Esercitazione 11

Nella propria home directory creare una sottodirectory chiamata es11, in cui metteremo tutti i file C di oggi.

Quando dovete usare o ritornare dei valori booleani, usate la seguente definizione:
`typedef enum {false, true} boolean;`
Inoltre quando un esercizio vi chiede di scrivere un "predicato", dovete scrivere una funzione che ritorni un valore di tipo boolean.

Ricordate che oltre alla funzione o procedura indicata dall'esercizio, si richiede che scriviate anche un main che usi tale funzione e ne dimostri *ogni* funzionalità.

Esercizi introduttivi:

1) Definire una procedura **elementiDi** che data una ListaDiInteri, restituisca una nuova ListaDiInteri ordinata e senza ripetizioni contenente tutti e soli gli elementi che compaiono nella lista data. La lista originale deve restare immutata.

2) Definire una procedura **inserisciDopoUltimoMaggiore** che data una ListaDiInteri ed un intero P, inserisca P dopo l'ultimo elemento di lista il cui campo info sia maggiore di P.
Se lista non contiene alcun elemento maggiore di P (o se la lista è vuota), la procedura lo inserisce in testa.

3) Scrivere una funzione che presi in input due liste di interi, crea una nuova lista dello stesso tipo contenente esattamente i valori interi comuni alle due liste senza ripetizioni e ritorna la lista creata.
Ad esempio se le due liste di input sono:
 2 -> 3 -> 5 -> 3 -> 5
 7 -> 2 -> 3 -> 5 -> 5 -> 7
Allora la funzione deve restituire la lista: 2 -> 3 -> 5 (l'ordine degli elementi non è importante).

4)Si vuole modellare un gioco con le carte mediante una lista concatenata. Una carta e' rappresentata dal suo valore (un intero da 1 a 10) e dal suo seme ((Q)Quadri,(C)Cuori,(P) Picche e (F)Fiori). Ogni nodo della lista deve rappresentare una carta con in aggiunta l'informazione di quante carte dello stesso seme seguono nella lista.
   a) Definire i tipi opportuni per rappresentare il gioco.
   b) Scrivere una funzione ricorsiva che data una lista di carte, un valore e il seme di una carta, controlli che tale carta appartenga alla lista sfruttando al meglio tutte le informazioni contenute nella lista.

c) Scrivere una procedura che data una lista di carte, un valore e il seme di una nuova carta, la inserisca prima della prima carta con lo stesso seme, se esiste, altrimenti la inserisca in coda.

5) Sia data una lista contenente almeno due elementi. Si scriva una funzione che ricevuta in ingresso la lista modifichi la stessa, memorizzando nell'ultimo nodo il prodotto del penultimo ed ultimo nodo, nel penultimo il prodotto del terzultimo e del penultimo e così via. Il primo record non deve essere modificato.
Ad esempio, una lista contenente la sequenza di interi 4 6 2 3 9 verrà modificata dalla funzione nella lista 4 24 12 6 27.

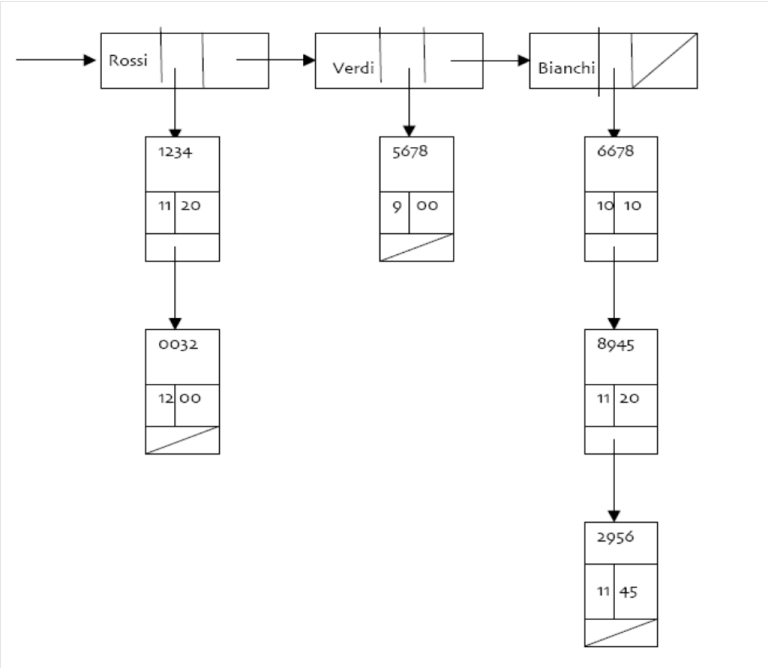
Esercizi:

6) Il grafico di una funzione definita sugli interi e a valori reali è rappresentato da una lista ordinata di punti, cioè di record aventi tre campi: un intero ***x*** (ascissa), un intero ***y*** (ordinata), e un campo ***next*** che punta al prossimo punto.
Nel grafico i punti sono ordinati per ascissa crescente, e non ci possono essere due punti aventi la stessa ascissa. Se *fun* è la funzione rappresentata dal grafico *gr*, la presenza del punto (x, y, p) in *gr* indica che ***fun(x) = y***.

a) Si definiscano i tipi di dati necessari per implementare in C la rappresentazione indicata. Si chiami *Grafico* il tipo di dati principale. Si definiscano quindi le seguenti procedure che operano su oggetti di tipo Grafico, rispettando il prototipo proposto:
b) ***void update(Grafico * fun, int argx, double argy);***
   Aggiorna il grafico passato per argomento aggiungendo un punto con ascissa **argx** e ordinata **argy**. Se fun aveva già un punto con ascissa argx, ne viene solo modificato il campo y con il nuovo valore. Altrimenti si inserisce un nuovo punto nel grafico, preservando l'ordinamento.
c) double ***simpleEval(Grafico fun, int arg);***
   Restituisce il valore della funzione rappresentata da fun per l'argomento intero arg. Restituisce 0 se non esiste in fun un punto con ascissa arg. Questa funzione DEVE essere definita in modo ricorsivo.
d) double length(Grafico fun);
   Restituisce la lunghezza del grafico, visto come una linea spezzata passante per i punti indicati.
   NB: Per il calcolo della radice quadrata si può usare la funzione con prototipo ***double sqrt(double)*** della libreria *math.h*.

7) Si vogliono rappresentare gli appuntamenti giornalieri di uno studio medico che ospita ogni giorno diversi medici.

Ad esempio



rappresenta gli appuntamenti relativi al 7 gennaio dei dottori che visitano in tale giorno. Gli appuntamenti sono memorizzati in liste ordinate per orario di appuntamento (l'orario può essere rappresentato con due interi ore e minuti) e i nomi dei pazienti sono identificati da un codice numerico per rispettare la privacy.

a) Si definiscano i tipi di dato necessari per implementare in C la rappresentazione indicata. Si identifichi con StudioMedico il tipo di dato principale.

b) Si definiscano le seguenti operazioni su oggetti di tipo StudioMedico mediante opportune procedure o funzioni.

1. Dato un medico, calcolare il numero di pazienti che deve visitare;
2. Aggiungere un nuovo medico con lista vuota di appuntamenti;
3. Dato un medico, inserire un nuovo appuntamento;
4. Dato un medico, controllare che i suoi appuntamenti distino minimo 20 minuti.