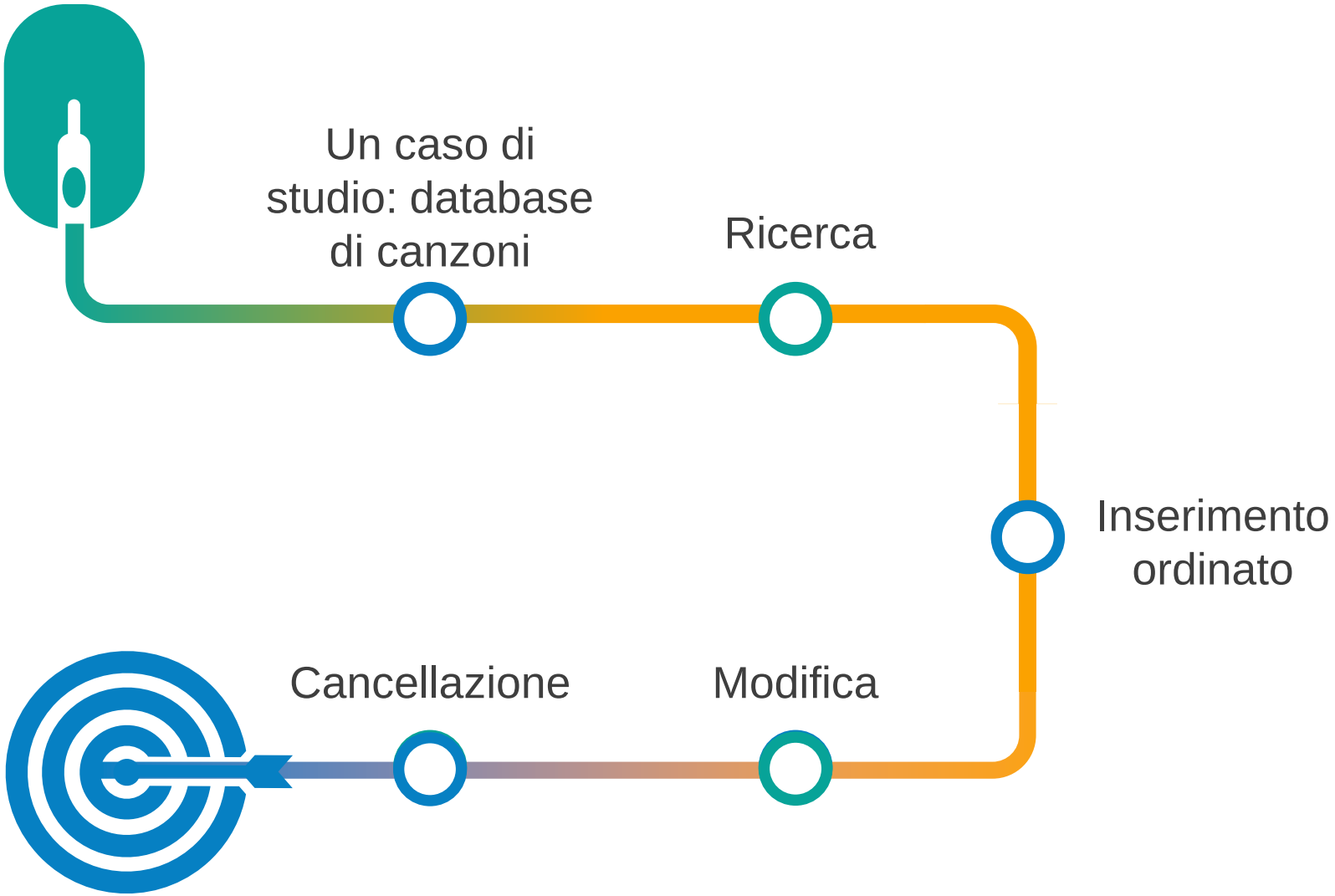
BASI DI DATI

FILE PROCESSING

Polese G. Caruccio L. Breve B.

a.a. 2023/2024

Outline


Un caso di studio: database di canzoni
Ricerca
Inserimento ordinato
Modifica
Cancellazione

Caso di studio: Database di canzoni

- Problema
  - WOLD, una stazione radio locale, vuole costruire un database di canzoni, per automatizzare le ricerche
  - Si è creato un file in cui sono stati inseriti degli elementi composti dai titoli e dai compositori delle canzoni
  - Si intende dare al disk-jockey la possibilità di:
    - cercare nel database tutte le canzoni di un particolare artista
    - inserire una nuova canzone
    - modificare il nome di un artista
    - cancellare un artista con tutte le sue canzoni

Caso di studio: Struttura del file

- Vediamo come strutturare il file
  - Ogni riga dovrà contenere
    - Nominativo artista
    - Titolo canzone
  - Il file dovrà essere ordinato
    - Nominativo artista
    - Titolo canzone, quando il nome dell'artista è lo stesso
  - Gestiamo i duplicati
    - Possono esistere più righe con lo stesso artista
    - Possono esistere più righe con la stessa canzone
    - Non possono esistere due righe uguali

Caso di studio: Ricerca

- Scenario di esempio

```
Inserisci il nome del file contenente il database di canzoni:
  ClassicRock.txt
File ClassicRock.txt loaded.
Inserisci l'artista da cercare:
  Beatles
Canzoni dei Beatles trovate:
Back_in_the_USSR
Paperback_writer
She_Loves_You
Inserisci l'artista da cercare:
  Mozart
Nessuna canzone di Mozart trovata
```

RICERCA: MAIN

- È utile costruire una funzione che si occupi dell'inserimento della riga nel file

```
/* Dichiarazione della funzione di ricerca*/
int search(FILE *f_in);
```

- La procedura generale può essere strutturata nel modo seguente:

1. Richiedi il nome del file su cui effettuare la ricerca
   - Utilizza la funzione "search"
2. Alla fine termina l'esecuzione

RICERCA (1): ESEMPIO

```c
/*Gestione database canzoni*/

#include <stdio.h>

/* Dichiarazione della funzione di ricerca*/
int search(FILE *f_in);

int main(){

    char nome_file[30];     /*nome del file database di canzoni*/

    FILE *f_in;             /*f_in=puntatore del file di input*/

    int res = -1; /* Variabile di controllo sull'output della funzione */

    printf("Inserisci il nome del file contenente il database di canzoni: \n");
    printf("? ");
    scanf("%s", nome_file);
    if((f_in=fopen(nome_file, "r"))== NULL) {
        printf("Il file non puo essere aperto\n\n");
        return -1;
    } /* end if */
    else {
        printf("File %s loaded \n", nome_file);
        res = search(f_in);
    }
    fclose(f_in);
    return 0;
}
```

Ricerca (2): search

- La funzione search può effettuare il seguente controllo iniziale:
  1. Cicla fino a quando si vuole ricercare
     - Richiede il nome dell'artista da ricercare
     - Un nuovo ciclo si occuperà della ricerca delle canzoni per l'artista inserito
  2. Un errore verrà generato non si trova nessuna canzone per quell'artista

- Una variabile `num_canzoni` ci permetterà di controllare se sono state trovate canzoni per l'artista inserito

RICERCA (3): ESEMPIO

```
int search(FILE *f_in) {

    char nome_artista[50];  /*nominativo artista*/
    char canzone[50];

    char nome_artista_new[50];  /*nominativo dell'artista da ricercare*/
    int num_canzoni=0;          /*numero canzoni trovate*/

    printf("Inserisci l'artista da cercare \n");
    printf("Inserisci NULL per terminare la ricerca \n");
    printf("? ");
    scanf("%s", nome_artista_new);

    /*Cicla fino a quando si vuole effettuare la ricerca*/
    while(!(strcmp(nome_artista_new,"NULL") == 0)) {

        num_canzoni=0;

...
```

RICERCA (4): ESEMPIO

```
...
    fscanf(f_in, "%s%s", nome_artista, canzone);

    /*legge i dati dal file*/
    while(!feof(f_in)) {
        if(strcmp(nome_artista,nome_artista_new) == 0) {
            if(num_canzoni==0) {
                printf("canzoni di %s trovate: \n",nome_artista_new);
            }
            num_canzoni++;
            printf("%s\n",canzone);
        }
        else if(strcmp(nome_artista,nome_artista_new) > 0){
            break;
        }
        fscanf(f_in, "%s%s", nome_artista, canzone);
    } /* end while */
    if(num_canzoni==0 && (strcmp(nome_artista_new,"NULL") != 0)) {
        printf("Nessuna canzone di %s trovata\n", nome_artista_new);
    }
    printf("\n");

    printf("Inserisci l'artista da cercare \n");
    printf("Inserisci NULL per terminare la ricerca \n");
    printf("? ");
    scanf("%s", nome_artista_new);
} /* end while */
return 0;
}
```

Caso di studio: Inserimento

- Scenario di esempio

```
Inserisci il nome della canzone da inserire:
  She_Loves_You
 Inserisci l'artista da associare alla canzone
  Beatles
 Canzone già presente nel database
 Inserisci il nome della canzone da inserire:
  Hey_Jude
 Inserisci l'artista da associare alla canzone
  Beatles
 Canzone inserita con successo
```

Inserimento: database canzoni

- Utilizziamo uno o più campi per inserire i dati nel file in maniera ordinata
  - Per il nostro esempio usiamo due campi
    - Nome artista: campo di ordinamento principale
    - Canzone: campo di ordinamento secondario

- Ogni volta che si inserisce una riga
  - Deve essere posizionata in modo corretto
  - Tutti gli altri dati devono continuare a persistere nel file
  - Si deve controllare che la sequenza dei valori di quei campi siano univoci

Inserimento Ordinato: main

- È utile costruire una funzione che si occupi dell'inserimento della riga nel file

```
/* Dichiarazione della funzione di inserimento*/
int insert_into(char *nome_artista_new, char *canzone_new);
```

- La procedura generale può essere strutturata nel modo seguente:

1. finché esistono dati da inserire
   - Utilizza la funzione "insert_into"
2. Alla fine termina l'esecuzione

Inserimento Ordinato (1): Esempio

```c
int main(){

    char nome_artista[50];  /*nominativo artista*/
    char canzone[50] /*titolo canzone*/
    int res = -1; /* Variabile di controllo sull'output della funzione */

    printf("Inserisci il nome della canzone da inserire\n");
    printf("Inserisci NULL per concludere l'inserimento dei dati\n");
    printf("? ");
    scanf("%s", canzone);

    /*Scrive i dati inseriti nel file*/
    while(!(strcmp(canzone,"NULL")==0)) {

        printf("Inserisci l'artista da associare alla canzone\n");
        printf("? ");
        scanf("%s", nome_artista);
        res = insert_into(nome_artista, canzone);
        if(res==0){
            printf("? Canzone %s inserita con successo\n", canzone);
        }
        else {
            printf("? Canzone %s NON inserita\n", canzone);
        }
        printf("Inserisci il nome della canzone da inserire\n");
        printf("Inserisci NULL per concludere l'inserimento dei dati\n");
        printf("? ");
        scanf("%s", canzone);
    } /* end while */
    return 0;   }
```

Inserimento Ordinato (2): insert_into

- In generale dobbiamo utilizzare il metodo del file temporaneo per aggiornare il file originale in modo consistente

```
FILE *f_in;          /*f_in=puntatore del file "classicRock.txt" */
FILE *f_temp;        /*f_temp=puntatore del file temporaneo */
```

- La funzione insert_into può effettuare il seguente controllo iniziale:

1. Se il file non esiste
   - Si apre in scrittura
   - Si scrive direttamente sul file classicRock.txt
2. Un errore verrà generato se non si può aprire neanche in scrittura

Inserimento Ordinato (3): insert_into

- Si parte con il processo di lettura dal file originale e inserimento nel file temporaneo
- È importante controllare il punto in cui inserire la nuova riga:

1. Si legge una riga
2. Finché si può continuare a leggere
   - Controllo se la riga letta ha l'artista uguale
     > Controllo se la riga letta ha la canzone maggiore di quella inserita
       - In caso affermativo, si inserisce la riga nuova
   - se, invece, la riga letta ha l'artista maggiore di quello inserito
     > In caso affermativo, si inserisce la riga nuova
   - Si inserisce la riga letta dal file

Inserimento Ordinato (4): Esempio

```
int insert_into(char *nome_artista_new, char *canzone_new) {

    char nome_artista[50];  /*nominativo artista*/
    char canzone[50];

    FILE *f_in;          /*f_in=puntatore del file "classicRock.txt" */
    FILE *f_temp;        /*f_temp=puntatore del file temporaneo */

    int inserted = 0;  /*Variabile di controllo per l'inserimento */

    /*si esce dal programma se non e' possibile aprire il file */
    if((f_in=fopen("classicRock.txt", "r"))== NULL) {
        if((f_in=fopen("classicRock.txt", "w"))== NULL) {
                printf("Il file non puo essere ne aperto ne creato\n");
                return -1;
        } /* end if */
        else {
            fprintf(f_in,"%s\t %s\n", nome_artista_new, canzone_new);
            fclose(f_in);
            return 0;
        }
    } /* end if */
    ...
```

Inserimento Ordinato (5): Esempio

```
...
else {
    /*si esce dal programma se non e' possibile creare il file */
    if((f_temp=fopen("temp.txt", "w"))== NULL) {
        printf("Il file non puo essere creato\n");
        fclose(f_in);
        return -1;
    } /* end if */
    else {
        fscanf(f_in, "%s%s", nome_artista,canzone);
        /*legge i dati dal file*/
        while(!feof(f_in)) {
            if(strcmp(nome_artista,nome_artista_new) == 0) {
                if(strcmp(canzone,canzone_new) == 0) {
                    printf("Canzone già presente nel database\n");
                    fclose(f_in);
                    fclose(f_temp);
                    remove("temp.txt");
                    return -1;
                }
                else if(strcmp(canzone,canzone_new) > 0) {
                    fprintf(f_temp,"%s\t %s\t\n", nome_artista_new, canzone_new);
                    inserted = 1;
                }
            }
...
```

Inserimento Ordinato (5): Esempio

```
...

else {
    /*si esce dal programma se non e' possibile creare il file */
    if((f_temp=fopen("temp.txt", "w"))== NULL) {
        printf("Il file non puo essere creato\n");
        fclose(f_in);
        return -1;
    } /* end if */
    else {
        fscanf(f_in, "%s%s", nome_artista,canzone);
        /*legge i dati dal file*/
        while(!feof(f_in)) {
            if(strcmp(nome_artista,nome_artista_new) == 0) {
                if(strcmp(canzone,canzone_new) == 0) {
                    printf("Canzone già presente nel database\n");
                    fclose(f_in);
                    fclose(f_temp);
                    remove("temp.txt");
                    return -1;
                }
                else if(strcmp(nome_artista,nome_artista_new) > 0){
                     if(inserted == 0)
                     {
                        fprintf(f_temp,"%s\t %s\t\n", nome_artista_new, canzone_new);
                        inserted = 1;
                     }
                }
...
```

Inserimento Ordinato (6): Esempio

```
...
                else if(strcmp(nome_artista,nome_artista_new) > 0){
                    fprintf(f_temp,"%s\t %s\t\n", nome_artista_new, canzone_new);
                    inserted = 1;
                }
                fprintf(f_temp,"%s\t %s\t\n", nome_artista, canzone);
                fscanf(f_in,"%s%s", nome_artista,canzone);
            } /* end while */
            if(inserted == 0){
                fprintf(f_temp,"%s\t %s\n", nome_artista_new, canzone_new);
            }
            fclose(f_temp); /*chiude il file temporaneo*/
        }
        fclose(f_in); /*chiude il file "classicRock.txt"*/
        remove("classicRock.txt");
        rename("temp.txt", "classicRock.txt");
    }

    return 0;
}
```

Caso di studio: Modifica

- Scenario di esempio

```
Inserisci il nome dell'artista da modificare:
  Beatle
Artista non trovato
Inserisci il nome dell'artista da modificare:
  Beatles
Inserisci il NUOVO nome dell'artista:
  The Beatles
Canzoni a cui è stato sostituito l'artista:
Back_in_the_USSR
Hey_Jude
Paperback_writer
She_Loves_You
```

MODIFICA: MAIN

- È utile costruire una funzione che si occupi dell'inserimento della riga nel file

```
/* Dichiarazione della funzione di modifica*/
int update(char *nome_artista_new);
```

- La procedura generale può essere strutturata nel modo seguente:

1. finché esistono dati da modificare
   - Utilizza la funzione "update"
2. Alla fine termina l'esecuzione

MODIFICA (1): UPDATE

```c
#include <stdio.h>

/* Dichiarazione della funzione di modifica*/
int update(char *nome_artista_new);

int main(){

    char nome_artista[50];  /*nominativo artista*/
    char canzone[50];       /* titolo canzone */
    int res = -1; /* Variabile di controllo sull'output della funzione */

    printf("Inserisci il nome dell'artista da modificare\n");
    printf("Inserisci NULL per concludere la modifica dei dati\n");
    printf("? ");
    scanf("%s", nome_artista);

    /*Modifica i dati inseriti nel file*/
    while(!(strcmp(nome_artista,"NULL")==0)) {
        res = update(nome_artista);
        if(res==0){
            printf("? Artista %s modificato con successo\n", nome_artista);
        }
        else {
            printf("? Artista %s NON modificato\n", nome_artista);
        }
        printf("\n");
        printf("Inserisci il nome dell'artista da modificare\n");
        printf("Inserisci NULL per concludere la modifica dei dati\n");
        printf("? ");
        scanf("%s", nome_artista);
    } /* end while */
    return 0;    }
```

Modifica (2): update

- In generale dobbiamo utilizzare il metodo del file temporaneo per aggiornare il file originale in modo consistente

```
FILE *f_in;          /*f_in=puntatore del file "classicRock.txt" */
FILE *f_temp;        /*f_temp=puntatore del file temporaneo */
```

- La funzione update può effettuare il seguente controllo iniziale:

1. Se il file non esiste
   - Verrà generato un errore

Modifica (3): update

- Si parte con il processo di lettura dal file originale e inserimento nel file temporaneo

- È importante controllare quali righe modificare:

1. Si legge una riga
2. Finché si può continuare a leggere
   - Controllo se la riga letta ha l'artista uguale a quello da modificare
     > In caso affermativo, si inserisce la riga modificata al posto di quella letta
     > In caso negativo, si inserisce la riga letta dal file

Modifica (4): Esempio

```
int update(char *nome_artista_new) {

    char nome_artista[50];  /*nominativo artista*/
    char nome_artista_update[50];   /*nominativo artista*/
    char canzone[50];   /*nome canzone*/

    FILE *f_in;         /*f_in=puntatore del file "studenti.txt" */
    FILE *f_temp;       /*f_temp=puntatore del file temporaneo */

    int inserted = 0;  /*Variabile di controllo per l'inserimento */

    /*si esce dal programma se non e' possibile aprire il file */
    if((f_in=fopen("classicRock.txt", "r"))== NULL) {
            printf("Il file non puo essere ne aperto ne creato\n");
            return -1;
    } /* end if */
    else {
        /*si esce dal programma se non e' possibile creare il file */
        if((f_temp=fopen("temp.txt", "w"))== NULL) {
            printf("Il file non puo essere creato\n");
            fclose(f_in);
            return -1;
        } /* end if */
...
```

Modifica (5): Esempio

```
    else {
        fscanf(f_in, "%s%s", nome_artista, canzone);
        while(!feof(f_in)) {
            if(strcmp(nome_artista,nome_artista_new) == 0) {
                if(inserted==0) {
                    printf("Inserisci il nome del nuovo artista\n");
                    scanf("%s", nome_artista_update);
                    printf("Canzoni a cui e' stato sostituito l'artista\n");
                }
                printf("%s\n", canzone);
                fprintf(f_temp,"%s\t %s\t\n", nome_artista_update, canzone);
                inserted = 1;
            }
            else {
                fprintf(f_temp,"%s\t %s\t\n", nome_artista, canzone);
            }
            fscanf(f_in,"%s%s", nome_artista,canzone);
        }/* end while */
        if(inserted == 0){
            printf("Artista non trovato\n");
            fclose(f_temp);
            fclose(f_in);
            remove("temp.txt");
            return -1;
        }
    }
    fclose(f_temp); /*chiude il file temporaneo*/
}
fclose(f_in); /*chiude il file "classicRock.txt"*/
remove("classicRock.txt");
rename("temp.txt", "classicRock.txt");
return 0;}
```

Caso di studio: Cancellazione

- Scenario di esempio

```
Inserisci il nome dell’artista da cancellare:
   Beatles
  Artista non trovato
  Inserisci il nome dell’artista da cancellare:
   The Beatles
  Artista cancellato:
  The Beatles
  Canzoni cancellate:
  Back_in_the_USSR
  Paperback_writer
  She_Loves_You
  Hey_Jude
```

CANCELLAZIONE: MAIN

- È utile costruire una funzione che si occupi dell'inserimento della riga nel file

```
/* Dichiarazione della funzione di modifica*/
int delete_artista(char *nome_artista_new);
```

- La procedura generale può essere strutturata nel modo seguente:

1. Finché esistono dati da cancellare
   - Utilizza la funzione "delete_artista"
2. Alla fine termina l'esecuzione

CANCELLAZIONE (1): ESEMPIO

```c
#include <stdio.h>

/* Dichiarazione della funzione di modifica*/
int delete_artista(char *nome_artista_new);

int main(){

    char nome_artista[50];  /*nominativo artista*/
    int res = -1; /* Variabile di controllo sull'output della funzione */

    printf("Inserisci il nome dell'artista da cancellare\n");
    printf("Inserisci NULL per concludere la cancellazione dei dati\n");
    printf("? ");
    scanf("%s", nome_artista);
    /*Scrive i dati inseriti nel file*/
    while(!(strcmp(nome_artista,"NULL")==0)) {
        res = delete_artista(nome_artista);
        if(res==0){
            printf("? Artista %s cancellato con successo\n", nome_artista);
        }
        else {
            printf("? Artista %s NON cancellato\n", nome_artista);
        }
        printf("\n");

        printf("Inserisci il nome dell'artista da modificare\n");
        printf("Inserisci NULL per concludere la cancellazione dei dati\n");
        printf("? ");
        scanf("%s", nome_artista);
    } /* end while */
    return 0;    }
```

Cancellazione (2): delete_artista

- In generale dobbiamo utilizzare il metodo del file temporaneo per aggiornare il file originale in modo consistente

```
FILE *f_in;          /*f_in=puntatore del file "classicRock.txt" */
FILE *f_temp;        /*f_temp=puntatore del file temporaneo */
```

- La funzione delete_artista può effettuare il seguente controllo iniziale:

1. Se il file non esiste
   - Verrà generato un errore

Cancellazione (3): delete_artista

- Si parte con il processo di lettura dal file originale e inserimento nel file temporaneo

- È importante controllare quali righe cancellare:

1. Si legge una riga
2. Finché si può continuare a leggere
   - Controllo se la riga letta ha l'artista uguale a quello da modificare
     > In caso affermativo, non si inserisce nulla
     > In caso negativo, si inserisce la riga letta dal file

CANCELLAZIONE (4): ESEMPIO

```
int delete_artista(char *nome_artista_new) {

    char nome_artista[50];  /*nominativo artista*/
    char canzone[50];   /*nome canzone*/

    FILE *f_in;         /*f_in=puntatore del file "studenti.txt" */
    FILE *f_temp;       /*f_temp=puntatore del file temporaneo */

    int deleted = 0;  /*Variabile di controllo per l'inserimento */

    /*si esce dal programma se non e' possibile aprire il file */
    if((f_in=fopen("classicRock.txt", "r"))== NULL) {
            printf("Il file non puo essere ne aperto ne creato\n");
            return -1;
    } /* end if */
    else {
        /*si esce dal programma se non e' possibile creare il file */
        if((f_temp=fopen("temp.txt", "w"))== NULL) {
            printf("Il file non puo essere creato\n");
            fclose(f_in);
            return -1;
        } /* end if */
...
```

Cancellazione (5): Esempio

```
        else {
            fscanf(f_in, "%s%s", nome_artista, canzone);
            /*legge i dati dal file*/
            while(!feof(f_in)) {
                if(strcmp(nome_artista,nome_artista_new) == 0) {
                    if(deleted==0) {
                        printf("Artista cancellato %s\n", nome_artista_new);
                        printf("Canzoni cancellate:\n");
                    }
                    printf("%s\n", canzone);
                    deleted = 1;
                }
                else {
                    fprintf(f_temp,"%s\t %s\t\n", nome_artista, canzone);
                }
                fscanf(f_in,"%s%s", nome_artista,canzone);
            }/* end while */
            if(deleted == 0){
                printf("Artista non trovato\n");
                fclose(f_temp);
                fclose(f_in);
                remove("temp.txt");
                return -1;
            }
        }
        fclose(f_temp); /*chiude il file temporaneo*/
    }
    fclose(f_in); /*chiude il file "studenti.txt"*/
    remove("classicRock.txt");
    rename("temp.txt", "classicRock.txt");
    return 0;
}
```

Caso di studio: Il main completo

- Per avere un unico programma che gestisca il database di canzoni creiamo un main completo
  - Usiamo uno switch case
    - Caso 1: Ricerca
    - Caso 2: Inserimento
    - Caso 3: Modifica
    - Caso 4: Cancellazione
  - Usiamo un ciclo while per gestire la ripetizione delle azioni
    - L'inserimento di -1 determinerà la chiusura del programma

Database canzoni (1): Main

```c
/*Gestione database canzoni*/

#include <stdio.h>

/* Dichiarazione della funzione di ricerca*/
int search(FILE *f_in);

/* Dichiarazione della funzione di inserimento*/
int insert_into(char *nome_artista_new, char *canzone_new);

/* Dichiarazione della funzione di modifica*/
int update(char *nome_artista_new);

/* Dichiarazione della funzione di modifica*/
int delete_artista(char *nome_artista_new);

int main(){

    char nome_file[30];      /*nome del file 'database di canzoni*/
    char nome_artista[50];   /*nominativo artista*/
    char canzone[50];        /* titolo canzone */

    FILE *f_in;              /*f_in=puntatore del file di input*/

    int scelta=0;
    int res = -1; /* Variabile di controllo sull'output della funzione */
...
```

Database canzoni (2): Main

```
...
/*Richiama la funzione per scrivere nel file*/
while(scelta!=-1) {
    printf("Inserisci l'azione da compiere: \n");
    printf("1) Ricerca delle canzoni di un artista \n");
    printf("2) Inserimento di una nuova canzone \n");
    printf("3) Modifica del nome di un artista \n");
    printf("4) Rimozione di un artista \n");
    printf("-1) per terminare \n");
    printf("? ");
    scanf("%d", &scelta);
    switch(scelta) {
        case 1:
            printf("Inserisci il nome del file contenente il database di canzoni: \n");
            printf("? ");
            scanf("%s", nome_file);
            if((f_in=fopen(nome_file, "r"))== NULL) {
                printf("Il file non puo essere aperto\n\n");
            return -1;
            } /* end if */
            else {
                printf("File %s loaded \n", nome_file);
                res = search(f_in);
            }
            fclose(f_in);
            break;

...
```

Database canzoni (3): Main

```
...
case 2:
    res=-1;

    printf("Inserisci il nome della canzone da inserire\n");
    printf("Inserisci NULL per concludere l'inserimento dei dati\n");
    printf("? ");
    scanf("%s", canzone);

    /*Scrive i dati inseriti nel file*/
    while(!(strcmp(canzone,"NULL")==0)) {

        printf("Inserisci l'artista da associare alla canzone\n");
        printf("? ");
        scanf("%s", nome_artista);

        res = insert_into(nome_artista, canzone);
        if(res==0){
            printf("? Canzone %s inserita con successo\n", canzone);
        }
        else {
            printf("? Canzone %s NON inserita\n", canzone);
        }
        printf("\n");

        printf("Inserisci il nome della canzone da inserire\n");
        printf("Inserisci NULL per concludere l'inserimento dei dati\n");
        printf("? ");
        scanf("%s", canzone);
    } /* end while */
    break;
...
```

Database canzoni (4): Main

```
...
    case 3:
        res=-1;

        printf("Inserisci il nome dell'artista da modificare\n");
        printf("Inserisci NULL per concludere la modifica dei dati\n");
        printf("? ");
        scanf("%s", nome_artista);

        /*Scrive i dati inseriti nel file*/
        while(!(strcmp(nome_artista,"NULL")==0)) {

            res = update(nome_artista);
            if(res==0){
                printf("? Artista %s modificato con successo\n", nome_artista);
            }
            else {
                printf("? Artista %s NON modificato\n", nome_artista);
            }
            printf("\n");

            printf("Inserisci il nome dell'artista da modificare\n");
            printf("Inserisci NULL per concludere la modifica dei dati\n");
            printf("? ");
            scanf("%s", nome_artista);
        } /* end while */
        break;
...
```

Database canzoni (5): Main

```
...
        case 4:
            res=-1;

            printf("Inserisci il nome dell'artista da cancellare\n");
            printf("Inserisci NULL per concludere la cancellazione dei dati\n");
            printf("? ");
            scanf("%s", nome_artista);

            /*Scrive i dati inseriti nel file*/
            while(!(strcmp(nome_artista,"NULL")==0)) {

                res = delete_artista(nome_artista);
                if(res==0){
                    printf("? Artista %s cancellato con successo\n", nome_artista);
                }
                else {
                    printf("? Artista %s NON cancellato\n", nome_artista);
                }
                printf("\n");

                printf("Inserisci il nome dell'artista da modificare\n");
                printf("Inserisci NULL per concludere l'inserimento dei dati\n");
                printf("? ");
                scanf("%s", nome_artista);
            } /* end while */
            break;
        default:
            break;
    }
} /* end while */

return 0;    }
```

Caso di studio: Database di canzoni

- Esercizio
  - Aggiungiamo un nuovo file che contenga i dettagli sugli artisti
    - Nominativo
    - Gruppo: SI/NO
    - Età: Rappresenta l'età anagrafica di un artista o gli anni di costituzione di un gruppo
    - Genere principale
  - Query: Visualizzare tutte le canzoni appartenenti ad artisti con un età anagrafica inferiore a 30 o appartenenti a gruppi costituiti da meno di 5 anni
  - Attenzione: È necessario sincronizzare lo scenario di modifica del nome e la cancellazione dell'artista su entrambi i file