

**SECONDA PROVA INTERMEDIA DEL MODULO DI  
ELEMENTI DI INFORMATICA  
CORSO DI LAUREA IN INGEGNERIA BIOMEDICA  
9 gennaio 2019**

**MOTIVARE IN MANIERA CHIARA LE SOLUZIONI PROPOSTE A CIASCUNO DEGLI ESERCIZI SVOLTI**

**ESERCIZIO 1 (28 punti)**

Dopo tre anni di duro sudore davanti ai libri ed al calcolatore, avete deciso di chiedere la tesi a Prof. Marcialis. Questi vi spiega che per una tesi di tipo sperimentale il massimo voto ammissibile è 30, per una di tipo progettuale è 26, per una di tipo bibliografico è 22. In tutti i casi, il numero di crediti associati al voto di tesi è 6. A questo punto avete necessità di capire cosa vi convenga fare per vedere se potete arrivare alla media finale desiderata, partendo dal vostro libretto. Questo è memorizzato in un file "libretto.txt" che è suddiviso in righe di tre colonne, la prima relativa al nome dell'esame (un'unica stringa), la seconda relativa al voto conseguito, la terza relativa ai crediti associati. Il numero di esami contenuti è 27.

Decidete quindi di scrivere un programma in C che, leggendo i dati del libretto e memorizzandoli in un vettore (statico o dinamico a vostra scelta) di 27 variabili strutturate caratterizzate da una stringa di 50 caratteri e due valori interi (la struttura individuale è definita con il nome `datiEsame`), dopo aver letto da tastiera la media desiderata:

- 1) calcoli la media massima ottenibile nei tre casi,
- 2) trovi il valore più vicino a quello della media desiderata,
- 3) in base ad esso scriva su video la tipologia di tesi che conviene chiedere
- 4) e scriva su un secondo file "esami.txt" i nomi degli esami per i quali si è conseguito un voto uguale a quello della media desiderata.

Per fare questo, decidete di scrivere le seguenti funzioni:

- (1) (5 punti) Funzione `calcolaMedia()` : che, ricevendo in ingresso un vettore di variabili strutturate di tipo `datiEsame` e un intero compreso tra 0 e 2, corrispondente alla tipologia di tesi richiesta (0 sta per bibliografica, 1 per progettuale, 2 per sperimentale), restituisce la media ottenuta dai valori presenti nel vettore dato e dal voto massimo associato alla tipologia indicata, senza dimenticare i relativi crediti.
- (2) (8 punti) Una funzione `miglioreTesi()` che ricevendo in ingresso un vettore di tre interi contenente la media ottenuta per i tre tipi di tesi, ed un intero relativo alla media desiderata, restituisca l'indice del vettore relativo al valore più vicino alla media desiderata.
- (3) (7 punti) Una funzione `scriviRisultati()` che, ricevendo in ingresso un vettore di variabili strutturate di tipo `datiEsame` ed un intero pari ad una data media, scriva su file "esami.txt" i nomi degli esami il cui voto è pari a quello della media passata.

Nota: si lascia agli studenti l'eventuale progetto di funzioni non richieste ma finalizzate a migliorare la modularità del codice. Un uso corretto di tale facoltà verrà riconosciuto con l'ulteriore bonus di **5 punti**.

**ESERCIZIO 2 (5 punti)**

1. (2 punti) Definire il tipo `listaConcatenata` caratterizzata da elementi di tipo intero. Il tipo definisce una lista i cui elementi sono concatenati l'uno all'altro tramite puntatori.
2. (3 punti) Scrivere l'implementazione della funzione `CONS` (inserimento dell'elemento in testa alla lista) o delle tre funzioni `HEAD`, `TAIL`, `ISEMPTY` come visto a lezione.

## Soluzione dell'esercizio 1.

Nel seguito una *possibile* soluzione con alcune funzioni, scritte in rosso, per migliorare la modularità.

```
/* Programma che legge libretto da file "libretto.txt",
   e calcola la massima media ottenibile
   a seconda della tesi richiesta e la tesi più adeguata da chiedere al relatore */

#include <stdio.h>
#include <stdlib.h>
#define N 27 /* numero complessivo di esami */

typedef struct
{
    char nome[50];
    int voto, cfu;
} datiEsame;

int calcolaMedia(datiEsame *l, int t)
{
    int i, media, scfu;

    for (i=0, media=0, scfu=6; i<N; i++) /* algoritmo della somma incrementale */
    {
        media = media + l[i].voto*l[i].cfu;
        scfu = scfu + l[i].cfu;
    }

    switch (t)
    {
        case 0: media = media + 22*6; break;
        case 1: media = media + 26*6; break;
        case 2: media = media + 30*6;
    }

    media=media/scfu; /* semplice divisione tra interi dato che media>scfu */

    return media;
}

int miglioreTesi(int *medie, int mdes)
{
    int i, ibest, d, dbest;

    ibest=0;
    dbest=abs(mdes-medie[0]); /* 'distanza' tra la media desiderata e la media di tesi */

    for (i=1; i<3; i++) /* algoritmo del MINIMO elemento in un vettore */
    {
        d=abs(mdes-medie[i]);
        if (d<dbest)
        {
            dbest=d;
            ibest=i;
        }
    }

    return ibest;
}
```

```

void scriviRisultati(datiEsame *l, int mbest)
{
    FILE *f;
    int i;

    f=fopen("esami.txt","w");

    for (i=0; i<N; i++) /* algoritmo di ricerca sequenziale con selezione */
        if (l[i].voto==mbest)
            fprintf(f,"%s\n",l[i].nome);

    fclose(f);
}

datiEsame* leggiDati(char *nomeFile)
{
    FILE *f;
    datiEsame *lib=NULL;
    int i;

    f=fopen(nomeFile, "r");
    if(!f) return NULL;

    lib=(datiEsame*)malloc(sizeof(datiEsame)*N);
    for (i=0; i<N; i++)
        fscanf(f,"%s %d %d",&lib[i].nome[0], &lib[i].voto, &lib[i].cfu);

    fclose(f);

    return lib;
}

void scriviMiglioreTesi(int ibest)
{
    printf("\nTi conviene chiedere una tesi: ");
    switch(ibest)
    {
        case 0: printf("BIBLIOGRAFICA.\n"); break;
        case 1: printf("PROGETTUALE.\n"); break;
        case 2: printf("SPERIMENTALE.\n");
    }
}

int main()
{
    datiEsame *libretto;
    int i, mdes;
    int medie[3];

    libretto=leggiDati("190109_libretto.txt"); /* in rosso le funzioni 'aggiunte' */

    if (libretto) /* equivale all'espressione libretto!=NULL */
    {
        printf("Inserisci la media desiderata: ");
        scanf("%d", &mdes);

        for(i=0; i<3; i++)
            medie[i]=calcolaMedia(libretto, i);

        i=miglioreTesi(&medie[0], mdes);
        scriviMiglioreTesi(i);

        scriviRisultati(libretto, mdes); /* la media più vicina è invece medie[i] */
        free(libretto);
    }
    else
        printf("\nLibretto non leggibile\n");

    return 0;
}

```

## Soluzione dell'esercizio 2.

### Domanda 1.

```
typedef struct ElementoDiLista
{
    int elemento;
    struct ElementoDiLista *successivo;
} Lista;
```

### Domanda 2.

```
/* CONS con inserimento in testa */
Lista *CONS(Lista *l, int v)
{
    Lista* n;

    n=(Lista*)malloc(sizeof(Lista));
    n->elemento=v;
    n->successivo=l;

    return n;
}

/* Altre funzioni TAIL, HEAD, ISEMPY */
Lista *TAIL (Lista *l)
{
    return l->successivo;
}

int HEAD(Lista *l)
{
    return l->elemento;
}

int ISEMPY(Lista *l)
{
    return l==NULL;
}
```