9 Array, strutture e funzioni

Introduzione al Passaggio dei Parametri

Il passaggio dei parametri in C può avvenire principalmente in due modi: per valore o per indirizzo. Nel primo caso, il valore del parametro attuale viene copiato nel parametro formale, mentre nel secondo caso è l'indirizzo del parametro attuale a essere copiato, consentendo modifiche dirette. Per il passaggio per indirizzo, si utilizza l'operatore `*` per accedere al parametro formale, mentre per il passaggio per valore è sufficiente elencare i parametri nella funzione.

Gli array rappresentano un'eccezione: sono passati automaticamente per riferimento senza dover usare l'operatore di indirezione `*` o il simbolo `&` durante la chiamata. Questo implica che tutte le modifiche effettuate su un array all'interno di una funzione si riflettono automaticamente sul parametro attuale.

Lavorare con gli Array

Un array passato per riferimento permette di realizzare funzioni che modificano direttamente il suo contenuto. Ad esempio, per riempire e visualizzare un array, è necessario scrivere un algoritmo che utilizzi procedure o funzioni specifiche. Analogamente, è possibile scrivere programmi per trovare il massimo valore in un array usando una combinazione di algoritmi, funzioni e procedure.

Strutture e Funzioni

In C, è possibile passare strutture come parametri di input o output di una funzione. Quando una struttura è passata per valore, tutte le componenti del parametro attuale vengono copiate nel parametro formale. Nel caso in cui una struttura sia passata per indirizzo, si utilizza l'operatore `&`, come per altri tipi di dati. Per accedere ai membri di una struttura tramite un puntatore, si utilizza l'operatore `->`, che semplifica l'accesso rispetto alla sintassi alternativa `(*nome_struttura).componente`.

Un esempio di struttura può essere una definizione per uno studente:

```
typedef struct {
    char nome[50];
    char cognome[50];
    int matricola;
} studente_t;
```

Questa struttura può essere passata a una funzione per elaborare i dati relativi a uno studente, per esempio calcolare la media dei voti o stampare le informazioni personali.

Restituire Strutture da Funzioni

Contrariamente agli array, le strutture possono essere restituite da una funzione come se fossero un dato elementare. Questo consente di costruire funzioni che generano una struttura, ad esempio per raccogliere dati da un utente o calcolare informazioni derivate. Esempio:

```
studente_t creaStudente(char* nome, char* cognome, int matricola) {
    studente_t s;
    strcpy(s.nome, nome);
    strcpy(s.cognome, cognome);
    s.matricola = matricola;
    return s;
}
```

Array di Record

Questo esercizio richiede di estendere il concetto di array combinandolo con strutture. Si tratta di creare un array in cui ogni elemento è una struttura, per esempio un catalogo di libri o una lista di studenti. Questa tecnica permette di organizzare e gestire grandi quantità di dati strutturati.

Esempio:

```
libro_t catalogo[100];  // Array di libri
int aggiungiLibro(libro_t catalogo[], int indice, libro_t libro) {
    catalogo[indice] = libro;
    return indice + 1;
}
```