



Corso di Informatica Medica

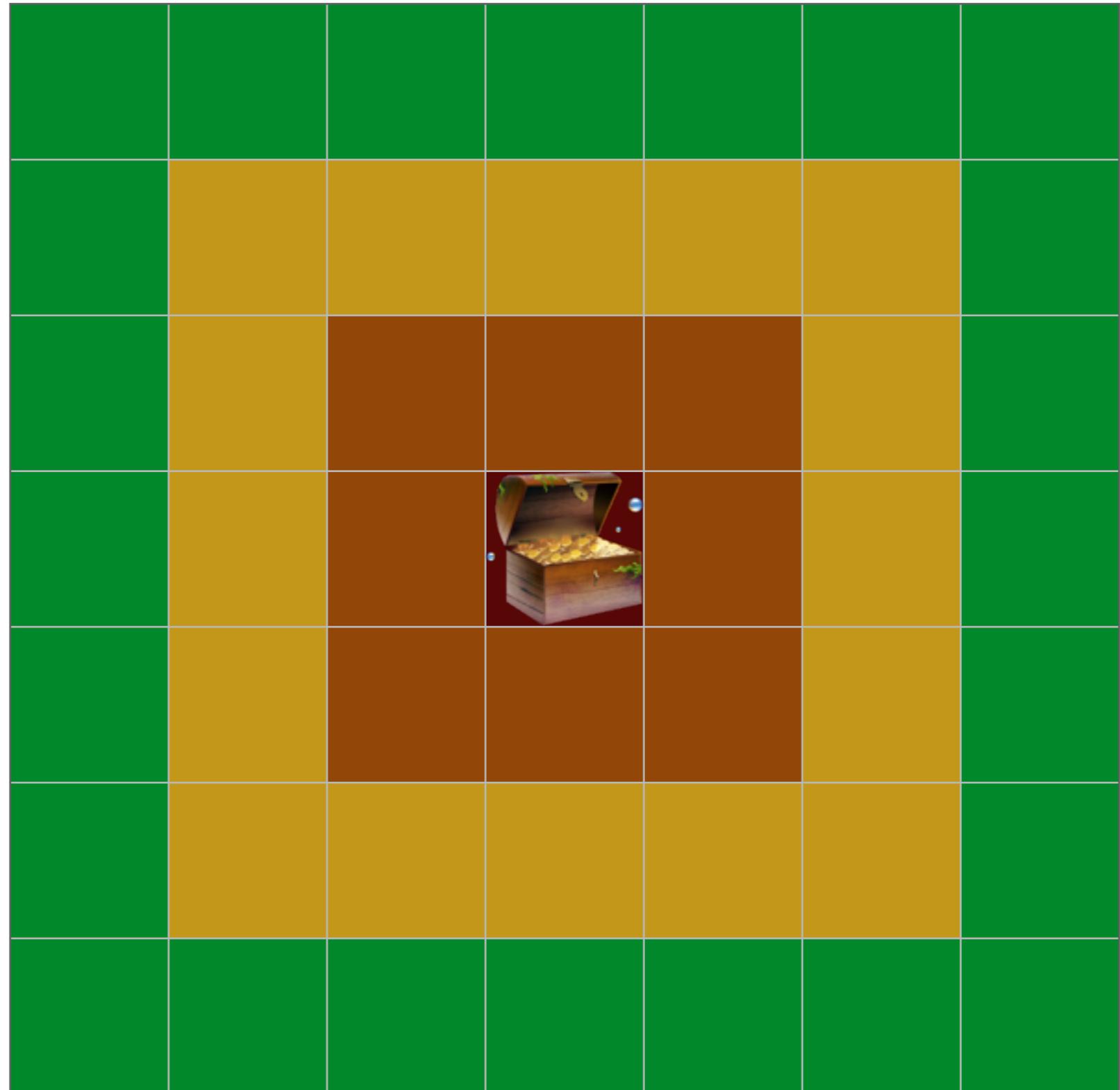
Esercitazione IX

Alessandro A. Nacci
nacci@elet.polimi.it - alessandronacci.com

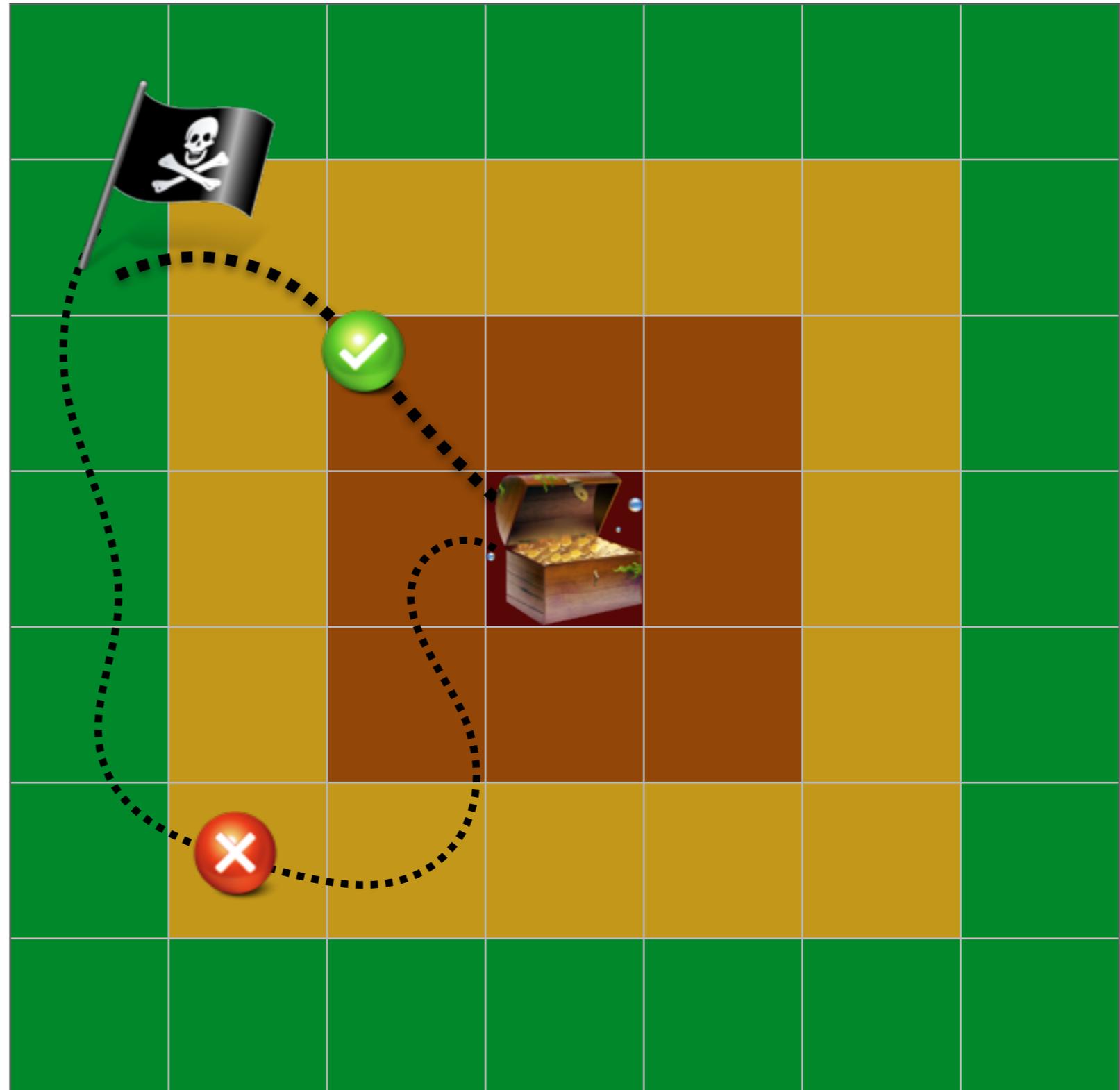
LA MAPPA DEL TESORO



- Immaginiamo di avere una mappa del tesoro... un po particolare!
- E' una sorta di mappa termica radiale della "vicinanza" al tesoro



- Un pirata che vuole muoversi su questa mappa per cercare il tesoro, si sposterà **sempre** dalle celle con colori più freddi verso celle con colori più caldi





- Scrivere un programma C che sia in grado di
 - gestire la mappa appena introdotta
 - date le coordinate del tesoro, creare la *mappa termica di vicinanza*
 - date delle coordinate di partenza, tracciare un percorso valido per arrivare al tesoro



- Come rappresentiamo la mappa termica?



- Come rappresentiamo la mappa termica?
 - Usiamo una matrice
 - I colori diventano numeri



Gestione della mappa termica

- Come rappresentiamo la mappa termica?
 - Usiamo una matrice
 - I colori diventano numeri

11	11	11	11	11	11	11	11	11	11
11	12	12	12	12	12	12	12	12	11
11	12	13	13	13	13	13	13	12	11
11	12	13	14	14	14	14	13	12	11
11	12	13	14	15	14	14	13	12	11
11	12	13	14	14	14	14	13	12	11
11	12	13	13	13	13	13	13	12	11
11	12	12	12	12	12	12	12	12	11
11	11	11	11	11	11	11	11	11	11



Include, define e dichiarazione variabili

```
#include <stdio.h>
#include <math.h>
```

```
#define W 10
#define H 10
```

```
#define VALORE_TESORO 15
```

VALORE_TESORO è legato alla
dimensione della mappa



```
int main()
{
    int mappa[W][H];
    int x,y;
    int trovato = 0;
```



Inizializziamo la mappa

```
void init_mappa(int mappa[W][H])  
{
```

Scrivere una funzione che riempia di '0' una matrice 'mappa' in ingresso.

```
}
```



Inizializziamo la mappa

```
void init_mappa(int mappa[W][H])
{

    int x,y;

    for (x = 0; x < W; x++)
    {
        for (y = 0; y < H; y++)
        {
            mappa[x][y] = 0;
        }
    }

}
```



Stampiamo la mappa

```
void stampa_mappa(int mappa[W][H])  
{
```

Scrivere una funzione che visualizzi a schermo una matrice 'mappa' in ingresso

```
}
```



Stampiamo la mappa

```
void stampa_mappa(int mappa[W][H])
{
    int x,y;

    for (x = 0; x < W; x++)
    {
        for (y = 0; y < H; y++)
        {
            printf("%d\t", mappa[x][y]);
        }
        printf("\n");
    }

    printf("\n\n\n");
}
```



Posiziona il tesoro sulla mappa...

```
void metti_tesoro(int mappa[W][H], int tes_x, int tes_y)
{
```

Scrivere una funzione che, date due coordinate `tes_x` e `tes_y` - che sono le coordinate del tesoro - crei la mappa termica radiale presentata in precedenza all'interno di una matrice 'mappa' passata in ingresso.

```
}
```



Posiziona il tesoro sulla mappa...

```
void metti_tesoro(int mappa[W][H], int tes_x, int tes_y)
{
    int radius;
    int x,y;
    int start_x, start_y;
    int end_x, end_y;

    int val_tesoro = VALORE_TESORO;

    mappa[tes_x][tes_y] = val_tesoro;

    for (radius = 0; radius < W; radius++)
    {
        start_x = tes_x - radius;
        start_y = tes_y - radius;

        end_x = tes_x + radius;
        end_y = tes_y + radius;

        if (start_x >= 0 && start_y >= 0 && end_x < W && end_y < H )
            for (x = start_x; x <= end_x; x++)
                for (y = start_y; y <= end_y; y++)
                    if ( ((x == start_x) || (x == end_x)) || ((y == start_y) || (y == end_y)))
                        mappa[x][y] = val_tesoro - radius;
    }
}
```

radius dovrebbe partire da 1 perchè...

il valore massimo di radius non dovrebbe essere 'w' perchè...

Così sbaglio a gestire i bordi!



Metti tesoro - corretto

```
void metti_tesoro(int mappa[W][H], int tes_x, int tes_y)
{

    int radius;
    int x,y;
    int start_x, start_y;
    int end_x, end_y;

    int val_tesoro = VALORE_TESORO;

    mappa[tes_x][tes_y] = val_tesoro;

    // attenzione alla dimensione massima del raggio!
    for (radius = 1; radius < W; radius++)
    {
        start_x = tes_x - radius;
        start_y = tes_y - radius;

        end_x = tes_x + radius;
        end_y = tes_y + radius;

        for (x = start_x; x <= end_x; x++)
            for (y = start_y; y <= end_y; y++)
                if ( ((x == start_x) || (x == end_x)) || ((y == start_y) || (y == end_y)))
                    if (x >= 0 && y >= 0 && x < W && y < H )
                        mappa[x][y] = val_tesoro - radius;
    }
}
```



Cerchiamo il tesoro e tracciamo il percorso...

```
int cerca_tesoro(int mappa[W][H], int start_x, int start_y)
{
```

Scrivere una funzione C che date due coordinate da cui il pirata parte ('start_x' e 'start_y'), cerchi un percorso corretto per arrivare al tesoro.

Per creare un percorso corretto, dato un punto generico in cui il pirata si trova, il pirata può spostarsi solo in una cella adiacente che abbia un valore superiore a quello della cella corrente.

Una volta che una cella viene visitata, è necessario 'marcarla' per indicare che quella cella è parte del percorso scelto.

```
}
```



Cerchiamo il tesoro e tracciamo il percorso...

```
int cerca_tesoro(int mappa[W][H], int start_x, int start_y)
{
    if (mappa[start_x][start_y] == VALORE_TESORO) return 1;

    int x,y;

    for (x = start_x - 1; x <= start_x + 1; x++)
        for (y = start_y - 1; y <= start_y + 1; y++)
            if (x>0 && y >= 0 && x<W && y<H)
                if (mappa[x][y] > mappa[start_x][start_y])
                {
                    mappa[start_x][start_y] = -1;
                    return cerca_tesoro(mappa, x, y);
                }

    return 0;
}
```



Mostriamo a schermo il percorso...

```
void stampa_percorso(int mappa[W][H])  
{
```

Scrivere una cella che visualizzi in modo chiaro quale è il percorso scelto dal pirata per raggiungere il tesoro.

```
}
```



Mostriamo a schermo il percorso...

```
void stampa_percorso(int mappa[W][H])
{
    int x,y;

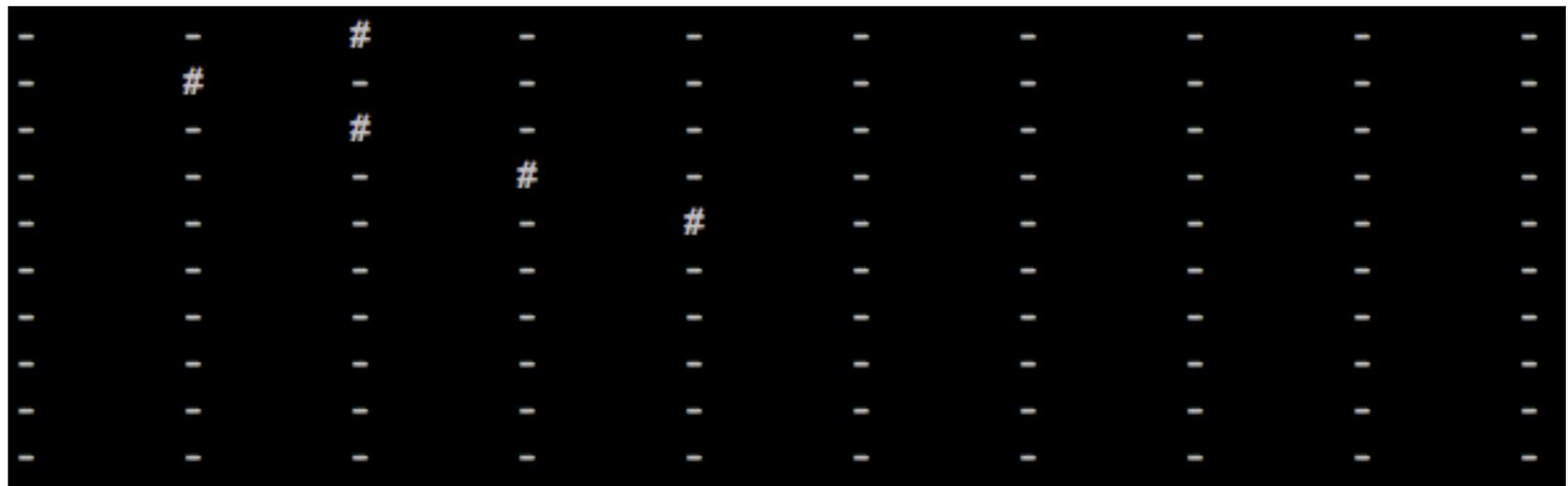
    for (x = 0; x < W; x++)
    {
        for (y = 0; y < H; y++)
        {
            if (mappa[x][y] == -1) printf("#\t");
            else printf("-\t");
        }
        printf("\n");
    }

    printf("\n\n\n");
}
```



E facciamo il main :)

```
int main()  
{  
    int mappa[W][H];  
    int x,y;  
    int trovato = 0;  
  
    init_mappa(mappa);  
    stampa_mappa(mappa);  
  
    metti_tesoro(mappa, 5,5);  
    stampa_mappa(mappa);  
  
    trovato = cerca_tesoro(mappa, 0,2);  
  
    if (trovato) printf("Ho trovato il tesoro!\n");  
  
    stampa_mappa(mappa);  
    stampa_percorso(mappa);  
}
```



GESTIONE AUTOMOBILI





- Rappresentare in C una automobile. Nel nostro caso, una automobile è descritta da un nome, un costo, un colore, da un insieme di componenti e da un libretto di circolazione.
- Un componente ha un nome, un costo ed una categoria. Le categorie possibili sono TRAZIONE, MULTIMEDIA, SICUREZZA
- Il libretto di circolazione riporta invece l'anno e la provincia di immatricolazione e in che classe Euro rientra.
- Il programma deve poter permettere la creazione di auto e la stampa a schermo di tutti i dati relativi ad un'auto
- Deve poter permettere inoltre di modificare il nome dell'auto
- Deve poter calcolare il costo totale per la produzione dell'auto



Automobili: Le strutture dati - Codice C

QUALCHE LEZIONE FA..

AUTOMOBILE

NOME

COSTO

COLORE

COMPONENTI

NOME

COSTO

CATEGORIA

LIBRETTO

ANNO IMM.

PROVINCIA

CLASSE EURO

```
typedef enum {TRAZIONE, MULTIMEDIA,  
             SICUREZZA} tipi_categoria;
```

```
typedef struct {  
    int anno_immatricolazione;  
    char provincia[STR_LEN];  
    int classe_euro;  
} libretto_circolazione;
```

```
typedef struct {  
    char nome[STR_LEN];  
    float costo;  
    tipi_categoria categoria;  
} componente;
```

```
typedef struct {  
    char nome[STR_LEN];  
    float costo;  
    char colore[STR_LEN];  
    int numero_componenti;  
    componente* componenti;  
    libretto_circolazione libretto;  
} automobile;
```



```
automobile crea_auto(char nome[STR_LEN], double costo, char colore[STR_LEN],
                    int numero_componenti, componente* componenti,
                    libretto_circolazione libretto)
{
    printf("Creo una nuova autovettura di nome: %s\n", nome);

    automobile autovettura;

    strcpy(autovettura.nome, nome);
    autovettura.costo = costo;
    strcpy(autovettura.colore, colore);
    autovettura.numero_componenti = numero_componenti;
    autovettura.componenti = componenti;
    autovettura.libretto = libretto;

    return autovettura;
}

libretto_circolazione crea_libretto_circolazione(int anno_immatricolazione,
                                                char provincia[STR_LEN], int classe_euro)
{
    libretto_circolazione libretto;
    libretto.anno_immatricolazione = anno_immatricolazione;
    strcpy(libretto.provincia, provincia);
    libretto.classe_euro = classe_euro;

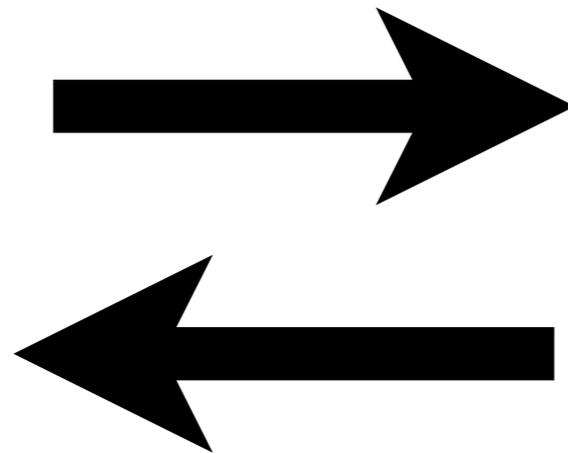
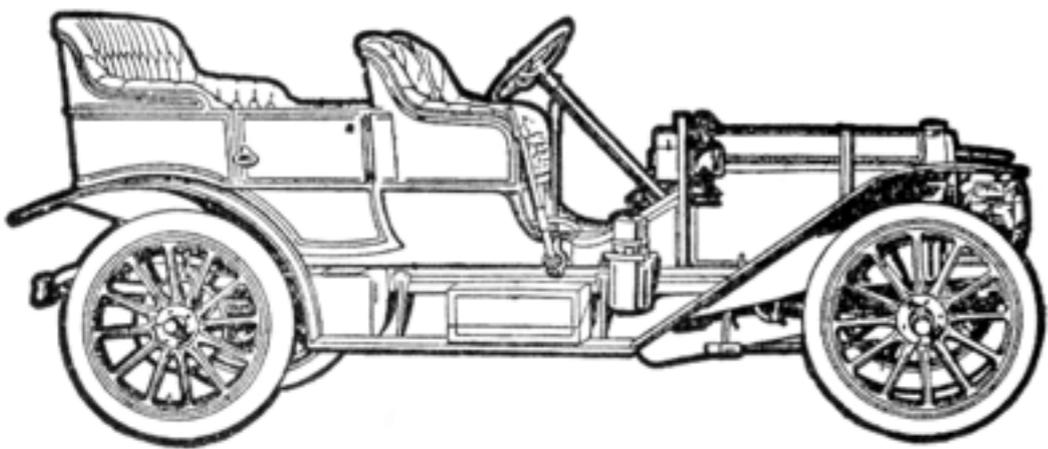
    return libretto;
}

componente crea_componente(char nome[STR_LEN],
                          double costo, int categoria)
{
    componente c;

    strcpy(c.nome, nome);
    c.costo = costo;
    c.categoria = categoria;

    return c;
}
```

- Vogliamo poter salvare tutte le informazioni di una automobile su file e poterle rileggere indietro





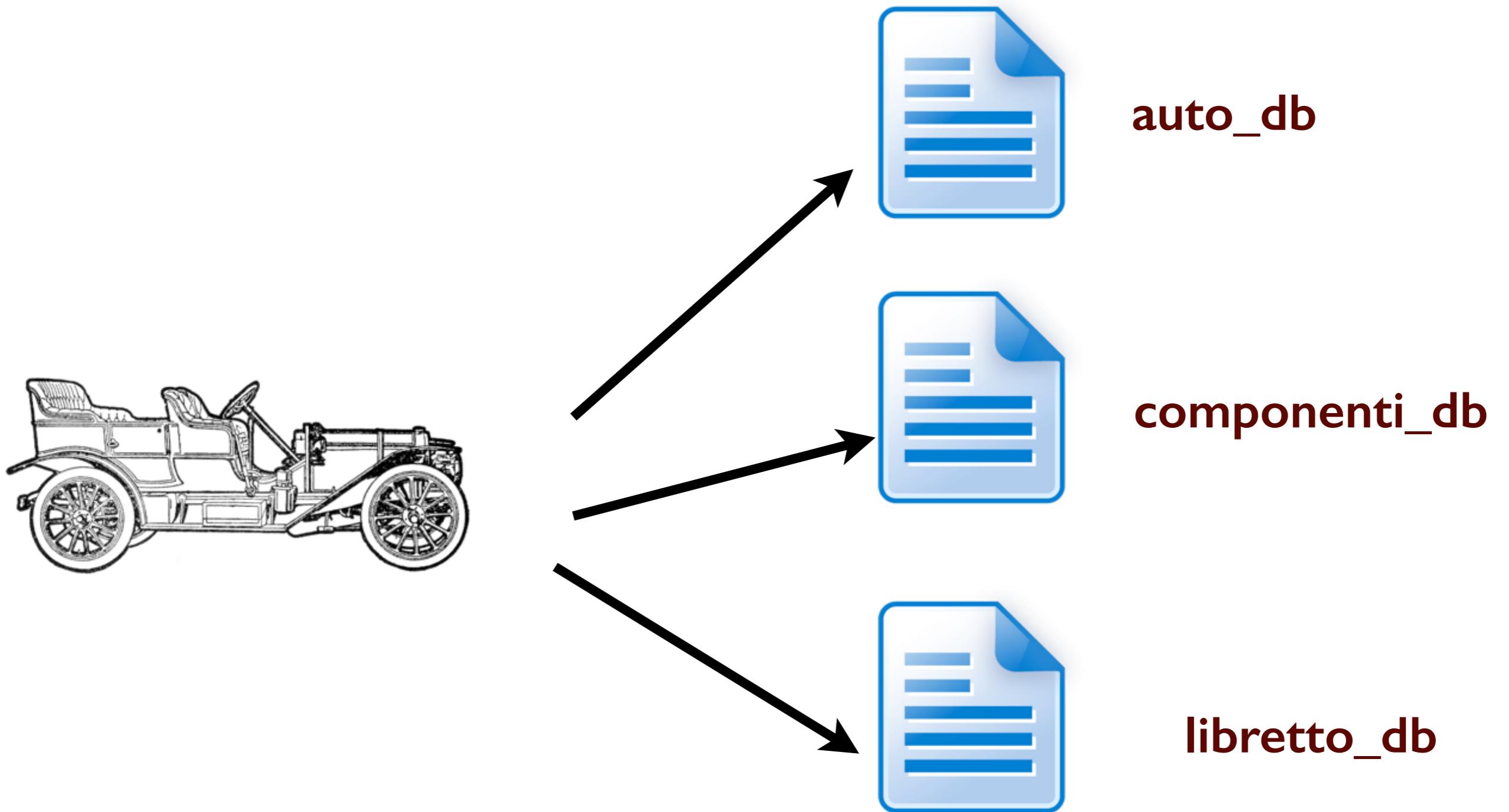
Per poter salvare un'automobile...

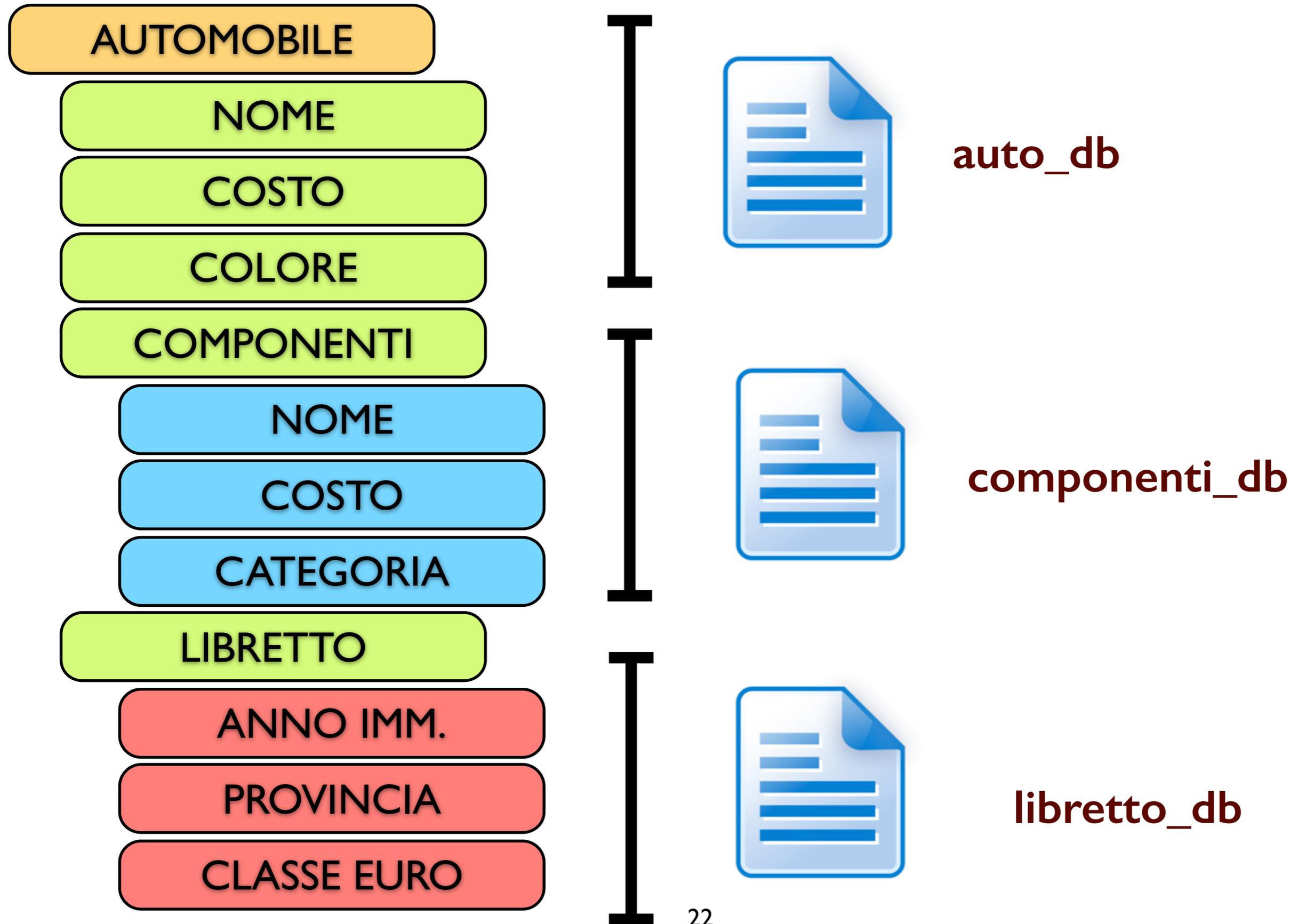


E' UN DATO STRUTTURATO!

**DOBBIAMO POTER
SALVARE I
COMPONENTI**

**DOBBIAMO POTER
SALVARE I
LIBRETTI**







- Ci è comodo avere una funzione per la scrittura di una singola linea di un file...

```
void scrivi_linea_su_file(char linea[MAX_FILE_LINE], char* nome_file, char* mode)
{
```

```
}
```



- Ci è comodo avere una funzione per la scrittura di una singola linea di un file...

```
void scrivi_linea_su_file(char linea[MAX_FILE_LINE], char* nome_file, char* mode)
{
    FILE *fp_mio_file;

    fp_mio_file = fopen (nome_file, mode);

    if (fp_mio_file==NULL)
        printf("Si e' verificato un errore nell'apertura del file.\n");
    else
        printf("File aperto correttamente.\n");

    fprintf(fp_mio_file,"%s",linea);

    if (fclose (fp_mio_file)==0)
        printf("File chiuso correttamente.\n");
}
```




Salviamo un libretto...

```
typedef struct {  
    int anno_immatricolazione;  
    char provincia[STR_LEN];  
    int classe_euro;  
} libretto_circolazione;
```

```
void salva_libretto_circolazione(char* nome_auto, libretto_circolazione libretto)  
{  
    char linea[MAX_FILE_LINE];  
    sprintf(linea, "%s\t%d\t%s\t%d\n", nome_auto,  
            libretto.anno_immatricolazione, libretto.provincia,  
            libretto.classe_euro);  
    scrivi_linea_su_file(linea, "libretto_db", "w");  
}
```



Salviamo un libretto

```
libretto_db
FIAT BRAVO 2010 COMO 5
```

```
typedef struct {
    int anno_immatricolazione;
    char provincia[STR_LEN];
    int classe_euro;
} libretto_circolazione;
```

```
void salva_libretto_circolazione(char* nome_auto, libretto_circolazione libretto)
{
    char linea[MAX_FILE_LINE];
    sprintf(linea, "%s\t%d\t%s\t%d\n", nome_auto,
            libretto.anno_immatricolazione, libretto.provincia,
            libretto.classe_euro);
    scrivi_linea_su_file(linea, "libretto_db", "w");
}
```



Salviamo i componenti...

```
typedef struct {  
    char nome[STR_LEN];  
    float costo;  
    tipi_categoria categoria;  
} componente;
```

```
void salva_componenti(char* nome_auto, componente* componenti, int numero_componenti)  
{
```

```
}
```



Salviamo i componenti...

```
typedef struct {  
    char nome[STR_LEN];  
    float costo;  
    tipi_categoria categoria;  
} componente;
```

```
void salva_componenti(char* nome_auto, componente* componenti, int numero_componenti)  
{  
    int i;  
    char linea[MAX_FILE_LINE];  
  
    for (i = 0; i < numero_componenti; i++)  
    {  
        sprintf(linea, "%s\t%s\t%f\t%s\n", nome_auto, componenti[i].nome,  
                componenti[i].costo, stringa_categoria(componenti[i].categoria));  
        scrivi_linea_su_file(linea, "componenti_db", "a");  
    }  
}
```



Salviamo i componenti...

```
typedef struct {  
    char nome[STR_LEN];  
    float costo;  
    tipi_categoria categoria;  
} componente;
```

nome	costo	categoria
FIAT_PUNTO	420.200012	TRAZIONE
FIAT_PUNTO	656.400024	TRAZIONE

```
void salva_componenti(char* nome_auto, componente* componenti, int numero_componenti)  
{  
    int i;  
    char linea[MAX_FILE_LINE];  
  
    for (i = 0; i < numero_componenti; i++)  
    {  
        sprintf(linea, "%s\t%s\t%f\t%s\n", nome_auto, componenti[i].nome,  
            componenti[i].costo, stringa_categoria(componenti[i].categoria));  
        scrivi_linea_su_file(linea, "componenti_db", "a");  
    }  
}
```



Salviamo l'auto...

```
typedef struct {  
    char nome[STR_LEN];  
    float costo;  
    char colore[STR_LEN];  
    int numero_componenti;  
    componente* componenti;  
    libretto_circolazione librett  
} automobile;
```

```
void salva_auto(automobile autovettura)  
{
```

```
}
```



Salviamo l'auto...

```
typedef struct {
    char nome[STR_LEN];
    float costo;
    char colore[STR_LEN];
    int numero_componenti;
    componente* componenti;
    libretto_circolazione libretto
} automobile;
```

```
void salva_auto(automobile autovettura)
{
    char linea[MAX_FILE_LINE];
    sprintf(linea, "%s\t%s\t%f\t%d\n", autovettura.nome, autovettura.colore,
        autovettura.costo, autovettura.numero_componenti );
    scrivi_linea_su_file(linea, "auto_db", "w");
    salva_componenti(autovettura.nome, autovettura.componenti,
        autovettura.numero_componenti);
    salva_libretto_circolazione(autovettura.nome, autovettura.libretto);
}
```



Salviamo l'auto...

```
auto_db -- Edited
FIAT_BRAVO    BLU    2000.000000    2
```

```
typedef struct {
    char nome[STR_LEN];
    float costo;
    char colore[STR_LEN];
    int numero_componenti;
    componente* componenti;
    libretto_circolazione librett
} automobile;
```

```
void salva_auto(automobile autovettura)
{
    char linea[MAX_FILE_LINE];
    sprintf(linea, "%s\t%s\t%f\t%d\n", autovettura.nome, autovettura.colore,
        autovettura.costo, autovettura.numero_componenti );
    scrivi_linea_su_file(linea, "auto_db", "w");
    salva_componenti(autovettura.nome, autovettura.componenti,
        autovettura.numero_componenti);
    salva_libretto_circolazione(autovettura.nome, autovettura.libretto);
}
```

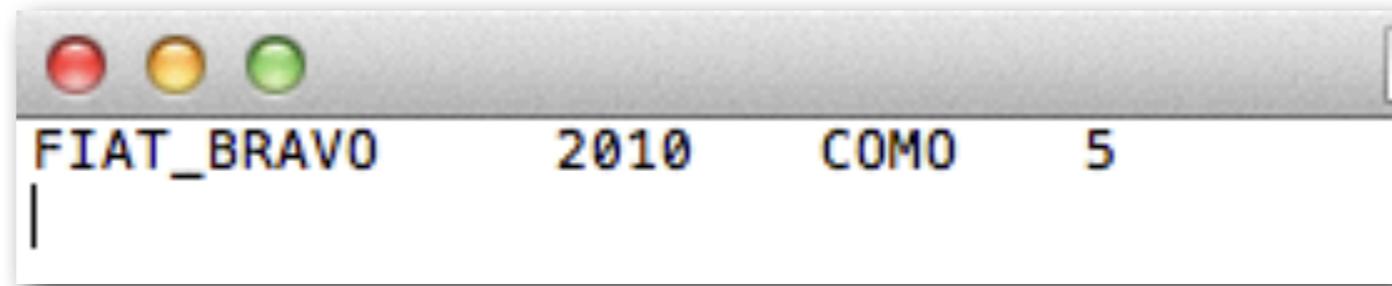


Scriviamo ora il codice
per leggere i dati da file..



Lettura del libretto di circolazione

```
libretto_circolazione carica_libretto()  
{
```



```
FIAT_BRAVO      2010      COMO      5  
|
```

```
}
```

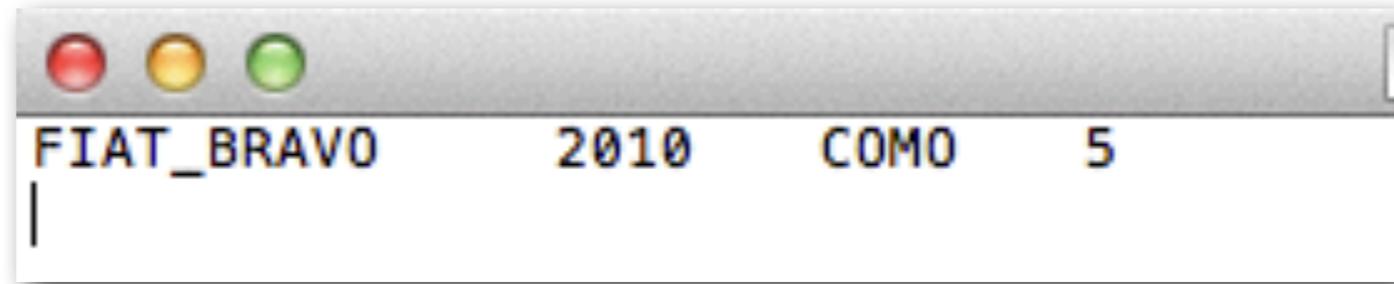


Letture del libretto di circolazione

```
libretto_circolazione carica_libretto()
{
    FILE *fp_mio_file;
    char nome_auto[STR_LEN];
    int anno_immatricolazione;
    char provincia[STR_LEN];
    int classe_euro;

    fp_mio_file = fopen ("libretto_db", "r");
    if (fp_mio_file==NULL)
        printf("Errore apertura file!\n");
    else{
        fscanf(fp_mio_file, "%s", nome_auto);
        fscanf(fp_mio_file, "%d", &anno_immatricolazione);
        fscanf(fp_mio_file, "%s", provincia);
        fscanf(fp_mio_file, "%d", &classe_euro);
    }

    return crea_libretto_circolazione(anno_immatricolazione, provincia, classe_euro);
}
```





Leggiamo i componenti

```
void carica_componenti(int numero_componenti, componente* componenti)  
{
```

Modello	Componente	Prezzo	Tipo
FIAT_PUNTO	FRENO	420.200012	TRAZIONE
FIAT_PUNTO	RUOTA	656.400024	TRAZIONE

Sono più linee!

E' una stringa!

```
}
```




Leggiamo i componenti

```
void carica_componenti(int numero_componenti, componente* componenti)
{
```

Model	Part Name	Price	Category
FIAT_PUNTO	FRENO	420.200012	TRAZIONE
FIAT_PUNTO	RUOTA	656.400024	TRAZIONE

Sono più linee!

E' una stringa!

```
tipi_categoria categoria_stringa(char* categoria_str)
{
    if (strcmp(categoria_str, "TRAZIONE")) return TRAZIONE;
    if (strcmp(categoria_str, "MULTIMEDIA")) return MULTIMEDIA;
    if (strcmp(categoria_str, "SICUREZZA")) return SICUREZZA;

    printf("Errore conversione categoria! - %s\n", categoria_str);

    return 0;
}
```



Leggiamo i componenti

```
void carica_componenti(int numero_componenti, componente* componenti)  
{
```

componenti_db			
FIAT_PUNTO	FRENO	420.200012	TRAZIONE
FIAT_PUNTO	RUOTA	656.400024	TRAZIONE

Sono più linee!

E' una stringa!

```
tipi_categoria categoria_stringa(char* categoria_str)  
{  
  
    if (strcmp(categoria_str, "TRAZIONE")) return TRAZIONE;  
    if (strcmp(categoria_str, "MULTIMEDIA")) return MULTIMEDIA;  
    if (strcmp(categoria_str, "SICUREZZA")) return SICUREZZA;  
  
    printf("Errore conversione categoria! - %s\n", categoria_str);  
  
    return 0;  
}
```



Leggiamo i componenti

```
void carica_componenti(int numero_componenti, componente* componenti)
{
    FILE *fp_mio_file;

    char nome_auto[STR_LEN];
    char nome[STR_LEN];
    float costo;
    char categoria_str[STR_LEN];
    tipi_categoria categoria;
    int i;

    printf("numero c:%d\n", numero_componenti);

    fp_mio_file = fopen ("componenti_db", "r");
    if (fp_mio_file==NULL)
        printf("Errore apertura file!\n");
    else{

        for (i = 0; i < numero_componenti; i++)
        {
            fscanf(fp_mio_file, "%s", nome_auto);
            fscanf(fp_mio_file, "%s", nome);
            fscanf(fp_mio_file, "%f", &costo);
            fscanf(fp_mio_file, "%s", categoria_str);
            categoria = categoria_stringa(categoria_str);

            componenti[i] = crea_componente(nome, costo, categoria);
        }
    }

    fclose(fp_mio_file);
}
```

nome_auto	nome	costo	categoria
FIAT_PUNTO	FRENO	420.200012	TRAZIONE
FIAT_PUNTO	RUOTA	656.400024	TRAZIONE



Leggiamo l'automobile

```
automobile carica_automobile(componente* componenti)  
{
```

The screenshot shows a terminal window titled "auto_db -- Edited". The window displays a single line of text representing a database record: "FIAT_BRAVO BLU 2000.000000 2".

FIAT_BRAVO	BLU	2000.000000	2
------------	-----	-------------	---

```
}
```



Leggiamo l'automobile

```
automobile carica_automobile(componente* componenti)
{
    FILE *fp_mio_file;
    char nome[STR_LEN];
    char colore[STR_LEN];
    float costo;
    int numero_componenti;
    libretto_circolazione libretto;

    fp_mio_file = fopen ("auto_db", "r");
    if (fp_mio_file==NULL)
        printf("Errore apertura file!\n");
    else{
        fscanf(fp_mio_file, "%s", nome);
        fscanf(fp_mio_file, "%s", colore);
        fscanf(fp_mio_file, "%f", &costo);
        fscanf(fp_mio_file, "%d", &numero_componenti);
    }

    fclose(fp_mio_file);
    carica_componenti(numero_componenti, componenti);
    libretto = carica_libretto();

    automobile autovettura = crea_auto(nome, costo, colore,
        numero_componenti, componenti, libretto);

    return autovettura;
}
```

FIAT_BRAVO	BLU	2000.000000	2
------------	-----	-------------	---



Finiamo il main() ...

```
int main () {

    automobile autovettura;
    automobile autovettura_da_file;
    componente componenti[MAX_COMP];
    libretto_circolazione libretto;

    // Creiamo il componente "FRENO"
    componenti[0] = crea_componente("FRENO", 420.20, TRAZIONE);
    componenti[1] = crea_componente("RUOTA", 656.40, TRAZIONE);

    // Creiamo le informazioni del libretto
    libretto = crea_libretto_circolazione(2010, "COMO", 5);

    // Creiamo una autovettura
    autovettura = crea_auto("FIAT_BRAVO", 2000.00, "BLU", 2, componenti, libretto);

    // Stampiamo quello che abbiamo creato
    printf("\nBenvenuto!\n\n");
    stampa_auto(autovettura);

    salva_auto(autovettura);

    printf("\n\n\n");

    autovettura_da_file = carica_automobile(componenti);
    stampa_auto(autovettura_da_file);

    return 0;
}
```

Potete lasciare il vostro giudizio qui:

<http://tinyurl.com/IEIMExe2014>

**Tutte il materiale sar  disponibile sul mio
sito internet:**

alessandronacci.com

