Laboratorio di
Algoritmi e Strutture Dati

Le Liste

Prof. Luigi Lamberti
2005

Premessa: i Vettori

L'uso degli **Array** monodimensionali, o **Vettori**, è il metodo più semplice per la memorizzazione di elenchi di elementi omogenei, pur presentando alcuni problemi:

Vantaggi

- ✓ L'ordinamento **logico** degli elementi corrisponde all'ordinamento **fisico** degli indirizzi.
- ✓ Ogni elemento occupa solo la memoria corrispondente alla sua quantità di informazione, senza spazi accessori.
- ✓ È possibile l'**accesso diretto** ad ogni elemento dell'insieme.
- ✓ Se l'insieme è ordinato, è possibile effettuare una ricerca **dicotomica**.

Svantaggi

- ✓ Occorre riservare preliminarmente uno spazio pari alla massima estensione prevedibile del vettore, causando uno spreco di memoria.
- ✓ L'inserimento o l'eliminazione di elementi richiede lo slittamento di tutta la parte rimanente dell'array.
- ✓ È possibile rappresentare solo organizzazioni lineari dei dati.

È dunque necessario ricorrere a un metodo che permetta di assegnare alle celle un ordinamento logico arbitrario, differente dal loro ordinamento fisico, migliorando l'efficienza delle operazioni di manutenzione dei dati.

Liste

Una **Lista** è una successione di elementi che occupano in memoria posizioni qualsiasi.
Ciascun elemento $\mathbf{H_i}$ è legato all'elemento successivo $\mathbf{H_{i+1}}$ tramite un puntatore contenente l'indirizzo di $\mathbf{H_{i+1}}$.
Ogni elemento, o Nodo, $\mathbf{H_i}$ della lista sarà formato da due campi:

- ✓ un campo **INFO** contenente tutte le informazioni legate al nodo;
- ✓ un campo **PUNT** contenente l'indirizzo del nodo successivo.

La chiave d'acccsso alla Lista, **puntatore di Testa**, è una variabile che contiene l'indirizzo del primo elemento.
L'ultimo elemento della lista riporta una marca speciale nel campo PUNT, detta **NIHIL** o **Ø**.

Vantaggi

- ✓ È possibile allocare solo lo spazio di memoria necessario, ad esempio nella Heap.
- ✓ L'inserimento o l'eliminazione di elementi richiede l'aggiornamento di due suli nodi.
- ✓ È possibile rappresentare organizzazioni non lineari dei dati.

Svantaggi

- ✓ L'unico modi di accedere ad un elemento è quello di scorrere tutta la lista.
- ✓ Ogni elemento occupa lo spazio supplementare per il PUNT; lo spreco è tanto più significatovo quanto più piccolo è il campo INFO.
- ✓ Anche se l'insieme è ordinato, non è possibile effettuare una ricerca **dicotomica** ma solo ricerche sequenziali.

Strutture

```
#define NIHIL 0

struct Frase { char           Lettere[100];
               struct Frase   *Punt;
             };

typedef struct Frase { char         Lettere[100];
                       struct Frase *Punt;
                     } FRASE;

int main (void)
{
  int   i, j;
  FRASE Righi[100];
  FRASE *Pt;

  for (i=0; i<3; i++)
    gets ( Righi[i].Lettere );

  Righi[0].Punt = Righi + 1;
  Righi[1].Punt = Righi + 2;
  Righi[2].Punt = NIHIL;

  Pt = Righi;
  while (Pt != NIHIL)
  { printf ("%s\n", Pt->Lettere);
    Pt = Pt->Punt;
  }
}
```