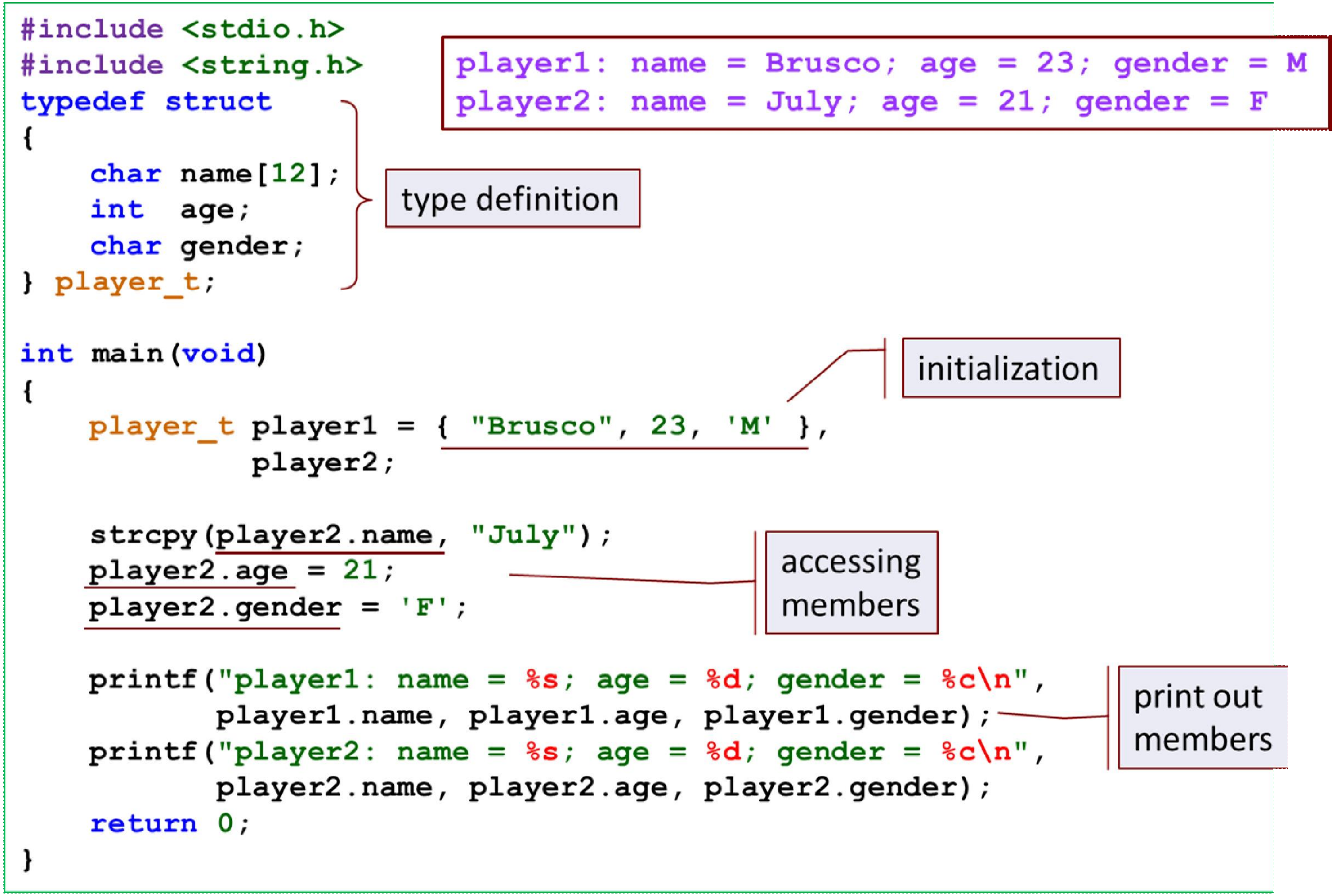
Data Structure

NOTES

Defining Structures Data Types

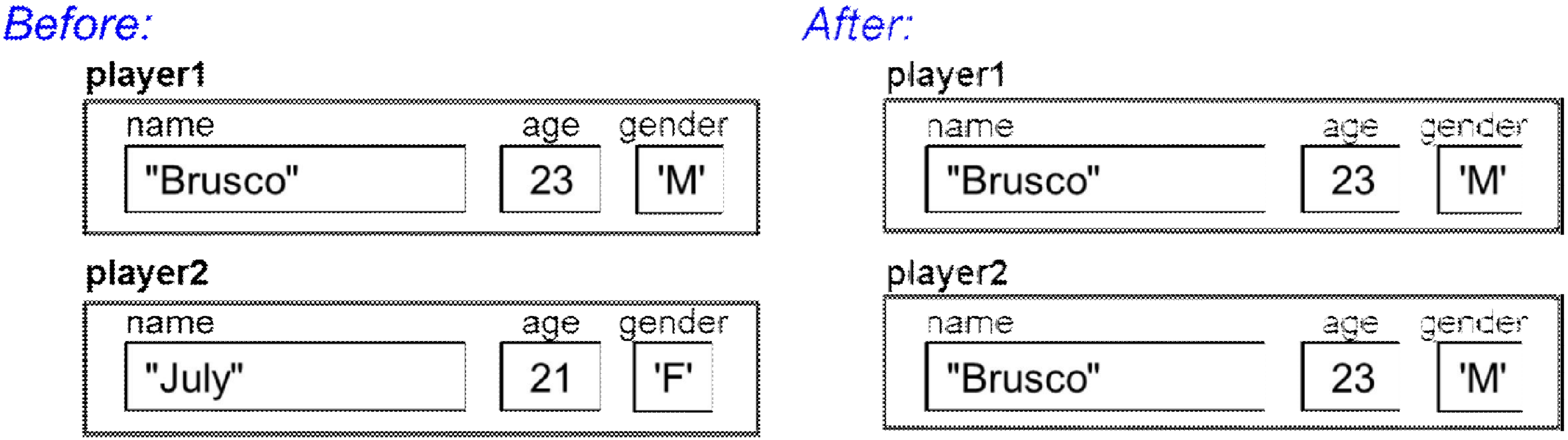


```
#include <stdio.h>
#include <string.h>
typedef struct
{
    char name[12];
    int  age;
    char gender;
} player_t;

int main(void)
{
    player_t player1 = { "Brusco", 23, 'M' },
             player2;

    strcpy(player2.name, "July");
    player2.age = 21;
    player2.gender = 'F';

    printf("player1: name = %s; age = %d; gender = %c\n",
           player1.name, player1.age, player1.gender);
    printf("player2: name = %s; age = %d; gender = %c\n",
           player2.name, player2.age, player2.gender);
    return 0;
}
```

```
player1: name = Brusco; age = 23; gender = M
player2: name = July; age = 21; gender = F
```

Assigning Structures

```
player2 = player1;
```



Before:

player1

name "Brusco"

age 23

gender 'M'

player2

name "July"

age 21

gender 'F'

After:

player1

name "Brusco"

age 23

gender 'M'

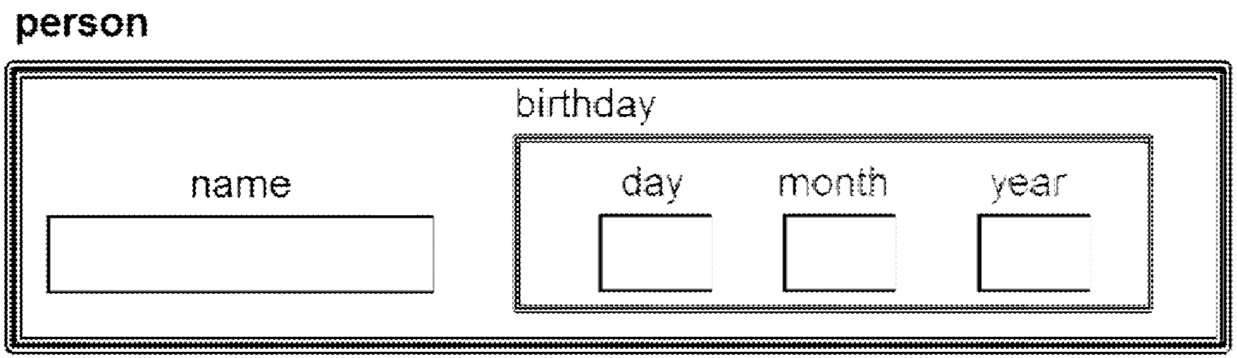
player2

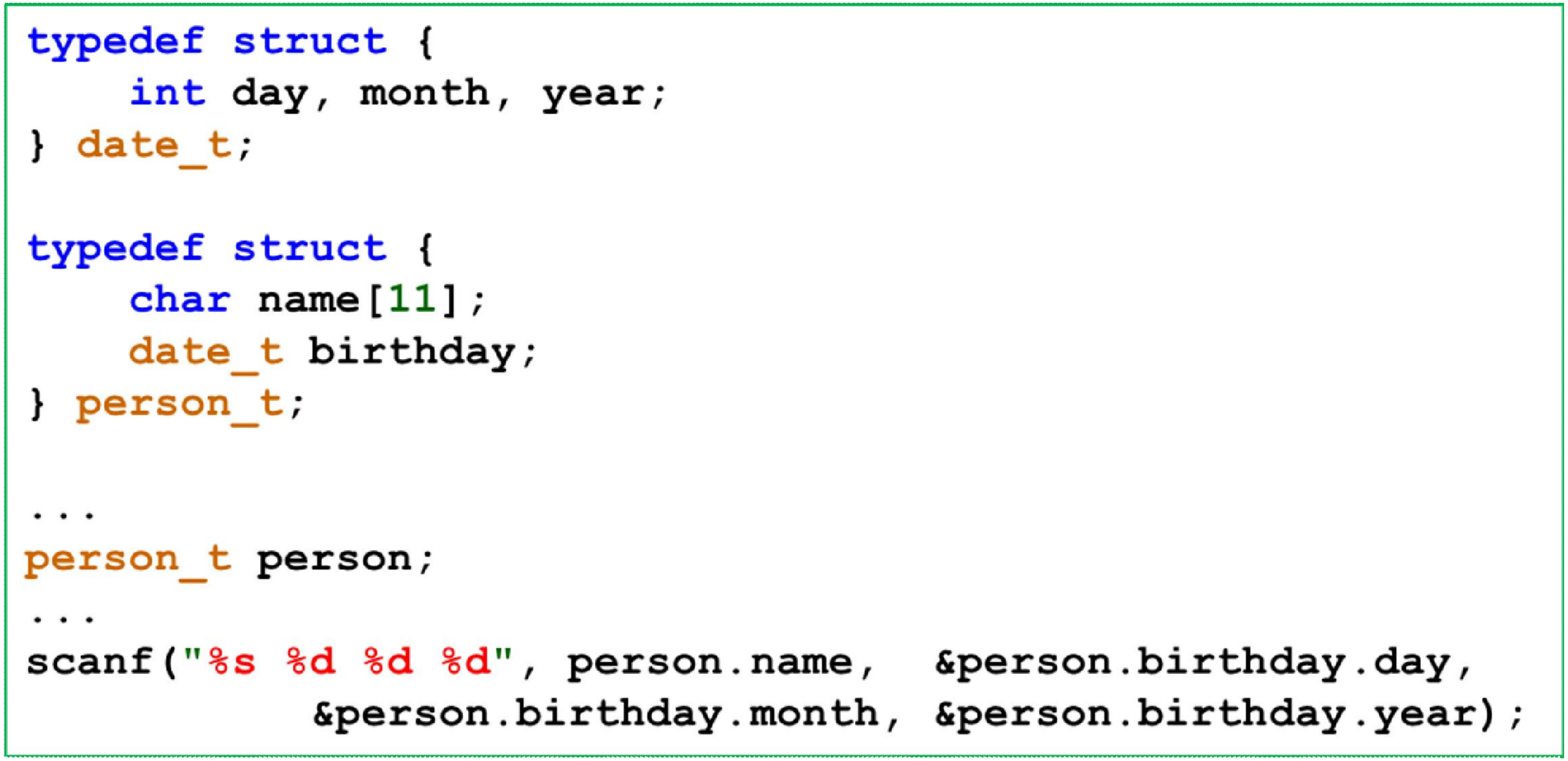
name "Brusco"

age 23

gender 'M'

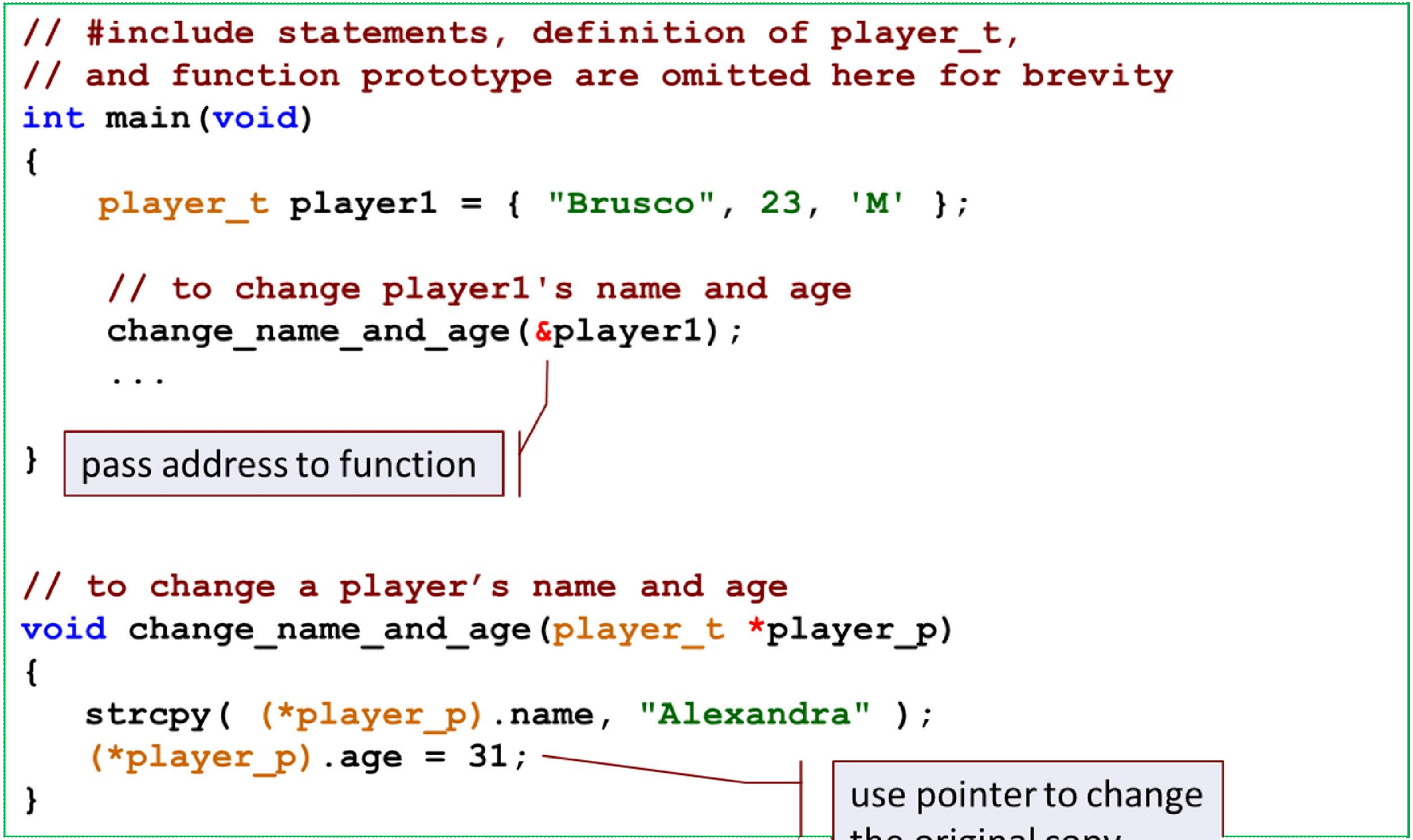
Nested Structure





```
typedef struct {
    int day, month, year;
} date_t;

typedef struct {
    char name[11];
    date_t birthday;
} person_t;

...
person_t person;
...
scanf("%s %d %d %d", person.name,  &person.birthday.day,
           &person.birthday.month, &person.birthday.year);
```

Passing to function



```
// #include statements, definition of player_t,
// and function prototype are omitted here for brevity
int main(void)
{
   player_t player1 = { "Brusco", 23, 'M' };

    // to change player1's name and age
    change_name_and_age(&player1);
    ...

}


// to change a player’s name and age
void change_name_and_age(player_t *player_p)
{
   strcpy( (*player_p).name, "Alexandra" );
   (*player_p).age = 31;
}
```

pass address to function

use pointer to change the original copy