INFORMATICA - CdL in FISICA
PROVA di PROVA SCRITTA – 10/06/2011

N.B.: In tutti gli esercizi viene valutata anche la leggibilità del codice proposto. Inoltre, non è consentito l'uso di istruzioni che alterino il normale flusso dell'esecuzione (come, ad esempio, `continue`, `break` e istruzioni di `return` all'interno di cicli che ne provochino l'uscita forzata).
Utilizzare il tipo `boolean` definito da `typedef enum {false, true} boolean` invece di interi come variabili booleane;
In tutti gli esercizi non è consentito l'uso di variabili `static`.

ESERCIZIO 1
Una sequenza di caratteri è chiamata **legale** se ogni lettera minuscola è seguita dal carattere '$' e ogni occorrenza del carattere '$' è preceduta da una lettera minuscola. Scrivere un programma che legge una sequenza di caratteri terminata da un carattere non alfabetico e diverso dal carattere speciale '$', e indichi con un messaggio se la sequenza è legale oppure no.

Per esempio, per i seguenti input il programma deve stampare `La sequenza e' legale`:

";" "aAoBCp$L-"

mentre per i seguenti deve stampare `La sequenza non e' legale`:

"a9" "AA$." "ABa$$."

ESERCIZIO 2
Scrivere una funzione *ricorsiva* che, dati i soli parametri `vet` (un vettore di interi), *dim* e n (un intero), controlli che il numero degli interi positivi in `vet` sia maggiore di n più il numero degli elementi negativi.
Se ad esempio `vet` e'

1	-8	3	7	-9	-7	0	2	2

e $n = 4$ la funzione deve restituire *false* mentre se $n = 3$ deve restituire *true*.

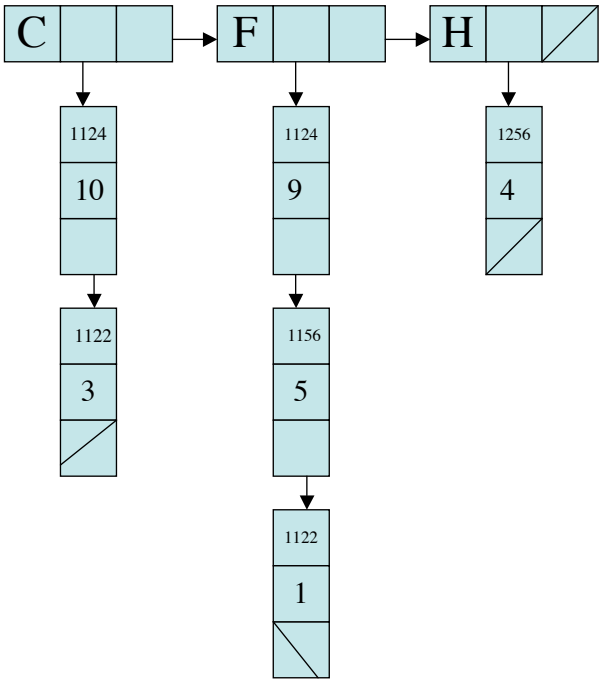
ESERCIZIO 3
Dato il seguente programma:

```
#include <stdio.h>
int p(int *x, int *y)
{
int ris = *y;
if (*x > 0)
  {
     *x = *x-1;
     *y = *y+2;
     ris = p(x,y);
  }
return ris;
}

main()
{
int a=A, b=B;
printf("a = %d, b = %d, risultato = %d", a, b, p(&a, &b));
}
```

dove `A` e `B` indicano generiche costanti intere, indicare l'output del programma in funzione dei valori di `A` e `B`.

ESERCIZIO 4
Si consideri una ditta che fornisce pasti agli uffici. Si vuole rappresentare l'informazione relativa alla prenotazione giornaliera dei menu' diversi che la ditta mette a disposizione. Vogliamo quindi avere una lista *ordinata*, dove ogni elemento (identificato da una lettera) rappresenta un menu che e' stato prenotato, con l'informazione relativa all'ufficio che l'ha prenotato (un codice numerico) e il numero di pasti di tale menu' prenotati. Di seguito riportiamo un esempio.



(i) (**2 punti)** Si definiscano i tipi di dato necessari per implementare in `C` la rappresentazione indicata. Si identifichi con `Catering` il tipo di dato principale.

Si definiscano le seguenti operazioni su oggetti di tipo `Catering` mediante opportune procedure o funzioni.

(ii) (**3 punti**) Dato un menu', contare il numero ordinato di pasti prenotati per tale menu';

(iv) (**3 punti**) Dato un menu' e un cliente, restituire il puntatore all'ordinazione del cliente;

(v) (**6 punti**) Dato un menu', un cliente e il numero di pasti richiesto, inserire la nuova ordinazione.
N.B. se nella lista il menu' non e' presente si deve inserire anche il menu'.

N.B. Le procedure/funzioni **NON** devono contenere alcuna istruzione di input/output (ad es. `scanf, printf, getchar, putchar,...`)