Classe 4 Bi
2-12-2002

Sia:

```
#define DIM 1000

        struct alunno {
                char    cognome[64],
                        nome[64];
                        classe[8];
        }
```

Siano:

```
        struct alunno   elenco[DIM];        /* elenco degli alunni del volta                    */

        int pagella[DIM][16];               /* elenco dei voti della pagella finale di ogni alunno
                                               Ogni linea contiene i voti finali di un alunno   */
```

L'alunno che occupa la posizione i-esima di elenco ha i voti riportati nel vettore di posizione i-esima di pagella.

Dopo aver discusso della struttura dati utilizzata, risolvere i seguenti problemi:

1. Stampare le pagelle relative agli alunni di una determinata classe.
2. Restituire l'elenco degli alunni con media dei voti sufficiente.
3. Cancellazione di un alunno con relativa pagella.
4. Ordinamento degli alunni in base alla media dei voti.

Produrre per ogni esercizio una breve descrizione del problema, un breve algoritmo e la codifica in C. L'esercizio verrà valutato solo se completo in tutte le parti.
Ogni esercizio vale punti 2.

Soluzione.

Suppongo che:

- Il vettore elenco è terminato da una struct che contiene come cognome una stringa di lunghezza zero, cioè una stringa che ha come primo elemento '\0';
- Ogni linea del vettore pagella, che contiene per ogni alunno i voti della pagella, è terminato da uno zero

Le funzioni in c che risolvono il problema sono le seguenti:

```
/*
      compito.h
*/

#define DIM 1000
#include <string.h>

struct alunno {
      char  cognome[64],
            nome[64],
            classe[8];
};


/*
 * Stampare le pagelle relative agli alunni di una determinata classe.
 */
void print(struct alunno x, int p[]) {
      int i;

      printf("%64s\n%64s\n","------------ cognome -----------",
                              "-------------- nome ------------");
      printf("%64s\n%64s\n",x.cognome, x.nome);
      for(i=0;p[i]!=0;++i) {
            if(i % 8 == 0) printf("\n");
            printf("%3d", p[i]);
      }
}


void stampa (struct alunno e[], int p[][16], char* cl ) {
      int i;

      for(i=0; strlen(e[i].cognome) > 0; ++i)
            if(strcmp(e[i].classe, cl) == 0) print(e[i], p[i]);
}


/*
 * Restituire l'elenco  degli alunni con media dei voti sufficiente.
 */

float media(int p[]) {
      int i;
      float somma=0;

      for(i=0;p[i]!=0;++i) somma+=p[i];
      return somma/i;
}

int *elenco (struct alunno e[], int p[][16]) {
      int i, k, *nuovo;

      nuovo=(int*)calloc(DIM, sizeof(int));

      for(i=0; strlen(e[i].cognome) > 0; ++i)
            if(media(p[i]) >= 6) nuovo[k++]=i;
      return nuovo;
}
```

```
/*
 * Cancellazione di un alunno con relativa pagella.
 */

int trova(struct alunno e[], char* c, char* n) {
      int i;

      for(i=0; (strlen(e[i].cognome) > 0) && (strcmp(c, e[i].cognome) !=0) &&
(strcmp(n, e[i].nome) !=0); ++i)
            ;
      return i;   /* per non trovato restituisce la posizione del "TAPPO" */
}

void vetcopy(int d[], int s[]) {
      int i;

      for(i=0;s[i]!=0;++i) d[i]=s[i];
      d[i]=0;
}

void cancella(struct alunno e[], int p[][16], char* cognome, char* nome) {
      int i;

      i = trova(e, cognome, nome);

      for( ;strlen(e[i].cognome) > 0; ++i) {
            e[i] = e[i+1];
            vetcopy(p[i], p[i+1]);
      }
}
/*
 * Ordinamento degli alunni in base alla media dei voti.
 */

void swap(struct alunno *a, struct alunno *b) {
      struct alunno tmp;

      tmp = *a;
      *a = *b;
      *b = tmp;
}

void swap1(int a[], int b[]) {
      int i, tmp;

      for(i=0; i<16; ++i) {
            tmp = a[i];
            a[i] = b[i];
            b[i] = tmp;
      }
}

void ordina(struct alunno e[], int p[][16]) {
      int i, k, pmax;

      for(i=0; strlen(e[i].cognome) > 0; ++i){
            pmax=i;
            for(k=i; strlen(e[k].cognome) > 0; ++k)
                  if(media(p[k]) > media(p[pmax])) pmax = k;
            swap(&e[i], &e[pmax]);
            swap1(p[i], p[pmax]);
      }
}
```