Soluzione Esercizio 1.

```
struct Corso{
    int codice_corso;
    int cfu;
    int voto; /*inizializzato a -1, per indicare esame non superato*/
    int anno_di_verbalizzazione; /*inizializzato a -1, per indicare esame non
superato*/
    struct Corso *next;
};

struct Studente{
    int matricola_studente;
    char nome_e_cognome[50];
    int anno_di_immatricolazione;
    float media_dei_voti; /*inizializzato a -1, per indicare che lo studente non
ha superato tutti gli esami*/
    struct Corso *piano_di_studio;
    struct Studente *next;
};

struct Studente *listaStudenti = NULL;



/*
funzione che presa come parametro la matricola di uno studente, verifica
se lo studente può presentarsi all'esame di laurea (cioè se ha superato
tutti gli esami previsti nel suo piano di studio) e calcoli la sua media
memorizzandola nell'apposito campo della lista principale

parametri in ingresso:
*elenco puntatore alla lista degli studenti
matricola matricola dello studente da cercare
*media media dello studente con la matricola passata

valore restituito:
la funzione restituisce 0 se lo studente NON può presentarsi all'esame di
laurea, -1 se lo studente NON è presente in elenco, 1 altrimenti
*/
int checkCercaStudenteSostenereEsameLaurea(struct Studente *elenco, int
matricola, float *media)
{
    struct Studente *tempStudente;
    int trovato = 0;
    int abilitato = 0;
    *media = 0;

    tempStudente = elenco;
    while (tempStudente != NULL && trovato == 0)
    {
        if (tempStudente->matricola_studente == matricola)
            trovato = 1;
        else
            tempStudente = tempStudente->next;
    }

    abilitato = checkStudenteSostenereEsameLaurea(tempStudente,media);
    if (abilitato > 0)
        tempStudente->media_dei_voti = *media;
    return abilitato;
}
```

```
/*
funzione che indica se lo studente passato in ingresso può presentarsi all'esame
di laurea
(cioè se ha superato tutti gli esami previsti nel suo piano di studio) e calcoli
la sua media
memorizzandola nell'apposito campo della lista principale

parametri in ingresso:
*studente puntatore allo studente
*media media dello studente con la matricola passata

valore restituito:
la funzione restituisce 0 se lo studente NON può presentarsi all'esame di
laurea, -1 se il puntatore allo studente è nullo, 1 altrimenti
*/
int checkStudenteSostenereEsameLaurea(struct Studente *studente, float *media)
{
    struct Corso *tempCorso;
    int countEsameNonSuperato = 0;
    int numeroEsamiSuperati = 0;
    float somma = 0;
    *media = 0;

    if (studente == NULL)
        return -1;

    tempCorso = studente->piano_di_studio;
    while (tempCorso != NULL)
    {
        if (tempCorso->voto > 0)
        {
            somma += tempCorso->voto;
            numeroEsamiSuperati++;
        }
        else
            countEsameNonSuperato++;
        tempCorso = tempCorso->next;
    }
    if (numeroEsamiSuperati > 0)
        *media = somma / numeroEsamiSuperati;
    if (countEsameNonSuperato == 0 && numeroEsamiSuperati > 0) /*se
numeroEsamiSuperati == 0 lo studente non ha piano di studi*/
        return 1;
    else
        return 0;
}


/*
funzione che restituisce la media complessiva degli studenti laureandi

parametri in ingresso:
*elenco puntatore alla lista degli studenti

valore restituito:
media di tutti gli studenti laureandi, -1 se non ci sono studenti laureandi
*/
float calcoloMediaStudenti(struct Studente *elenco)
{
    struct Studente *tempStudente;
    int numeroStudenti = 0;
```

```
    int abilitato = 0;
    float mediaStudente = 0;
    float somma = 0;
    float media = -1;

    tempStudente = elenco;
    while (tempStudente != NULL)
    {
        mediaStudente = 0;
        abilitato =
checkStudenteSostenereEsameLaurea(tempStudente,&mediaStudente);
        if (abilitato == 1)
        {
            numeroStudenti++;
            somma += mediaStudente;
        }
        tempStudente = tempStudente->next;
    }
    if (numeroStudenti > 0)
        media = somma / numeroStudenti;
    return media;
}
```

Soluzione Esercizio 2.

Prima chiamata

1 5 6 2 7 4

Seconda chiamata

4 6 2