

ESEMPIO COMPLETO FILE BINARIO

È dato un file di binario `people.dat` i cui record rappresentano ciascuno i dati di una persona, secondo il seguente formato:

- **cognome** (al più 30 caratteri)
- **nome** (al più 30 caratteri)
- **sesso** (un singolo carattere, 'M' o 'F')
- **anno di nascita**

Si noti che la creazione del file binario deve essere fatta da programma, mentre per i file di testo può essere fatta con un text editor.

CREAZIONE FILE BINARIO

Per creare un file binario è necessario scrivere un programma che lo crei strutturandolo modo che ogni record contenga una **struct persona**

```
struct persona{
    char cognome[31], nome[31], sesso[2];
    int anno;
};
```

I dati di ogni persona da inserire nel file vengono richiesti all'utente tramite la funzione `leggi1()` che non ha parametri e restituisce come valore di ritorno la **struct persona** letta. Quindi il prototipo è:

```
struct persona leggi1();
```

CREAZIONE FILE BINARIO

Mentre la definizione e':

```
struct persona leggiel(){
    struct  persona e;

    printf("Cognome ? ");
    scanf("%s", e.cognome);
    printf("\n Nome ? ");
    scanf("%s", e.nome);
    printf("\nSesso ? ");
    scanf("%s", e.sesso);
    printf("\nAnno nascita ? ");
    scanf("%d", e.anno);
    return e;
}
```

CREAZIONE FILE BINARIO

```
#include <stdio.h>
#include <stdlib.h>
struct persona{
    char cognome[31], nome[31], sesso[2];
    int anno;
};
struct persona leggiel();
main(){
FILE *f; struct persona e; int fine=0;
f=fopen("people.dat", "wb");
    while (!fine)
        { e=leggiel();
          fwrite(&e, sizeof(struct persona), 1, f);
          printf("\nFine (SI=1, NO=0) ? ");
          scanf("%d", &fine);
        }
    fclose(f);
}
```

CREAZIONE FILE BINARIO

L'esecuzione del programma precedente crea il file binario contenente i dati immessi dall'utente. Solo a questo punto il file può essere utilizzato.

Il file `people.dat` non è visualizzabile tramite un text editor: questo è il risultato

```
rossi >    @^   T    8           3
mario               Aw O  F^    
D  M             nuinH2 1"  1~  1
```

ESEMPIO COMPLETO FILE BINARIO

Ora si vuole scrivere un programma che

- legga record per record i dati dal file
 - e ponga i dati in un array di persone
 - (*poi svolgeremo elaborazioni su essi*)
-

ESEMPIO COMPLETO FILE BINARIO

Come organizzarsi?

- 1) Definire una struttura **persona**

Poi, nel main:

- 2) Definire un array di strutture **persona**
- 3) Aprire il file in lettura
- 4) Leggere un record per volta, e porre i dati di quella persona in una cella dell'array
→ Servirà un indice per indicare la prossima cella libera nell'array.

ESEMPIO COMPLETO FILE BINARIO

- 1) Definire una struttura di tipo **persona**

Occorre definire una struct adatta a ospitare i dati elencati:

- **cognome** → array di 30+1 caratteri
- **nome** → array di 30+1 caratteri
- **sex** → array di 1+1 caratteri
- **anno di nascita** → un intero

ricordarsi lo spazio per il terminatore

```
struct persona{
    char cognome[31], nome[31], sesso[2];
    int anno;
};
```

ESEMPIO COMPLETO FILE BINARIO

Poi, nel main:

- 2) definire un array di `struct persona`
- 3) aprire il file in lettura

```
main() {
    struct persona v[DIM];
    FILE* f = fopen("people.dat", "r");
    if (f==NULL) {
        .../* controllo che il file sia
            effettivamente aperto */
    }
    ...
}
```

Hp: massimo DIM
persone

ESEMPIO COMPLETO FILE BINARIO

Poi, nel main:

- 2) definire un array di `struct persona`
- 3) aprire il file in lettura

```
main() {
    struct persona v[DIM];
    FILE* f = fopen("people.dat", "r");
    if (f==NULL) {
        printf("Il file non esiste");
        exit(1); /* terminazione del programma */
    }
    ...
}
```

ESEMPIO COMPLETO FILE BINARIO

Poi, nel main:

- 4) leggere i record dal file, e porre i dati di ogni persona in una cella dell'array

Come organizzare la lettura?

```
int fread(addr, int dim, int n, FILE *f);
```

- legge dal file *n* elementi, ognuno grande *dim* byte (complessivamente, legge quindi $n \times dim$ byte)
- gli elementi da leggere vengono scritti in memoria a partire dall'indirizzo *addr*

Uso fread

ESEMPIO COMPLETO FILE BINARIO

Poi, nel main:

- 4) leggere i record dal file, e porre i dati di ogni persona in una cella dell'array

Cosa far leggere a fread?

- *L'intero vettore di strutture: unica lettura per DIM record*

```
fread(v, sizeof(struct persona), DIM, f)
```

- *Un record alla volta all'interno di un ciclo*

```
i=0
while(!feof(f)) {
    fread(&v[i], sizeof(struct persona), 1, f);
    i++
}
```

ESEMPIO COMPLETO FILE BINARIO

Poi, nel main:

- 4) leggere i record dal file, e porre i dati di ogni persona in una cella dell'array

Dove mettere quello che si legge?

- Abbiamo definito un array di `struct persona`, `v`
- L'indice `k` indica la prima cella libera $\rightarrow v[k]$
- Tale cella è una struttura fatta di *cognome*, *nome*, *sex*, *anno* \rightarrow ciò che si estrae da un record va direttamente nella struttura `v[k]`

ESEMPIO COMPLETO FILE BINARIO

```
#define DIM 30
#include <stdio.h>
#include <stdlib.h>

struct persona{
    char cognome[31], nome[31], sesso[2];
    int anno;
};

main() {
    struct persona v[DIM]; int i=0; FILE* f;
    if ((f=fopen("people.dat", "r"))==NULL) {
        printf("Il file non esiste!"); exit(1); }
    while(fread(&v[i], sizeof(struct persona), 1, f)>0) {
        i++;
    }
}
```

Dichiara la procedura `exit()`

