Tipi derivati: struct
File

Andrea Marin

Università Ca' Foscari Venezia
Laurea in Informatica
Corso di Programmazione

a.a. 2012/2013

Section 1

Struct

Aggregazione di dati eterogenei

- Un array è un tipo derivato che può essere visto come un'aggregazione di dati dello stesso tipo (omogenei)
  - L'accesso ad ognuno di questi avviene mediante un indice
- In C è possibile definire anche aggregazioni eterogeneee
  - Per esempio una data potrebbe essere composta da un `int` che rappresenta il giorno, una stringa di 10 caratteri che rappresenta il mese, e un altro `int` che denota l'anno
- Per definire queste aggregazioni si usa lo `struct`

Esempio

```
struct data {
    int giorno;
    char mese[11];
    int anno;
};

struct data variabile;
```

- `variabile` è una variabile di tipo `struct data`

Accesso ai campi della struttura

- Per leggere o scrivere in una locazione di una struttura si usa la sintassi *nome_variabile.nome_campo*
- Esempi:
  - `varibile.giorno = 30;`
  - `strcpy(variabile.mese,‘‘Marzo’’);`
  - `variabile.giorno = variabile.giorno-10;`

Semplificare la sintassi: `typedef`

- La keyword `typedef` consente di definire nuovi tipi
- Esempio:

```
typedef long int** miotipo;
```

  Farà sì che il compilatore sostiuisca ogni occorrenza di `miotipo` con `long int**`

- Nel caso di struct:

```
typedef struct data tdata;
```

  consente di definire variabili di tipo `struct data` semplicemente come:

```
tdata compleanno;
```

Allocazione dinamica di strutture

- Come le normali variabili anche le strutture possono essere allocati in heap

```
struct data{
    int gg;
    char mese[11];
    int mese;
};

typedef struct data tdata;

int main() {
   tdata* pnatale;
   pnatale = malloc(sizeof(tdata));
   if (pnatale) {
      (*pnatale).gg=25;
      strcpy( (*pnatale).mese, ``Dicembre'');
      (*pnatale).anno=2011;
      ..
      free(pnatale);
   }
}
```

L'operatore $->$

- se pc è un puntatore ad una struttura, per accedere ad un campo, al posto della sintassi:

  ```
  (*pc).campo
  ```

  si può usare la sintassi:

  ```
  pc->campo
  ```

- Esempio:

  ```
  pnatale->gg = 25;
  ```

Section 2

I File

Introduzione

- Il linguaggio C non mette a disposizione istruzioni di I/O
- L'I/O si effettua facendo uso di chiamate a funzioni di libreria
  - Esempio: stdio.h
- Le funzioni di lettura e scrittura si basano sul concetto di *stream*
  - Sia che si acquisiscano da tastiera che si leggano dati da un file...

Definizioni

- Flusso (stream): è l'interfaccia di lettura o scrittura
- File: è il dispositivo effettivo dal quale si legge o si scrive

Flussi

- Ad ogni dispositivo fisico viene associato un dispositivo logico chiamato flusso
- Poichè tutti i flussi hanno le stesse funzionalità possono essere usati in modo analogo (astrazione)
- Tipi di flussi
  - Binari: flusso di byte con corrispondenza 1-1 con i byte del dispositivo fisico
  - Testo: Sequenza di carattere generalmente suddivisa in linee dal carattere di newline

File

- Secondo la definizione data, esempi di file sono un disco, un terminale o una stampante
- L'operazione di apertura consiste nell'associare uno stream ad un file
- Ad ogni operazione di apertura deve corrispondere una conseguente chiusura

Flussi e codifica in memoria

- Ogni flusso ha associato una struttura chiamata FILE contenente i seguenti campi:
  - Modalità di utilizzo (lettura, scrittura o lettura/scrittura)
  - Posizione corrente sul file
  - Indicatore di errore di I/O
  - Indicatore di End Of File (EOF)

Apertura di un file

- L'apertura di un file avviene mediante l'operazione `fopen` in cui prototipo è;

```
FILE* fopen(char *nome, char *modo)
```

- `nome` è il nome del dispositivo fisico
- `modo` è la modalità di apertura
- La funzione ritorna l'indirizzo di allocazione della struttura FILE

Modalità di apertura

Modalità	Descrizione
r	Apre file di testo in lettura
w	Crea file di testo in scrittura
a	Apre file testo in append
rb	Apre file binario in lettura
wb	Crea file binario in scrittura
ab	Apre file binario in append
r+	Apre file di testo in lettura/scrittura
w+	Crea file di testo in lettura/scrittura
a+	Apre o crea file testo in append lettura/scrittura
r+b	Apre file binario in lettura/scrittura
w+b	Crea file binario in lettura/scrittura
a+b	Apre o crea file binario in append lettura/scrittura

Aprire un file di testo in lettura

```
int main(){
   FILE *f;
   f = fopen(``parole.txt'',``r'');
   if (f == NULL) {
        printf(``Errore'');
   }
   else ..
```

Chiusura di un file

- La funzione `fclose` chiude un file

```
int fclose(FILE* f);
```

- `f` è il puntatore alla struttura FILE associata allo stream del file

Fine del file

- La funzione `feof` restituisce un valore che codifica true se si è raggiunta la fine del file, false altrimenti.

```
int feof(FILE *f);
```

Leggere e scrivere da file

- La lettura e scrittura avvengono come per l'I/O da standard input
- Le funzioni `fscanf` e `fprintf` si comportano come le analoghe già note ma hanno come primo parametro lo stream sulle quale lavorano

Esempio: Lettura e stampa di file su stdout (errata)

- Dato un file contenente nome, cognome, anno di nascita di un elenco di persone, leggerlo, spamparlo e calcolare l'età media

File `Elenco.txt`:
Andrea Marin 36 Marco Rossi 42

- Attenzione: il flag EOF viene impostato *dopo* il raggiungimento della fine del file

Errore frequente

```c
#include<stdio.h>

int main() {
    FILE *pf;
    char nome[100];
    char cognome[100];
    int eta;
    int letture;
    int totale = 0;

    pf = fopen("Elenco.txt","r");

    if (pf) {
        letture = 0;
        while (!feof(pf)) {
            fscanf(pf, "%s %s %d", nome, cognome, &eta);
            letture++;
            totale += eta;
            printf("%s, %s, %d\n", nome, cognome, eta);
        }
        fclose(pf);
        printf("Eta' media: %f\n", totale*1.0/letture);
    }
    else
        printf("Errore in lettura \n");

    return 0;
}
```

Soluzione corretta

```
#include<stdio.h>

int main() {
    FILE *pf;
    char nome[100];
    char cognome[100];
    int eta;
    int letture;
    int totale = 0;

    pf = fopen("Elenco.txt","r");

    if (pf) {
        letture = 0;
        while (fscanf(pf, "%s %s %d", nome, cognome, &eta) && !feof(pf)) {
            letture++;
            totale += eta;
            printf("%s, %s, %d\n", nome, cognome, eta);
        }
        fclose(pf);
        printf("Eta' media: %f\n", totale*1.0/letture);
    }
    else
        printf("Errore in lettura \n");

    return 0;
}
```