	Politecnico di Milano Facoltà di Ingegneria Industriale **INFORMATICA B** Appello del 1 settembre 2010		COGNOME E NOME
	RIGA	COLONNA	MATRICOLA
			Spazio riservato ai docenti

- Il presente plico contiene 4 esercizi, deve essere debitamente compilato con cognome e nome, numero di matricola, posizione durante lo scritto (comunicata dal docente).
- Il tempo a disposizione è di 1 ora e 45 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.

Esercizio 1 (11 punti)

Definire un tipo di dato studentiCorso come array di struct. Il tipo deve rappresentare i dati degli studenti iscritti ad un corso e le votazioni sostenute nei diversi appelli d'esame di un certo anno accademico.

Il tipo deve permettere di memorizzare dati per un numero di studenti non noto a priori, ma sicuramente inferiore a 300. Per ogni singolo studente è necessario memorizzare la matricola, il nome, il cognome. È inoltre necessario memorizzare i dati delle prove d'esame. Si assume che gli appelli disponibili per il corso siano 4. Per ogni appello, è necessario memorizzare se lo studente ha sostenuto l'esame (presente/non presente) e l'eventuale valutazione conseguita (un intero tra 0 e 33).

Scrivere poi il frammento di codice che, assumendo l'esistenza della variabile StudentiCorso2008, di tipo studentiCorso, acquisisca da tastiera la matricola di uno studente e, per ogni appello d'esame, stampi a video il voto conseguito nel caso di presenza all'appello, oppure il messaggio "non presente". Di seguito, si riporta un esempio di output per un ipotetico studente con matricola 12345:

Valutazioni per la matricola 12345
Appello 1: 16
Appello 2: non presente
Appello 3: 24
Appello 4: non presente

Soluzione

```
typedef struct {
  char nomeEsame[100] ; /* non necessario */
  int presente; /* 1 se presente, 0 altrimenti */
  int voto;
} Appello;

typedef struct {
  char matricola[6] ;
  char nome[100] ;
  char cognome[100] ;
  Appello appelli[4]
} Studente;

typedef StudentiCorso Studente[300];

StudentiCorso StudentiCorso2008;

int i,j;

for(i=0;i<300 ;i++){
        if(matricola>0) /* test per verificare se la casella dell'array è valida */
        {
                printf("\nValutazioni per la matricola %s\n ", StudentiCorso2008[i].matricola);
                for(j=0;j<4;j++){
                        printf("Appello %d: ", StudentiCorso2008[i].matricola);
                        if(StudentiCorso2008[i].appelli[j].presente==1)
                                printf("%d\n", StudentiCorso2008[i].appelli[j].voto);
                        else
                                printf("non presente");
                }
        }
}
```

Esercizio 2 (12 punti)

Si implementi una funzione che, preso come parametro un array A, calcola il massimo comun divisore (MCD) fra tutti gli elementi di A. Esempio: se la funzione è chiamata sull'array [9 18 6 27 30 42] deve restituire 3

Si ricorda che, dati tre numeri a, b, c, l'MCD tra a, b e c è uguale all'MCD tra c e l'MCD tra a e b.

Soluzione

```
function [res] = MCD (x,y)
   res = min(x,y);
   flag = 0;
   while res > 1 && flag ==0
      if mod(x,res) == 0 && mod(y,res) ==0
         flag =1;
      else
         res = res -1;
      end
   end
end

function [res] = esercizio(A)
   n = length(A);
   res = A(1);
   for i=2:n
      res = MCD(res,A(i));
   end
end
```

Esercizio 3 (11 punti)

È noto, dalla definizione del coefficiente binomiale $\binom{n}{k}$, che, se $n>0$ e $0<k<n$, vale la seguente relazione ricorsiva

$$\binom{n}{k}=\binom{n-1}{k-1}+\binom{n-1}{k}$$

dove la base della ricorsione è data da $\binom{n}{k}=1$ se $n=0$ o $k=0$ o $k=n$.

Utilizzare tale definizione ricorsiva per scrivere una funzione Matlab ricorsiva che calcoli il valore del coefficiente binomiale $\binom{n}{k}$ a partire dai due parametri *n* e *k*.

Simulare l'esecuzione della funzione che calcola il coefficiente $\binom{3}{2}$ mostrando la sequenza delle chiamate ricorsive che hanno luogo durante il calcolo.

Soluzione

```
function [c]=coefBinRic(n, k)
if n==0 || k==0 || k==n
    c=1;
else c=coefBinRic(n-1,k-1)+coefBinRic(n-1,k);
end
```

La chiamata `coefBinRic(3,2)` dà origine a `coefBinRic(2,1)+coefBinRic(2,2)`; la prima di queste dà origine a `coefBinRic(1,0)+coefBinRic(1,1)`, entrambe corrispondenti al caso base della ricorsione. Viene quindi restituito 2, che sommato al valore di `coefBinRic(2,2)`, pure corrispondente al caso base e quindi pari a 1, dà correttamente il valore 3.