Esercitazione stdio.h, stringhe, strutture

1) Utilizzare la libreria stdio.h per creare un file misure che contenga 150 numeri interi nell'intervallo (0,20] generati casualmente con la funzione rand(). I numeri devono essere scritti uno per linea separati da '\n'.

2)Utilizzare il file misure generato nell'esercizio precedente e calcolare il numero di valori generati che ricadono nei 10 intervalli
(0, 2] (2,4] ..... (18,20]
stampare il numero dei valori per ciascun intervallo sullo standard output.

3)Ripetere i due esercizi precedenti considerando numeri reali.

I numeri reali possono essere generati usando la seguente formula
((double) rand())+ 1.0) / ((((double) RAND_MAX) + 1.0) / 20)
Avete altre idee? Funzionano? Se no, perche'?

4) Scrivete una funzione C nuovo_mazzo() che crea un mazzo (mischiato!) di 40 carte utilizzando i seguenti tipi:

```
/** i valori delle carte */
typedef enum valori{ASSO,DUE,TRE,QUATTRO,CINQUE,SEI,SETTE,FANTE,DONNA,RE}
valori_t;
/** i semi delle carte */
typedef enum semi{CUORI,QUADRI,FIORI,PICCHE } semi_t;
/** una carta e' data da un valore ed un seme */
typedef struct carta {
  /** valore */
  valori_t val;
  /** seme */
  semi_t seme;
} carta_t;
```

ed una funzione stampa_mazzo() che stampa le carte del mazzo sullo standard output in modo gradevole. Definite i parametri ed i valori restituiti dale funzioni opportunamente.

5) Utilizzando le funzioni dell'esercizio precedente realizzate un programma C che gioca a briscola con un utente. Il programma crea il mazzo di carte, stampa sullo standard output il nome della briscola e le carte in mano all'utente ed attende sullo standard input la giocata. Il programma puo' giocare con una strategia semplice a piacere (ad esempio la prima carta della mano). Ad esempio:

```
$./briscola
Nuova partita, briscola CUORI
Mano #1: Hai in mano
4Fiori 5Picche QCuori
Cosa giochi ?
se digitiamo
4Fiori
io gioco 2Cuori, preso
Mano #2: Hai in mano
KFiori 5Picche QCuori
io gioco 7Picche
Cosa giochi ?
```

Il gioco continua fino all'esaurimento delle carte. Alla fine il programma stampa il vincitore ed i punti totalizzati. Ad esempio:
Hai vinto con 87 punti!

6) Definire un nuovo tipo di dato capace di rappresentare una data.
Scrivere poi delle opportune funzioni/procedure che:
- ricevuta una data la aggiorni al giorno successivo (ignorando gli anni bisestili);
ricevute due date verifichino che la prima preceda la seconda;
ricevute due date, ritornino la differenza in anni fra la prima e la seconda.

7)Definire un nuovo tipo di dato capace di rappresentare i dipendenti di una ditta: di tali dipendenti interessa il codice numerico identificativo unico, la data di assunzione, il grado raggiunto (un intero) e lo stipendio.
Usare il tipo di dato definito nell'esercizio precedente per tutte le date.
Scrivere le seguenti funzioni/procedure:
•calcolaAnzianita che, dato un dipendente, calcoli il numero di anni trascorsi dalla data di assunzione al 2102;
•aggiornaStipendio che, dato un dipendente, aumenti del 1% il suo stipendio per ogni anno di anzianita' accumulato (attenzione l'interesse da calcolare e' un interesse composto).
•superiore, che dati due dipendenti ritorni 1,0 o -1 se il primo e' rispettivamente un superiore, un pari grado o un sottoposto
del secondo. Notare che a parita' di grado, il superiore e' il dipendente con l'anzianita' maggiore. Se e' pari anche l'anzianita', i dipendenti sono pari grado.
Scrivere quindi un programma che richieda all'utente di inserire i dati di due impiegati, utilizzi le due procedure di cui sopra e ne stampi l'esito.

8)Scrivere un programma C che legge una sequenza di studenti dal file anagrafe_studenti. Ogni studente e' memorizzato su file in una singola linea contenente tre stringhe di caratteri separate da ':' e terminata da '\n' secondo il formato
cognome:nome:numero_di_matricola
quindi ad esempio
...
Rossi:Mario:234445
Bixio:Nino:435678
Garibaldi:Giuseppe:787899
...
Il programma legge da file gli studenti e memorizza i dati relativi a ciascun studente in un array di strutture di tipo:

```
#define N 50

typedef struct {
  char nome[N+1];
  char cognome[N+1];
  unsigned matricola;
  } studente_t;
```

L'array viene poi ordinato per il campo cognome e nel caso di cognomi uguali per il campo nome e poi stampato sullo standard output.
Suggerimento: Per la lettura da file usare fscanf() con una opportuna stringa di formattazione oppure fgets() per leggere fino al primo \n e strchr() per localizzare i caratteri separatori :