni sono memorizzate in un vettore `data` di `n` elementi, ciascuno dei quali è un puntatore a un vettore di puntatori a stringhe. Quest'ultimo vettore elenca i piatti disponibili per una certa portata. Il numero di questi piatti per portata non è dato esplicitamente, ma quando l'elemento contiene un puntatore a NULL, l'elenco dei piatti di quella portata si considera terminato. Ogni stringa rappresenta un piatto disponibile.

Esempio:



Si scriva una funzione C con il seguente prototipo

```
void build_menu(char **data[], int n);
```

che, utilizzando un algoritmo ricorsivo, determini tutte le composizioni possibili del menu e le visualizzi. Nel caso dell'esempio, l'output deve essere:

```
(pasta, carne, gelato), (pasta, carne, torta), (pasta, pesce, gelato),
(pasta, pesce, torta), (pasta, formaggio, gelato), (pasta, formaggio, torta)
(riso, carne, gelato), (riso, carne, torta), (riso, pesce, gelato),
(riso, pesce, torta), (riso, formaggio, gelato), (riso, formaggio, torta)
```

02MNO Algoritmi e Programmazione 01JKE APA I / 01JKF APA II

Appello del 27/06/2014 - Prova di programmazione (18 punti)

Un archivio produttore-modello-accessori è organizzato come segue:

- il primo file memorizza l'elenco dei produttori, in ragione di un produttore per ciascuna riga. Il numero di produttori non è noto a priori, né è presente in alcun modo nel file. Per ciascun produttore viene riportato il suo nome e il nome del file che contiene i modelli da esso prodotti
- i file successivi, tanti quanti sono i produttori, riportano per ogni produttore quali modelli esso produce in ragione di un modello per ciascuna riga. Per ogni modello viene riportato il suo nome e il nome del file che contiene gli accessori relativi a quel modello
- i file successivi, tanti quanti sono i modelli, riportano per ogni modello la lista degli accessori di cui dispone in ragione di un accessorio per ciascuna riga. Per ogni accessorio viene riportato il suo nome e il suo costo in Euro.

Si osservi che tutti i nomi sono stringhe alfanumeriche nelle quali gli spazi sono sostituiti dal carattere di sottolineatura "_" e di lunghezza massima uguale a 100. Inoltre tutti i nomi del produttore, del modello e dell'accessorio, sono univoci all'interno dell'intera base dati.

Esempio

```
Produttori.txt
Audi  Audi_modelli.txt
BMW   BMW_modelli.txt
FIAT  FIAT_modelli.txt
Ford  Ford_modelli.txt
Renault Renault_modelli.txt
Toyota Toyota_modelli.txt
Volkswagen Volkswagen_modelli.txt

FIAT_modelli.txt
Cinquecento Cinquecento_accessori.txt
Bravo Bravo_accessori.txt
Doblò Doblò_accessori.txt
Freemont Freemont_accessori.txt
Panda Panda_accessori.txt
Punto Punto_accessori.txt

Bravo_accessori.txt
Barre_portasurf 150,00
Spoiler 220,50
Cerchi_lega 75,25
Vivavoce 450,50
Sensori_parcheggio 700,50
```

Si scriva un programma in grado di:

- ricevere il nome del file produttori sulla riga di comando
- leggere i file dei modelli e degli accessori e memorizzare tutti i dati relativi a produttori, modelli e accessori in un'opportuna struttura dati
- realizzare un menu con i seguenti comandi:
  - leggere da tastiera il nome di un produttore, visualizzando (a video) l'elenco dei modelli da esso prodotti con complessità al più logaritmica nel numero di produttori
  - leggere da tastiera il nome di un modello, visualizzando (a video) l'elenco degli accessori ad esso associati con complessità al più logaritmica nel numero totale di modelli
  - cancellare un produttore, i modelli ad esso relativi e gli accessori associati a quei modelli
  - cancellare un modello e gli accessori ad esso associati
  - cancellare un accessorio
  - incorporare il produttore2 nel produttore1: tutti i modelli ed i relativi accessori del produttore2 sono attribuiti al produttore1, il produttore2 viene cancellato dalla base dati
  - uscita.

Le strutture dati siano gestite come ADT di I categoria.

PER ENTRAMBE LE PROVE DI PROGRAMMAZIONE (18 o 12 punti):

- ***indicare nell'elaborato e nella relazione (oltre a nome, cognome e numero di matricola) anche il nome del corso per cui si sta sostenendo l'esame (AP, APA I+II).***
- ***È consentito utilizzare chiamate a funzioni standard, quali ordinamento per vettori, inserzione/estrazione/ricerca relative a FIFO, LIFO, liste, BST, tabelle di hash e altre strutture dati, considerate come librerie esterne. Gli header file delle librerie utilizzate devono essere allegati all'elaborato. Le funzioni richiamate, inoltre, dovranno essere incluse nella versione del programma allegata alla relazione.***
- ***Consegna delle relazioni (per entrambe le tipologie di prova di programmazione): entro mercoledì 02/07/2014, alle ore 23:59, via e-mail all'indirizzo: danilo.vendraminetto@polito.it, usando come subject (oggetto) la stringa APA#<m>, essendo <m> il proprio numero di matricola. L'allegato alla mail deve essere costituito da un unico file: un archivio compresso, contenente sia il codice corretto, sia la relazione (NO eseguibili). QUALORA IL CODICE SPEDITO CON LA RELAZIONE NON COMPILI CORRETTAMENTE, VERRÀ APPLICATA UNA PENALIZZAZIONE. Si ricorda che la valutazione del compito viene fatta senza discussione o esame orale, sulla base dell'elaborato svolto in aula. Non verranno corretti i compiti di cui non sarà stata inviata la relazione nei tempi stabiliti.***

PER I LAUREANDI DELLA SESSIONE DI LUGLIO 2014: per il calendario di invio relazioni ed esami orali vedere avviso sula pagina del corso 02MNO AP sul Portale.