4 - Tipi definiti dall'utente, definizione di tipo e enumerativi

Tipi definiti dall'utente

Sono tipi di dati creati e gestiti direttamente dal programmatore, utilizzando le strutture e le parole chiave del linguaggio. Questi tipi permettono una maggiore flessibilità nella programmazione rispetto ai tipi predefiniti del linguaggio.

Definizione di Tipo

Una definizione di tipo in C può:

- Specificare i valori che caratterizzano il tipo.
- Strutturare insiemi di valori.
  Ad esempio:

```
typedef struct {
    int giorno;
    int mese;
    int anno;
} Data_t;
```

Qui, `Data_t` è un nuovo tipo che rappresenta una struttura contenente informazioni sulla data.

Utilizzo della parola chiave typedef

`typedef` permette di assegnare un nome a un nuovo tipo. La sintassi generale è:

```
typedef <tipo_esistente> <nome_nuovo_tipo>;
```

Esempio:

```
typedef unsigned int uint_t;
```

Tipi di dati enumerativi

Un tipo enumerativo (`enum`) consente di definire un tipo di dato che può assumere valori solo in un insieme finito e predefinito. La sintassi è:

```
typedef enum {
    LUNEDI,
    MARTEDI,
    MERCOLEDI,
    GIOVEDI,
    VENERDI,
    SABATO,
    DOMENICA
} Giorno_t;
```

Le caratteristiche principali sono:

1. Gli elementi di un tipo enumerativo sono mappati automaticamente a valori interi (di default da 0 in poi).
2. È possibile utilizzare operatori relazionali e aritmetici con variabili di tipo enumerativo. Esempio di utilizzo:

```
Giorno_t oggi = MERCOLEDI;
if (oggi < VENERDI) {
    printf("La settimana lavorativa non è finita.\n");
}
```

Suggerimenti e Considerazioni

1. Il tipo enumerativo non è circolare: se si utilizza `DOMENICA`, non c'è un ritorno automatico a `LUNEDI`.
2. È possibile forzare il primo valore a un valore specifico:

   ```
   typedef enum {
       LUNEDI = 1,
       MARTEDI,
       MERCOLEDI
   } GiornoSettimana_t;
   ```

   Qui, `MARTEDI` avrà automaticamente il valore 2, e così via.