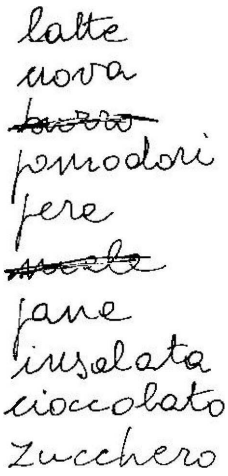
Liste concatenate

INTRODUZIONE

Nell'uso quotidiano il termine "lista" si riferisce ad una collezione lineare di dati. Nella figura (a) vediamo una lista della spesa: un primo elemento, un secondo... l'ultimo. Spesso capita di dover aggiungere o cancellare qualche articolo, come in figura (b), nella quale sono state aggiunte tre voci in fine lista *e* ne sono state cancellate altre due.

(a)



(b)

Nell'elaborazione dei dati capita spesso di dover immagazzinare in memoria ed elaborare dati organizzati in liste. Un modo di raccogliere questi dati è con gli array. Ricordiamo che la relazione lineare tra gli elementi di un array corrisponde alla relazione fisica tra dati in memoria, non alle informazioni contenute nei dati stessi. Questo rende facile calcolare l'indirizzo di un elemento in array. Però ci sono anche degli svantaggi: inserimento e cancellazione di elementi, per esempio, negli array sono piuttosto costosi. Visto poi che di solito un array occupa un blocco di spazio di memoria, quando serve altro spazio non è possibile semplicemente raddoppiare o triplicare la dimensione dell'array stesso. *(È per questo che agli array ci si riferisce col nome di liste, e vengono definiti strutture dati statiche.)*

Un altro modo di conservare in memoria una lista è far si che ogni elemento di questa contenga un campo detto *link* (catena) o *pointer* (indice) che contiene l'indirizzo dell'elemento successivo; non c'è così bisogno che elementi successivi della lista occupino spazi adiacenti in memoria, e risulta più facile inserirne o cancellarne. Se poi si è semplicemente interessati alla ricerca per inserire e cancellare dati, come nel caso dell'elaborazione dei testi (word processing) è proprio il caso di non conservare dati in un array ma in una lista coi pointer. Quest'ultimo tipo di struttura dati si chiama ***lista concatenata*** **("linked list*")***.*

LISTE CONCATENATE

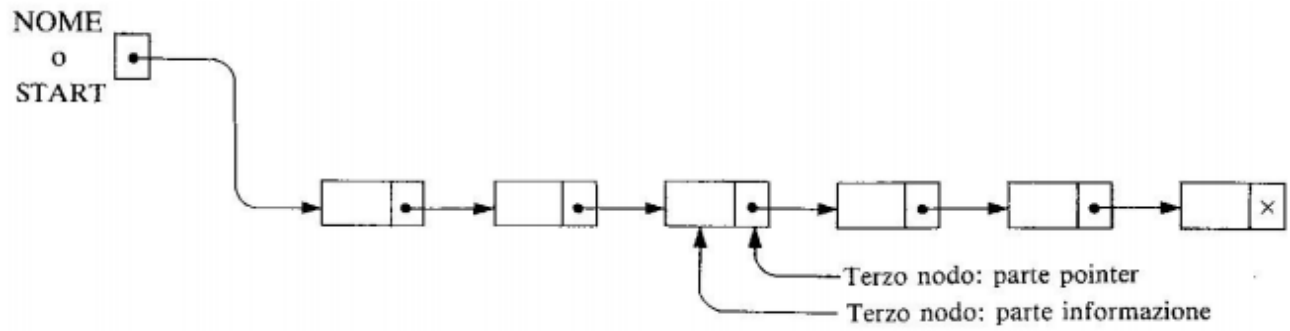
Lista *concatenata,* o *linked,* o *a senso unico è* quella collezione lineare di data element, detti ***nodi****,* in cui l'ordine è assegnato per mezzo di indici ("pointer").
Ogni nodo dunque è diviso in due parti:

1. la prima contiene **l'informazione** portata dall'elemento;
2. la seconda, detta *campo di* ***link*** o di pointer successivo, contiene l'indirizzo del successivo pointer della lista.

In figura abbiamo un diagramma schematico di "linked list" a 6 nodi. Ogni nodo si compone di due parti:

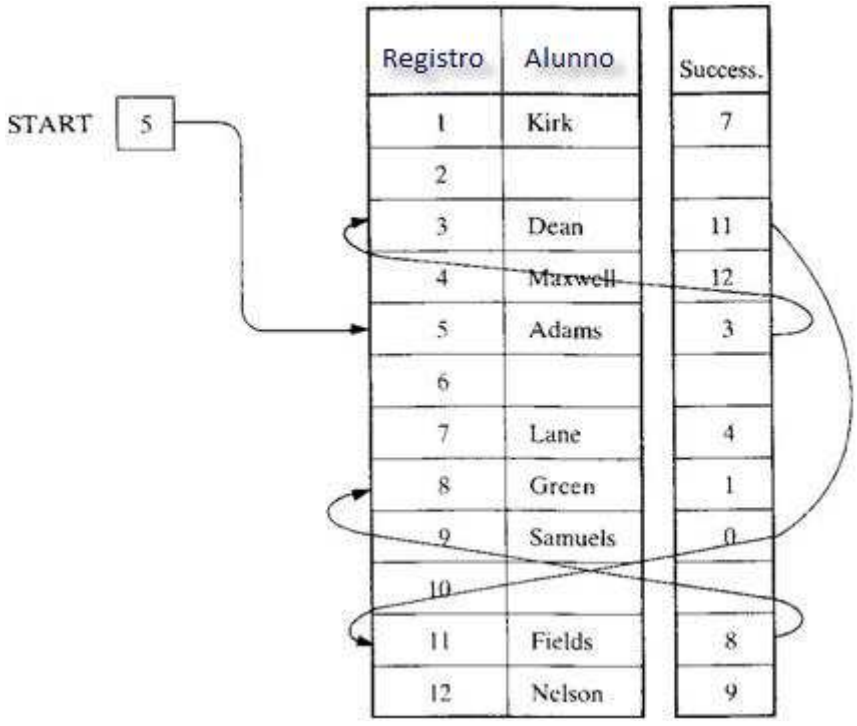
1. quella di sinistra è la parte informazione e può contenere un record intero di data item (per es. NOME, INDIRIZZO, ...)
2. quella di destra è il campo di pointer successivo del nodo, collegato con una freccia a quello successivo della lista, secondo l'uso per cui si collega il campo al nodo quando l'indirizzo di quest'ultimo compare nel campo assegnato.

Il pointer dell'ultimo nodo contiene un valore speciale, detto **pointer zero o *null pointer*:** un qualsiasi indirizzo non valido. (Nella pratica si usa lo O o un numero negativo.) Il pointer zero è indicato nel diagramma con una x e indica la fine della lista; questa contiene anche una *variabile pointer di lista,* chiamata **START**.



Lista concatenata a sei nodi.

Esempio:



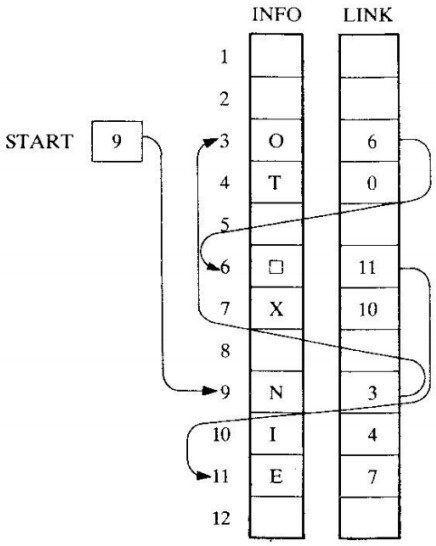
START contiene l'indirizzo del primo nodo; c'è quindi una freccia da START al primo nodo. Basta evidentemente questo indirizzo in START per risalire la lista. Nel caso particolare in cui questa non ha nodi, si parla di *lista vuota,* o *nulla,* e nella variabile START troveremo il pointer zero.

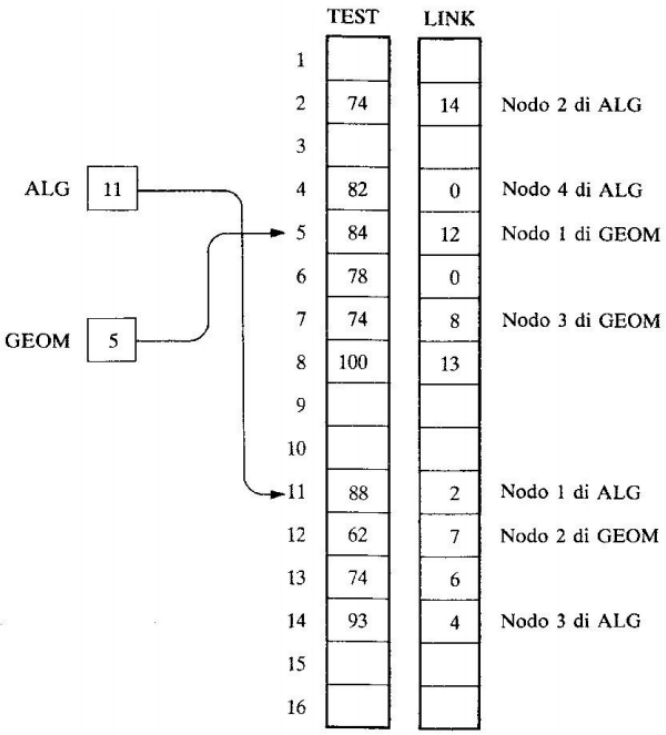
RAPPRESENTAZIONE IN MEMORIA DI LISTE CONCATENATE

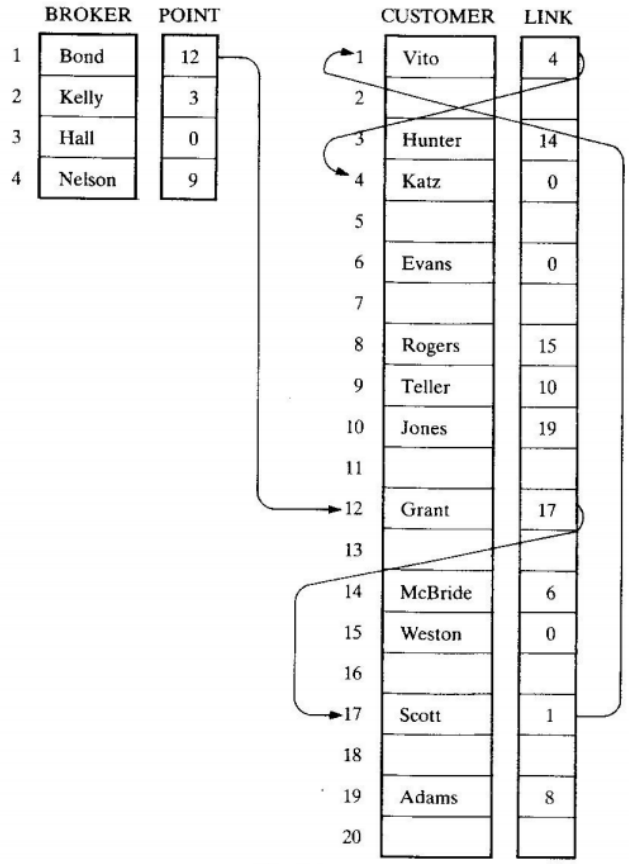
Sia LIST una lista concatenata: essa sarà mantenuta in memoria, se non diversamente implicito o specificato, con due array paralleli INFO e LINK e una variabile pointer START.
Prima di tutto ogni nodo N di LIST corrisponde a un indice intero K tale che:

1. INFO[K] contiene i dati al nodo N.
2. LINK[K] contiene la locazione del nodo che segue N.

START conterrà poi la locazione del primo nodo di LIST.
Inoltre il campo LINK dell'ultimo nodo di LIST conterrà un valore sentinella, indicato con NULL, a indicare la fine di LIST, oppure, se LIST è vuota, sarà START a contenere NULL. (Visto che gli indici degli array paralleli INFO e LINK saranno di solito positivi sceglieremo, a meno di indicazione contraria, NULL = O.)
Gli esempi di liste concatenate che seguono indicano che i nodi di una lista non debbono necessariamente occupare elementi adiacenti negli array INFO e LINK, e che negli stessi array INFO e LINK è possibile mantenere più di una lista. Ciascuna però deve avere la sua variabile pointer con la locazione del primo nodo.






BROKER
POINT
CUSTOMER
LINK
1 Bond 12
2 Kelly 3
3 Hall 0
4 Nelson 9
1 Vito 4
2
3 Hunter 14
4 Katz 0
5
6 Evans 0
7
8 Rogers 15
9 Teller 10
10 Jones 19
11
12 Grant 17
13
14 McBride 6
15 Weston 0
16
17 Scott 1
18
19 Adams 8
20