

LASD-settembre

Aniello Murano, Silvia Stranieri

Settembre 2022

1 Esercizio 1 (10 punti)

Data una lista doppiamente puntata e circolare, scrivere una funzione che rimpiazzi ogni elemento in posizione dispari con la media tra gli elementi alla sua destra e alla sua sinistra. Per la numerazione delle posizioni, si assume che il primo elemento si trovi in posizione 1.

2 Esercizio 2 (10 punti)

Dato un albero e una chiave k , scrivere una funzione che verifichi che l'albero sia binario di ricerca e restituiscia il valore minimo del sottoalbero radicato in k .

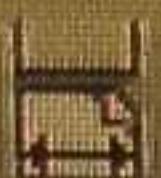
3 Esercizio 3 (10 punti)

Dato un grafo orientato pesato rappresentato con liste di adiacenza, scrivere una funzione che rimuova tutti gli archi con peso pari e tutti i nodi con grado uscente dispari.





50%



Laboratorio di Algoritmi e Strutture Dati I

Docente: Murano

--- --- --- 12 Giugno 2019 --- ---

Laurea in Informatica

Università degli Studi di Napoli "Federico II"

Nome e Cognome

Numero di Matricola:

Spazio riservato alla correzione

1	2	3	Totale
7/8	7/8	7/14	7/30

Per tutti gli esercizi, descrivere la complessità asintotica delle funzioni implementate

- Si consideri una coda Q, implementata con array Q[MAX+2], riempita con interi. Si implementi la funzione ricorsiva void toglinegativi-positivi(int Q[])) che, utilizzando una libreria di funzioni di accesso alla coda, prendendo in input Q, restituisce la coda con gli elementi nello stesso ordine, rimuovendo, secondo la scelta, tutti i numeri negativi o positivi. Stampare a video la coda prima e dopo la modifica.
- Si considerino due liste di numeri interi Lista1 e Lista2 implementate come liste doppiamente puntate e non circolari, utilizzando la seguente struttura

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

```
struct elemento *Lista1,*Lista2;
```

Si implementi una funzione ricorsiva che presi in input Lista1 e Lista2

- rimuova da Lista1 i negativi e si inseriscano in testa alla Lista2
- rimuova da Lista2 i positivi e li inserisca in testa alla lista1.
- Restituisca le due liste modificate.

- Siano G e H due grafi orientati pesati entrambi con pesi positivi, di n vertici 0, 1, ..., n-1 e rappresentati con liste di adiacenza utilizzando la seguente struttura:

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

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

- scrivere in linguaggio C una funzione che, permetta di calcolare il grado adiacente e incidente di G e H.
- scrivere in linguaggio C una funzione che, presi in input G e H permetta di creare un terzo grafo T con la seguente regola: In T ci saranno archi presenti in G e H con peso dato dalla somma dei pesi corrispondenti in G e H, che sia pari e superiore a 10.

Laboratorio di Algoritmi e Strutture Dati I

Docente: Murano

— — — 9 GENNAIO 2020 — — —

Laurea in Informatica

Università degli Studi di Napoli "Federico II"

Nome e Cognome

Numero di Matricola:

Spazio riservato alla correzione

1	2	2 ridotto	Totale
/12	/12	/12	/36

Per tutti gli esercizi, descrivere la complessità asintotica delle funzioni implementate.

1. Siano Q_0 e Q_1 due code statiche riempite con interi positivi e S_1 e S_2 due stack riempiti il primo con valori Booleani e il secondo con interi positivi. Si implementi una funzione che dopo aver controllato che Q_0 , Q_1 , S_1 e S_2 hanno lo stesso numero di elementi e riempiti correttamente, rinnova ricorsivamente da Q_i l'elemento in posizione j , dove i è dato come top di S_1 e j come top di S_2 (valori i e j vanno rimossi ad ogni chiamata ricorsiva). Ad ogni chiamata ricorsiva controllare anche che non si accede fuori dalle code né che un elemento sia stato già eliminato
Esempio: $Q_0 = 1, 2, 3, 4, 5$ – $Q_1 = 11, 12, 13, 14, 15$ – $S_1 = 0, 1, 0, 1, 0$ $S_2 = 1, 2, 1, 6, 4$. Allora $Q_1 = 2, 3, 5$ – $Q_2 = 11, 13, 14, 15$. Importante non cambiare l'ordine delle code.
E' importante che le strutture si possano riempire da tastiera, che si utilizzino solo funzioni di libreria per l'accesso alle code ed agli stack e che sia possibile visualizzare i contenuti delle strutture di dati prima e dopo le operazioni.
2. Si consideri quattro liste doppiaamente punitate non circolari L_1, L_2, L_3, L_4 . A turno rimuovere il minimo dalle 4 liste, fino a che una delle liste non diventi vuota. Ad ogni turno c'è una priorità che avanza di 1. All'inizio è $1 > 2 > 3 > 4$, al secondo turno è $2 > 3 > 4 > 1$. A parità di minimi pede (si toglie l'elemento dalla lista con priorità minore).
E' importante che le liste si possano riempire da tastiera e che sia possibile visualizzare i contenuti delle liste date prima e dopo le operazioni. Facoltativo: Si può usare lo stack S del primo esercizio invece di L_2 , se si preferisce.
3. Siano G_1, G_2 e G_3 tre grafi orientati e pesati, entrambi di n nodi. Si controlli che G_3 è l'unione (con somma di pesi) di G_1 e G_2 .
Si crei una interfaccia che sia in grado di creare i grafi G_1, G_2 e G_3 , aggiungere e togliere archi, modificare i pesi dei singoli archi, stampare i grafi in ogni momento, avviare la funzione dell'esercizio.





50%



Laboratorio di Algoritmi e Strutture Dati I

Docente: Murano

--- --- --- 03 Settembre 2019 --- --- ---

Laurea in Informatica

Università degli Studi di Napoli "Federico II"

Nome e Cognome

Numero di Matricola:

Spazio riservato alla correzione

1	2	3	Totale
/10	/6	/14	/30

Per tutti gli esercizi, descrivere la complessità asintotica delle funzioni implementate

1. Si consideri uno stack implementato con array $S[MAX+1]$, riempito con interi. Si implementi la funzione ricorsiva void $minmax(int S[])$ che, utilizzando una libreria di funzioni di accesso allo stack, restituisce S con gli elementi riordinati nel seguente modo. Partendo dalla base ogni terna avrà il mediano eliminato e il minimo sotto il massimo o viceversa a seconda che sia una terna dispari o pari. Se l'ultima terna ha meno di tre elementi, non sarà eliminato nessuno valore, ma solo eseguito l'ordinamento. Si crei lo stack da tastiera e si stampi lo stack modificato.
Esempio: stack iniziale 3|7|5|4|9|8|2|6 (3 in base) – risultato 3|7|9|4|2|6
2. Si considerino due alberi T_1 e T_2 . Dopo aver verificato che T_1 e T_2 sono alberi binari di ricerca, verificare T_1 è un sottoalbero di T_2 .
3. Siano G e H due grafi orientati pesati entrambi con pesi positivi, di n vertici $0, 1, \dots, n-1$ e rappresentati con liste di adiacenza utilizzando la seguente struttura:

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

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

- scrivere in linguaggio C una funzione che, permetta di fare la differenza tra G e H . Archi negativi vanno rimossi. Nodi con peso totale degli archi uscenti meno di 10 vanno rimossi (10).
- Scrivere la trasposta del grafo ottenuto e restituirla sia con rappresentazione a lista che a matrice (4).

Tutti gli esercizi devono prevedere una interfaccia per la gestione dei dati: riempimento manuale/random, modifica del contenuto della struttura dati, esecuzione dell'esercizio, stampa prima e dopo l'esecuzione.



50%



-- -- -- 12 Giugno 2019 -- -- --

Nome e Cognome

Numero di Matricola:

Spazio riservato alla correzione

1	2	3	Total
/8	/8	/14	/30

Per tutti gli esercizi, descrivere la complessità asintotica delle funzioni implementate

1. Si consideri una coda Q, implementata con array Q[MAX+2], riempita con interi. Si implementi la funzione ricorsiva void toglinegativi-positivi(int Q[], int n) che, utilizzando una libreria di funzioni di accesso alla coda, prendendo in input Q, restituisce la coda con gli elementi nello stesso ordine, rimuovendo, secondo la scelta, tutti i numeri negativi o positivi. Stampare a video la coda prima e dopo la modifica.
2. Si considerino due liste di numeri interi Lista1 e Lista2 implementate come liste doppiamente puntate e non circolari, utilizzando la seguente struttura

```
struct elemento {  
    struct elemento *prev;  
    int inf;  
    struct elemento *next;};  
  
struct elemento *Lista1,*Lista2;
```

Si implementi una funzione ricorsiva che presi in input Lista1 e Lista2

- a. rimuova da Lista1 i negativi e si inseriscano in testa alla Lista2
- b. rimuova da Lista2 i positivi e li inserisca in testa alla lista1.
- c. Restituisca le due liste modificate.

3. Siano G e H due grafi orientati pesati entrambi con pesi positivi, di n vertici 0, 1, ..., n-1 e rappresentati con liste di adiacenza utilizzando la seguente struttura:

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

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

- a. scrivere in linguaggio C una funzione che, permetta di calcolare il grado adiacente e incidente di G e H.
- b. scrivere in linguaggio C una funzione che, presi in input G e H permetta di creare un terzo grafo T con la seguente regola: In T ci saranno archi presenti in G e H con peso dato dalla somma dei pesi corrispondenti in G e H, che sia pari e superiore a 10.



— — — 9 GENNAIO 2020 — — —

Nome e Cognome

Numero di Matricola:

Spazio riservato alla correzione

1	2	2-ridotto	Totale
/12	/12	/12	/36

Per tutti gli esercizi, descrivere la complessità asintotica delle funzioni implementate.

- Siano Q_0 e Q_1 due code statiche riempite con interi positivi e S_1 e S_2 due stack riempiti il primo con valori Booleani e il secondo con interi positivi. Si implementi una funzione che dopo aver controllato che Q_0 , Q_1 , S_1 e S_2 hanno lo stesso numero di elementi e riempiti correttamente, rimuova ricorsivamente da Q_i l'elemento in posizione j , dove i è dato come top di S_1 e j come top di S_2 (valori i e j vanno rimossi ad ogni chiamata ricorsiva). Ad ogni chiamata ricorsiva controllare anche che non si accede fuori dalle code né che un elemento sia stato già eliminato
Esempio: $Q_0 = 1, 2, 3, 4, 5$ – $Q_1 = 11, 12, 13, 14, 15$ – $S_1 = 0, 1, 0, 1, 0$ $S_2 = 1, 2, 1, 6, 4$. Allora $Q_1 = 2, 3, 5$ – $Q_2 = 11, 13, 14, 15$. Importante non cambiare l'ordine delle code.

E' importante che le strutture si possano riempire da tastiera, che si utilizzino solo funzioni di libreria per l'accesso alle code ed agli stack e che sia possibile visualizzare i contenuti delle strutture di dati prima e dopo le operazioni.

- Si consideri quattro liste doppiamente puntate non circolari L_1, L_2, L_3, L_4 . A turno rimuovere il minimo dalle 4 liste, fino a che una delle liste non diventi vuota. Ad ogni turno c'è una priorità che avanza di 1. All'inizio è $1 > 2 > 3 > 4$, al secondo turno è $2 > 3 > 4 > 1$. A parità di minimi pede (si toglie l'elemento dalla lista con priorità minore).

E' importante che le liste si possano riempire da tastiera e che sia possibile visualizzare i contenuti delle liste date prima e dopo le operazioni. Facoltativo: Si può usare lo stack S del primo esercizio invece di L_2 , se si preferisce.

- Siano G_1, G_2 e G_3 tre grafi orientati e pesati, entrambi di n nodi. Si controlli che G_3 è l'unione (con somma di pesi) di G_1 e G_2 .

Si crei una interfaccia che sia in grado di creare i grafi G_1 , G_2 e G_3 , aggiungere e togliere archi, modificare i pesi dei singoli archi, stampare i grafi in ogni momento, avviare la funzione dell'esercizio.



50%

**Laboratorio di Algoritmi e Strutture Dati I****Docente: Murano****--- --- --- 03 Settembre 2019 --- --- ---****Laurea in Informatica****Università degli Studi di Napoli "Federico II"****Nome e Cognome****Numero di Matricola:****Spazio riservato alla correzione**

1	2	3	Totale
7/10	7/6	7/14	7/30

Per tutti gli esercizi, descrivere la complessità asintotica delle funzioni implementate

1. Si consideri uno stack implementato con array $S[MAX+1]$, riempito con interi. Si implementi la funzione ricorsiva $void minmax(int S[])$ che, utilizzando una libreria di funzioni di accesso allo stack, restituisce S con gli elementi riordinati nel seguente modo. Partendo dalla base ogni terna avrà il mediano eliminato e il minimo sotto il massimo o viceversa a seconda che sia una terna dispari o pari. Se l'ultima terna ha meno di tre elementi, non sarà eliminato nessuno valore, ma solo eseguito l'ordinamento. Si crei lo stack da tastiera e si stampi lo stack modificato.
Esempio: stack iniziale 3|7|5|4|9|8|2|6 (3 in base) – risultato 3|7|9|4|2|6
2. Si considerino due alberi T_1 e T_2 . Dopo aver verificato che T_1 e T_2 sono alberi binari di ricerca, verificare T_1 è un sottoalbero di T_2 .

3. Siano G e H due grafi orientati pesati entrambi con pesi positivi, di n vertici $0, 1, \dots, n-1$ e rappresentati con liste di adiacenza utilizzando la seguente struttura:

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

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

- a. scrivere in linguaggio C una funzione che, permetta di fare la differenza tra G e H . Archi negativi vanno rimossi. Nodi con peso totale degli archi uscenti meno di 10 vanno rimossi (10).
- b. Scrivere la trasposta del grafo ottenuto e restituirla sia con rappresentazione a lista che a matrice (4).

Tutti gli esercizi devono prevedere una interfaccia per la gestione dei dati: riempimento manuale/random, modifica del contenuto della struttura dati, esecuzione dell'esercizio, stampa prima e dopo l'esecuzione.

Traccia LASD-Giugno 2022

Silvia Stranieri

Luglio 2022

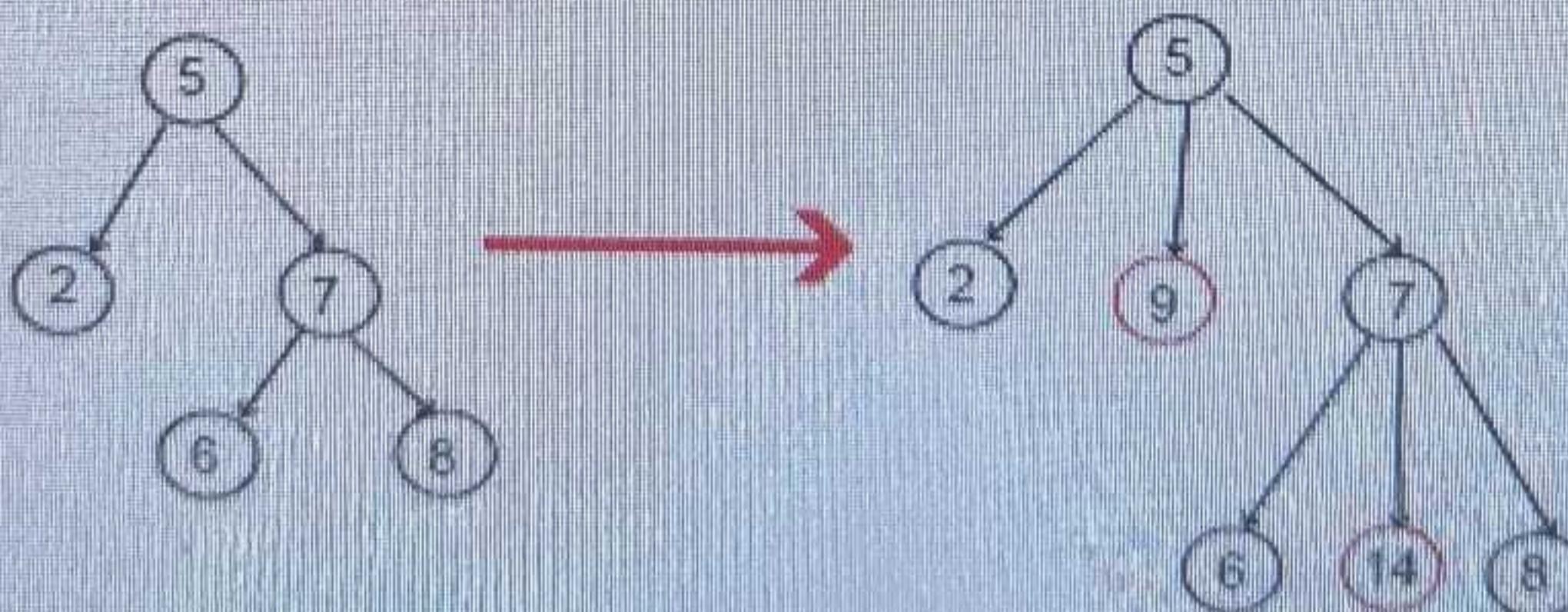
1 Esercizio 1: 10 punti

Scrivere una funzione che, data una coda Q di interi, modifichi Q in modo che sostituisca ad ogni coppia di valori pari consecutivi, il doppio della loro somma. La funzione restituisce la coda modificata.

Esempio: $Q = 2|4|6|7|6|8, Q = 12|6|28$

2 Esercizio 2: 10 punti

Scrivere una funzione che, dato un ABR T , verifichi che T sia un ABR e costruisca un albero T' ternario in modo che, per ogni nodo in T che abbia entrambi i figli, si aggiunge in T' un terzo figlio *middle*, che è un nodo foglia, contenente la somma delle chiavi dei due fratelli.



Esempio:

3 Esercizio 3: 12 punti

Scrivere una funzione che, dati due grafi orientati posati G e H e una lista L , crea un terzo grafo T risultante dall'unione di G e H , con la regola che l'arco

Traccia LASD-Giugno 2022

Aniello Murano, Silvia Stranieri

Giugno 2022

1 Esercizio 1: 10 punti

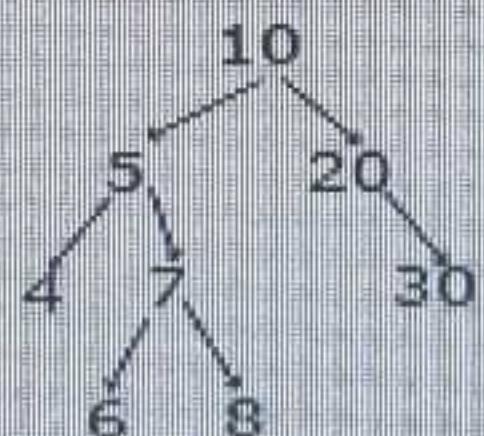
Scrivere una funzione che, data una stringa s , restituisca una lista L doppiamente puntata contenente tutte le vocali di s in ordine inverso a come compaiono in s . Inoltre, si scriva una funzione ricorsiva che elimini da L le ripetizioni consecutive.

Esempio:

```
s = "esempio"  
output = L = o ↳ i ↳ e ↳ NULL
```

2 Esercizio 2: 10 punti

Scrivere una funzione che, dato un ABR T e un intero x , verifichi che T sia un ABR e restituisca la chiave minima del sottoalbero radicato in x .



Esempio:

```
x = 7  
output = "si", 6
```

3 Esercizio 3: 12 punti

Scrivere una funzione che, dati due grafi G e H , verifichi se H è un sottografo di G . Si ricorda che $H = \langle V', E' \rangle$ si dice sottografo di $G = \langle V, E \rangle$ se $V' \subseteq V$ e $\forall (u, v) \in E', (u, v) \in E$.

Laboratorio di Algoritmi e Strutture Dati I**Docente: Murano****— — — FEBBRAIO 2020 — — —****Laurea in Informatica****Università degli Studi di Napoli "Federico II"****Nome e Cognome****Numero di Matricola:****Spazio riservato alla correzione**

1	2	3	Totale
/10	/10	/10	/30

Per tutti gli esercizi, descrivere la complessità asintotica delle funzioni implementate

1. Si consideri una coda Q con implementazione statica e due Liste, L0 e L1, dinamiche doppiamente puntate, non circolari. La coda è riempita con valori booleani mentre le liste con valori interi. Scrivere una procedura che, facendo uso solo di costrutti ricorsivi, dapprima rimuova i numeri modulo 2 da L0 e modulo 3 da L1, e poi crea una lista L2 ottenuta scorrendo Q utilizzando la seguente regola: con 0 si prende l'elemento da L0, con 1 si prende l'elemento da L1. La coda deve rimanere inalterata. Scrivere un procedura che permetta di costruire Q, L0 e L1 da tastiera. Permetta di stampare Q, L0 e L1 dati in input, di stamparli dopo la rimozione degli elementi ed infine di stampare la lista L2 a fine esercizio.
2. Si consideri un heap riempito con interi (valori da tastiera). Scrivere una procedura per la ricerca ed eliminazione di un valore.
3. Siano G e H due grafi orientati pesati entrambi con pesi positivi, di n vertici 0, 1, ..., n-1 e rappresentati con liste di adiacenza. Scrivere una procedura per il calcolo del grafo differenza T. Il grafo T avrà l'arco (a,b) con peso p se l'arco è presente in G con peso p, ma non in H, oppure è presente in entrambi i grafi ma la differenza dei pesi p è positiva.



Laboratorio di Algoritmi e Strutture Dati I

Docente: Murano

--- MARZO 2020 ---

Laurea in Informatica

Università degli Studi di Napoli "Federico II"

Nome e Cognome

Numero di Matricola:

Spazio riservato alla correzione

1	2	3	Totale
/10	/8	/12	/34

Per tutti gli esercizi, descrivere la complessità asintotica delle funzioni implementate

1. Si consideri una coda Q con implementazione statica e due Liste, L0 e L1, dinamiche doppiamente puntate, non circolari. La coda è riempita con valori booleani mentre le liste con valori interi. Scrivere una procedura che, facendo uso solo di costrutti ricorsivi, dapprima rimuova i numeri modulo 2 da L0 e modulo 3 da L1, e poi operi su L0 e L1 nel seguente modo: con 0 si prende l'elemento in testa a L0 e si sposta in coda a L1, con 1 ciò viceversa, si prende l'elemento in testa a L1 e si sposta in coda a L0. La coda deve rimanere inalterata. Scrivere un procedura che permetta di costruire Q, L0 e L1 da tastiera. Permetta di stampare Q, L0 e L1 dati in input e di stamparli dopo l'esecuzione della regola di cui sopra.
2. Si consideri un ABR. Scrivere una procedura per la creazione e la stampa di un albero "specchio".
3. Siano G e H due grafi orientati pesati entrambi con pesi positivi, di n vertici 0, 1, ..., n-1 e rappresentati con liste di adiacenza. Scrivere una procedura per il calcolo del grafo differenza T. Il grafo T avrà l'arco (a,b) con peso p se l'arco è presente in G con peso p, ma non in H, oppure è presente in entrambi i grafi ma la differenza dei pesi p è positiva pari ed è unica (la seconda volta che appare un arco con peso p già presente, il nuovo arco non si aggiunge).
Scrivere una funzione che permetta di creare il grafo parametrizzato sul numero n, che permetta di aggiungere e togliere archi da G e H, permetta di stampare i grafi in ogni momento, permetta di eseguire la creazione di T e stampare tutti i grafi dopo tale operazione.

ESAME LASD 23/10/2023

Esercizio 1 [6]

Siano **Q1** e **Q2** due code statiche, di elementi positivi, con lo stesso numero **n** di elementi caricati da tastiera. Scrivere una funzione che per ogni coppia di elementi della coda (di pari posizione), controlli che il valore in **Q1** sia pari e strettamente minore di quello in **Q2**, altrimenti si rimuova la coppia dei due valori dalle due code. La funzione deve restituire le code modificate. Esempio: **Q1** : 4, 7, 5 e **Q2** : 5, 6, 9, allora saranno rimossi tutti gli elementi eccetto i numeri 4 in **Q1** e 5 in **Q2**.

Usare (dopo averle implementate) solo funzioni (libreria) di accesso alle code.

Esercizio 2 [8]

Data una lista doppiamente puntata di interi positivi **L**, caricata da tastiera, scrivere una funzione ricorsiva che rimuova tutte le occorrenze di multipli di 2 e di 3.

Esempio:

L1 = 1, 3, 4, 5, 6, 3, 4, 7 → 1, 5, 7

Esercizio 3 [6]

Sia **H** un heap di interi positivi. Scrivere una funzione che presi in input **H** ed un numero **n**, in tempo logaritmico, rimuova **n** da **H**.

Esercizio 4 [14]

Sia **G** un grafo orientato e pesato (con pesi positivi) e di **n** vertici interi **0, 1, n-1**, implementato con liste di adiacenza e caricato da tastiera. Sia **H** un grafo orientato e pesato (con pesi positivi) e di **n** vertici interi **0, 1, n-1**, implementato con matrici di adiacenza.

Scrivere una funzione in **C** che, faccia la differenza (**H - G**) tra i due grafi in un terzo grafo **D** in modo che un arco (**a, b**) sarà presente con peso **p** in **D** se l'arco (**a, b**) è presente in **H** con peso **p1** ed è presente in **G** con peso **p2** e $p = p1 - p2 > 0$. **D** deve essere restituito come matrice di adiacenza.

Traccia LASD gennaio

Prof. Aniello Murano

26 gennaio 2024

Tutti gli esercizi devono prevedere un caricamento da tastiera. Va consegnato l'eseguibile compilato il quale deve mostrare il funzionamento dell'esercizio.

Esercizio 1 [6]

Siano S1 e S2 due stack statici, di elementi positivi, con lo stesso numero n di elementi caricati da tastiera. Scrivere una funzione **ricorsiva** che per ogni coppia di elementi negli stack (stessa profondità), controlli che il valore in S1 sia il doppio di quello in S2, altrimenti si rimuova la coppia dei due valori dai due stack. La funzione deve restituire le code modificate. Esempio: S1: 4,7,6 E S2: 2,6,3 allora sarà rimosso 7 in S1 e 6 in S2. **Usare (dopo averle implementate) solo funzioni (libreria) di accesso agli stack.**

Esercizio 2 [10]

Date due liste **ordinate crescenti** L1 e L2 doppiamente puntate non circolari di interi positivi L caricate da tastiera, scrivere una funzione **ricorsiva** che prima rimuova tutte le occorrenze di **multipli di 2 da entrambe le liste** e poi ne faccia lo shuffle, “inserimento a pettine” mantenendo l'ordine crescente.

Esercizio 3 [16]

Sia G1 un grafo orientato e pesato (con pesi positivi) e di n vertici interi 0, 1, n-1, implementato con liste di adiacenza e caricato da tastiera. Sia G2 un grafo orientato e pesato (con pesi positivi) e di n vertici interi 0, 1, n-1, implementato con matrice di adiacenza.

Scrivere

1. una funzione in C che, faccia la differenza (G1-G2) tra i due grafi e restituisca un grafo DM, rappresentato con matrice di adiacenza in modo che un arco (a,b) sarà presente con peso p in DM se l'arco (a,b) è presente in G1 e, se presente o meno in G2, la differenza dei pesi è p ed è positiva.
2. una funzione in C che, come nel caso precedente faccia la differenza (G2-G1) tra i due grafi e restituisca un grafo DL, rappresentato con lista di adiacenza in modo che un arco (a,b) sarà presente con peso p in DL se l'arco (a,b) è presente in G2 e, se presente o meno in G1, la differenza dei pesi è p ed è positiva.