```c
/*
 * -- M.G. Maffucci --
 * 10.
 * 10.1. E' dato un elenco di N studenti con nome, classe, voto
 * finale.
 *
 * 10.2. Stampare il voto piu' basso, quello piu' alto e il nome e
 * la classe degli studenti ai quali appartengono.
 * (NOTA: algoritmo di ricerca del massimo e del minimo).
 *
 * 10.3. Calcolare la media aritmetica dei voti e verificare se e'
 * sufficiente o insufficiente.
 * (NOTA: calcolo aritmetico e condizione).
 *
 * 10.4. Stampare nome e classe degli studenti che hanno il voto
 * finale inferiore alla media aritmetica dei voti.
 * (NOTA: algoritmo di ricerca degli elementi in un vettore che
 * rispondono ad un criterio dato).
 *
 * 10.5. Contare il numero dei promossi e dei respinti e calcolarne
 * le rispettive percentuali.
 * (NOTA: algoritmo di ricerca degli elementi in un vettore che
 * rispondono ad un criterio dato e calcolo aritmetico).
 *
 * 10.6. Stampare l'elenco dei promossi cioe' di coloro che hanno il
 * voto finale sufficiente.
 * (NOTA: algoritmo di ricerca degli elementi in un vettore che
 * rispondono ad un criterio dato; chi volesse fare le cose in modo
 * elegante potrebbe visualizzare l'elenco in ordine alfabetico).
 *
 * NOTA GENERALE: visto che l'esercizio richiede molteplici operazioni
 * e visualizzazioni, dovete necessariamente svolgerlo nel modo piu'
 * ordinato possibile, suddividendolo in punti da affrontare uno
 * per volta. In realta' questo esercizio e' l'unione di piu'
 * esercizi che avete gia' svolto, l'unico problema che in realta'
 * potreste avere e' doverli affrontare tutti in un unico esercizio,
 * ma cio' non vuol dire che dobbiate risolverli tutti insieme ...
 * ... divide et impera.
 */
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define NMAX 2000       //numero massimo di studenti
#define LMAX 20         //lunghezza massima del nome degli studenti

int main()
{
    //input
    char nomi[NMAX][LMAX];      //elenco dei nomi degli studenti
    int classi[NMAX];           //classi; per semplicita' evito di
                                // inserire le sezioni
    int voti[NMAX];             //voto finale di ogni studente
    int n;                      //numero di studenti che si vogliono
                                // caricare
    //lavoro
    int i;                      //indice dei vettori
    int posMin;                 //posizione nel vettore voti[]
                                // del voto minimo
    int posMax;                 //posizione nel vettore voti[]
                                // del voto massimo
    float pPromossi;              //contatore degli studenti promossi
    float pRespinti;              //contatore degli studenti respinti
    //output
    float media;                //media aritmetica dei voti
    /*
```

```c
     * 10.1. E' dato un elenco di N studenti con nome, classe, voto
     * finale.
     * Inserisco il numero di studenti da caricare,
     * controllando la validita' dell'input.
     */
    do{
        printf("Inserisci il numero di studenti da caricare (1-2000): ");
        scanf("%d", &n);
    } while( (n < 1) || (n > NMAX));

    /*
     * Ciclo di caricamento dei tre vettori.
     * Il caricamento viene fatto parallelamente, in un unico ciclo.
     */
    for(i = 0; i < n; i++){
        /*
         * Il while seguente serve solo per ripulire lo standard input (stdin)
         */
        while(getchar() != '\n');
        printf("Inserisci il nome dello studente: ");
        fgets(nomi[i], LMAX, stdin);
        nomi[i][strlen(nomi[i]) - 1] = '\0';
        printf("Inserisci la classe frequentata dallo studente (1-5): ");
        scanf("%d", &classi[i]);
        printf("Inserisci il voto finale dello studente: ");
        scanf("%d", &voti[i]);
    }
    /*
     * 10.2. Stampare il voto piu' basso, quello piu' alto e il nome e
     * la classe degli studenti ai quali appartengono.
     * (NOTA: algoritmo di ricerca del massimo e del minimo).
     */
    for(i = 1, posMin = 0; i < n; i++){
        if(voti[i] < voti[posMin]){
            posMin = i;
        }
    }
    printf("\nVoto piu' basso: %d\nElenco studenti\n", voti[posMin]);
    printf("|\tNome\t     |Classe|\n");
    for(i = 0; i < n; i++){
        if(voti[i] == voti[posMin]){
            /*
             * La stringa di conversione %-20.20s indica che si visualizzara'
             * una stringa allineata a sinistra (-) in uno spazio complessivo
             * di 20 caratteri (20) e si considereranno solo i primi 20
             * caratteri della stringa visualizzata (.20).
             */
            printf("|%-20.20s|%d\t    |\n", nomi[i], classi[i]);
        }
    }
   for(i = 1, posMax = 0; i < n; i++){
        if(voti[i] > voti[posMax]){
            posMax = i;
        }
    }
    printf("\nVoto piu' alto: %d\nElenco studenti\n", voti[posMax]);
    printf("|\tNome\t     |Classe|\n");
    for(i = 0; i < n; i++){
        if(voti[i] == voti[posMax]){
            printf("|%-20.20s|%d\t    |\n", nomi[i], classi[i]);
        }
    }
    /*
     * 10.3. Calcolare la media aritmetica dei voti e verificare se e'
     * sufficiente o insufficiente.
     * (NOTA: calcolo aritmetico e condizione).
```

```c
     */
    for(i = 0, media = 0; i < n; i++){
        media += voti[i];
    }
    media /= n;
    printf("\nLa media scolastica e' %.2f e quindi ", media);
    if(media < 6){
        printf("insufficiente\n");
    } else {
        printf("sufficiente\n");
    }
    /*
     * 10.4. Stampare nome e classe degli studenti che hanno il voto
     * finale inferiore alla media aritmetica dei voti.
     * (NOTA: algoritmo di ricerca degli elementi in un vettore che
     * rispondono ad un criterio dato).
     */
    printf("\nStudenti con voto inferiore alla media scolastica %.2f\n", media);
    printf("|\tNome\t     |Classe|\n");
    for(i = 0; i < n; i++){
        if(voti[i] < media){
            printf("|%-20.20s|%d\t    |\n", nomi[i], classi[i]);
        }
    }
    /*
     * 10.5. Contare il numero dei promossi e dei respinti e calcolarne
     * le rispettive percentuali.
     * (NOTA: algoritmo di ricerca degli elementi in un vettore che
     * rispondono ad un criterio dato e calcolo aritmetico).
     */
    for(i = 0, pPromossi = 0, pRespinti = 0; i < n; i++){
        if(voti[i] >= 6)
            pPromossi++;
        else
            pRespinti++;
    }
    /*
     * Nella stringa costante la sequenza %% permette di visualizzare il
     * carattere %.
     */
    printf("\nPercentuale promossi: %.2f%%\n", (pPromossi*100/n));
    printf("Percentuale respinti: %.2f%%\n", (pRespinti*100/n));
    /*
     * 10.6. Stampare l'elenco dei promossi cioe' di coloro che hanno il
     * voto finale sufficiente.
     * (NOTA: algoritmo di ricerca degli elementi in un vettore che
     * rispondono ad un criterio dato; chi volesse fare le cose in modo
     * elegante potrebbe visualizzare l'elenco in ordine alfabetico).
     */
    printf("\nStudenti promossi\n");
    for(i = 0; i < n; i++){
        if(voti[i] >= 6){
            printf("%s\n", nomi[i]);
        }
    }

    return 0;
}
```