Fondamenti di Programmazione – Prova di laboratorio
02-09-09 – AD – Andrea Sterbini

Svolgete due esercizi a scelta fra quelli della fila segnata:

A **B**

Chiamate i file **Nome.Cognome.1A.c** eccetera

Esercizio 1 A

La trasposta di una matrice **C** di **m** righe e **n** colonne è una matrice **CT** di **n** righe e **m** colonne. La generica cella **i**, **j** della matrice **CT** contiene il valore **C[j,i]** della matrice **C** di cui è trasposta.
Il risultato della somma fra due matrici, **A** e **B** di **m** righe ed **n** colonne, è una nuova matrice **C** di **m** righe e **n** colonne.
Il valore della generica cella **i, j** della matrice **C** è dato dalla somma dei valori **A[i,j]** e **B[i.j]**.
Si implementi una funzione **somma** che riceve 3 matrici di **long** (**A**, **B** e **C**) e tutti gli altri argomenti che ritenete necessari e memorizza nella matrice **C** il risultato della somma fra la matrice **A** e la matrice **B**.
Implementare quindi una funzione **trasposta** che riceva una matrice **C** e una matrice **CT**, e tutti gli altri argomenti che ritenete necessari, e calcoli la trasposta di **C** in **CT**.
Scrivere quindi un programma che legga due interi **m** ed **n**, allochi le matrici **A** e **B**, di **m** righe e **n** colonne, inizializzandone le celle con valori **random** compresi fra 0 e 100 . Il programma deve quindi stampare la matrice trasposta della somma fra **A** e **B**.
Per ottenere valori random, usate la funzione **rand()** della libreria **stdlib.h**.
Si ricorda che la funzione rand ha prototipo **int rand (void);**
per avere valori fra 0 e 100, potete usare l'operazione di modulo **%**

Esercizio 1 B

Il risultato del prodotto fra due matrici, **A** e **B**, rispettivamente di **m** righe ed **n** colonne e **n** righe e **l** colonne, è una nuova matrice **C** di **m** righe e **l** colonne.

Il valore della generica cella **i, j** della matrice **C** è dato dalla seguente formula: $\sum_{r=1}^{n} A[i,r] * B[r,j]$.

Si implementi una funzione **prodotto** che riceve 3 matrici di **long** (**A**, **B** e **C**) e tutti gli altri argomenti che ritenete necessari e memorizza nella matrice **C** il risultato del prodotto fra la matrice **A** e la matrice **B**.
Scrivere quindi un programma che legga tre interi **m**, **n** ed **l**, allochi le matrici **A** e **B (**rispettivamente di **m** righe e **n** colonne e **n** righe e **l** colonne**)** inizializzandone le celle con valori **random,** compresi fra 0 e 100. Il programma deve quindi stampare la matrice prodotto fra **A** e **B**.
Per ottenere valori random, potete usare la funzione **rand()** della libreria **stdlib.h**.
Si ricorda che la funzione rand ha prototipo **int rand (void);**
per avere valori fra 0 e 100, potete usare l'operazione di modulo **%**

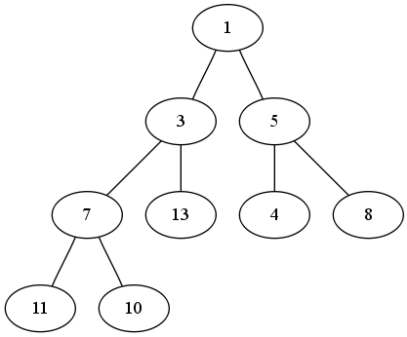
Esercizio 2 A

Si implementi la funzione di inserimento di un valore in una **coda**.
Scrivere quindi un programma che legga una successione di numeri interi, terminata dal valore -1, e li inserisca in una **coda**.
Terminati gli inserimenti, il progamma deve stampare i valori presenti nella coda in **ordine non-decrescente**.
Prestate particolare attenzione all'allocazione e deallocazione della memoria.

Esercizio 2 B

Si implementino le funzioni di gestione di una **lista ordinata** (inserimento, ricerca, cancellazione). La lista deve avere ordinamento **crescente**.
Scrivere quindi un programma che legga una successione di numeri interi, terminata dal valore -1, e li inserisca in una lista ordinata. Qualora un valore della successione sia già presente all'interno della lista al momento della lettura, tale valore deve essere eliminato dalla lista anzi che inserito.
Terminati gli inserimenti, il programma deve stampare il contenuto della lista.
Prestate particolare attenzione all'allocazione e deallocazione della memoria.

Esercizio 3 A



È possibile utilizzare un vettore per rappresentare un albero binario completo come segue:

Il figlio sinistro di un nodo con indice **i** è il nodo con indice **2i+1**; il figlio destro è il nodo con indice **2i+2**.

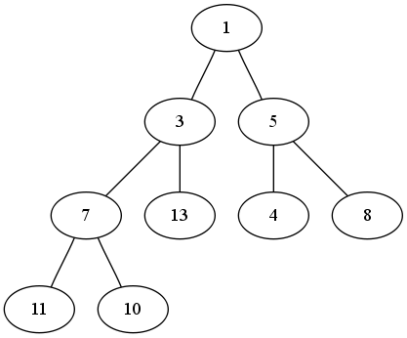
Esempio: il vettore contenente i numeri 1 3 5 7 13 4 8 11 10
corrisponde all'albero a fianco

Scrivere un programma che legge in input un intero n e quindi n interi, li memorizza in un vettore (che potete dichiarare staticamente di 100 elementi, o -meglio- allocare dinamicamente dopo la lettura del valore n), e quindi stampa i valori con una visita **in-ordine** dell'albero da sinistra verso destra, dando quindi precedenza ai figli sinistri sui figli destri.

Quindi in questo caso l'output dell'esempio sarebbe: 11 7 10 3 13 1 4 5 8

Non è necessario implementare gli alberi, usate direttamente il vettore.

Esercizio 3 B



È possibile utilizzare un vettore per rappresentare un albero binario completo come segue:

Il figlio sinistro di un nodo con indice **i** è il nodo con indice **2i+1**; il figlio destro è il nodo con indice **2i+2**.

Esempio: il vettore contenente i numeri 1 3 5 7 13 4 8 11 10
corrisponde all'albero a fianco

Scrivere un programma che legge in input un intero n e quindi n interi, li memorizza in un vettore (che potete dichiarare staticamente di 100 elementi, o -meglio- allocare dinamicamente dopo la lettura del valore n), e quindi stampa i valori con una visita **in-ordine** dell'albero **da destra verso sinistra**, dando quindi precedenza ai figli destri sui figli sinistri.

Quindi in questo caso l'output dell'esempio sarebbe: 8 5 4 1 13 3 10 7 11

Non è necessario implementare gli alberi, usate direttamente il vettore.