	Politecnico di Milano Facoltà di Ingegneria Industriale **INFORMATICA B** Appello 14 settembre 2016	COGNOME E NOME
		MATRICOLA
		Spazio riservato ai docenti

- Il presente plico contiene 3 esercizi e **deve essere debitamente compilato con cognome e nome, numero di matricola**.
- Il tempo a disposizione è di 1 ora e 40 minuti.
- Non separate questi fogli. Scrivete la soluzione solo sui fogli distribuiti, utilizzando il retro delle pagine in caso di necessità. Cancellate le parti di brutta (o ripudiate) con un tratto di penna.
- Ogni parte non cancellata a penna sarà considerata parte integrante della soluzione.
- È possibile scrivere a matita (e non occorre ricalcare al momento della consegna!).
- **È vietato utilizzare calcolatrici, telefoni o pc**. Chi tenti di farlo vedrà annullata la sua prova.
- È ammessa la consultazione di libri e appunti, purché con pacata discrezione e senza disturbare.
- **Qualsiasi tentativo di comunicare con altri studenti comporta l'espulsione dall'aula**.
- È possibile ritirarsi senza penalità.
- **Non è possibile lasciare l'aula conservando il tema della prova in corso**.
- L'esame orale è parte integrante dell'esame e deve essere realizzato almeno sufficientemente per il superamento dell'esame complessivo.
- Per accedere all'esame orale è necessario dimostrare sufficienti competenze di C e Matlab nello scritto.

Esercizio 1 (10 punti)

Si consideri un programma scritto in linguaggio C per la gestione del magazzino di una farmacia. Questo programma mantiene i dati sui farmaci disponibili in una variabile `db`. La variabile `db` è un array di elementi di tipo `Farmaco.` Il numero effettivo di valori contenuti in tale array è memorizzato nella variabile `numeroFarmaci`.

Il tipo `Farmaco` definisce i campi per il nome del farmaco (`nomeFarmaco`), il numero di scatole disponibili (`nscatole`) e le caratteristiche di ciascuna scatola di farmaco (il campo `scatole` rappresenta un insieme di scatole di farmaco).

Le informazioni relative a ciascuna scatola sono rappresentate con il tipo `Scatola`. Questo tipo mantiene informazioni su: quantità (`quantita`) di farmaco presente nella scatola (uno stesso farmaco può essere disponibile in scatole che contengono quantità diverse di medicinale, per esempio, farmaco Marmellad disponibile in scatole da 15 o 30 pillole) e il mese di scadenza della scatola, definito utilizzando il seguente tipo enumerativo:

```
typedef enum {gen, feb, mar, apr, mag, giu, lug, ago, set, ott, nov, dic}Mese;
```

Domanda A

Data la descrizione al punto precedente specificare in linguaggio C i tipi `Scatola` e `Farmaco` e dichiarare le variabili `db` e `numeroFarmaci`. Ipotizzare che il numero massimo di farmaci sia 1000 e che il numero massimo di scatole per ciascun farmaco sia 100.

Soluzione

```
#define N 100
#define M 1000

typedef char stringa[30];

typedef enum {gen, feb, mar, apr, mag, giu, lug, ago, set, ott, nov, dic} Mese;

typedef struct {
    int quantita;
    Mese scadenza;
} Scatola;

typedef struct {
    stringa nomeFarmaco;
    int nscatole;
    Scatola scatole[N];
} Farmaco;

int numeroFarmaci;
Farmaco db[M];
```

Domanda B

Si scriva una porzione di codice in linguaggio C che, ipotizzando di avere il nome di un farmaco nella variabile di tipo stringa `nomeF` e un mese specificato in una variabile `mese` di tipo `Mese`, stampi a video la somma totale delle quantità del farmaco la cui scadenza è posteriore al valore della variabile `mese`.

Si supponga che tutte le variabili necessarie siano state opportunamente inizializzate. In particolare che la variabile `db` contenga un numero di dati validi di tipo `Farmaco` pari al valore contenuto in

numeroFarmaci, e che nomeF e mese siano state inizializzate con il nome del farmaco di cui verificare la disponibilità e il mese di scadenza.

Esempio:

Se nomeF = "Marmellad" e mese = lug e se supponiamo di avere fra i farmaci nella variabile db il farmaco:

```
nomeFarmaco: "Marmellad"
nscatole: 3
scatole[0]: { quantita: 20, scadenza: apr }
scatole[1]: { quantita: 15, scadenza: ott }

scatole[2]: { quantita: 30, scadenza: ago }
```

il programma dovrà stampare:

```
Quantità di Marmellad disponibile dopo il mese 7: 45
```

Poiché solo una scatola da 30 e una da 15 unità non saranno ancora scadute dopo luglio.

Soluzione:

```
int quantita_tot = 0;

Farmaco f;
for(int i = 0; i < numeroFarmaci; i++)
    if(strcmp(db[i].nomeFarmaco, nomeF) == 0) {
        f = db[i];
        for(int j = 0; j < f.nscatole; j++)
                if(f.scatole[j].scadenza > mese)
                    quantita_tot += f.scatole[j].quantita;
    }
printf("Quantità di %s disponibile dopo il mese %d: %d\n", nomeF, mese, quantit
a_tot);
```

Esercizio 2 (10 punti)

Data la seguente funzione ricorsiva Matlab:

```
1.  function [c, d] = funzione (x, y)
2.      if (x < y)
3.          d = 0;
4.          c = x;
5.      else
6.          [a, b] = funzione (x - y, y);
7.          d = b + 1;
8.          c = a;
9.      end
10. end
```

1) Simulare il funzionamento della funzione, evidenziando in modo opportuno i valori dei parametri e dei valori di ritorno di ogni chiamata ricorsiva, per le seguenti coppie (x, y) di valori in ingresso:

i] (3, 6) ii] (6, 6) iii] (4, 2) iv] (5, 2)

2) Spiegare cosa calcola la funzione, motivando la risposta.

3) Scrivere in linguaggio Matlab una funzione non ricorsiva che svolga lo stesso calcolo.

4) Rispondere alla seguente domanda articolando e argomentando adeguatamente la risposta:
E' sempre vero che la ricorsione è vantaggiosa in termini di tempo e memoria rispetto all'iterazione?

Soluzione

1)

(3, 6)
- 1ª chiamata: x = 3
  - y = 6
  - x < y → d = 0, c = 3
- fine

(6, 6)
- 1ª chiamata x = 6
  - y = 6
  - x < y è falso → chiamata ricorsiva [a, b] = funzione(0, 6);
  - 2ª chiamata x = 0
    - y = 6
    - x < y → d = 0, c = 0
  - fine
  - a = 0
  - b = 0
  - d = 0 + 1 = 1
  - c = 0
- fine

(4, 2)
- 1ª chiamata x = 4
  - y = 2
  - x < y è falso → chiamata ricorsiva [a, b] = funzione(2, 2);
  - 2ª chiamata x = 2
    - y = 2

x < y è falso → chiamata ricorsiva [a, b] = funzione(0, 2);
3ª chiamata x = 0
y = 2
x < y → d = 0, c = 0
fine
a = 0
b = 0
d = 0 + 1 = 1
c = 0
fine
a = 0
b = 1
d = 1 + 1 = 2
c = 0
fine

(5, 2)
1ª chiamata x = 5
y = 2
x < y è falso → chiamata ricorsiva [a, b] = funzione(3, 2);
2ª chiamata x = 3
y = 2
x > y → chiamata ricorsiva [a, b] = funzione(1, 2);
3ª chiamata x = 1
y = 2
x < y → d = 0, c = 1
fine
a = 1
b = 0
d = 0 + 1 = 1
c = 1
fine
a = 1
b = 1
d = 1 + 1 = 2
c = 1
fine

2) La funziona calcola e restituisce in uscita i valori c e d, rispettivamente corrispondenti a resto e quoziente della divisione tra i valori di x e y dati in ingresso. Il quoziente è calcolato ricorsivamente sottraendo a ogni chiamata alla funzione il divisore dal dividendo, fino al caso base x < y, che indica che x non è ulteriormente divisibile per y.

3)

```
1.  function [c, d] = funzione_non_ricorsiva (x, y)
2.      d = 0;
3.      while (x >= y)
4.          x = x - y;
5.          d = d + 1;
6.      end
7.      c = x;
8.  end
```

4) No, non è sempre vero che la ricorsione è vantaggiosa in termini di tempo e memoria rispetto all'iterazione; spesso le soluzioni ricorsive sono molto eleganti e più vicine alla definizione del problema, ma possono essere inefficienti, perché chiamare un sottoprogramma equivale ad allocare memoria a run-time.

Esercizio 3 (6 punti)

- **A.** Si determini la codifica in virgola mobile del valore decimale 23.3 secondo lo Standard IEEE 754-1985 a precisione singola, riportando i calcoli effettuati.
- **B.** La codifica in virgola mobile del valore decimale 23.3, calcolata al punto A, è esatta? Giustificare la risposta.

Soluzione

A.

Parte intera: 23 → 10111
Parte frazionaria: 0.3
0.3 * 2 = 0.6
0.6 * 2 = 1.2
0.2 * 2 = 0.4
0.4 * 2 = 0.8
0.8 * 2 = 1.6
0.6 * 2 = 1.2 … → 0.01001100110011001….. → 0.0(1001)

23.3 → $1.01110(1001) \times 2^4$

E: 4 + 127 = 131 → 11000001

131 : 2 = 65 (1)
65 : 2 = 32 (1)
32 : 2 = 16 (0)
16 : 2 = 8 (0)
8 : 2 = 4 (0)
4 : 2 = 2 (0)
2 : 2 = 1 (0)
1 : 2 = 0 (1)

Codifica finale
S:0
E:10000011
M: 01110100110011001100110

B.

La codifica in virgola mobile del valore decimale 23.3 non è esatta dato che la sua mantissa è periodica.