Corso di Laurea in INFORMATICA

Algoritmi e Strutture Dati

MODULO N. 3

Strutture lineari di dati: Liste

Il concetto di sequenza. Le Liste: specifiche e realizzazioni attraverso rappresentazioni sequenziali e collegate .

Questi lucidi sono stati preparati da Oronzo Altamura, Maria Francesca Costabile e Floriana Esposito, Università degli Studi di Bari, per uso didattico. Essi contengono materiale originale di proprietà dell'Università degli Studi di Bari e/o figure di proprietà di altri autori, società e organizzazioni di cui e' riportato il riferimento. Tutto o parte del materiale può essere fotocopiato per uso personale o didattico ma non può essere distribuito per uso commerciale. Qualunque altro uso richiede una specifica autorizzazione da parte dell'Università degli Studi di Bari e degli altri autori coinvolti.

CONSIDERIAMO STRUTTURE DI DATI CHE SI SVILUPPANO IN UNA DIMENSIONE E POSSONO ESSERE CONSIDERATE COME SEQUENZE DI OGGETTI CIOE'

LE STRUTTURE LINEARI DI DATI

IN ESSE E' IMPORTANTE L'ESISTENZA DI UNA RELAZIONE D'ORDINE TRA GLI OGGETTI, CHE CI AIUTA AD INDIVIDUARLI E A SELEZIONARLI.

PER DISTINGUERE LE DIVERSE STRUTTURE E' RILEVANTE CONSIDERARE:

a) I MODI DI INDIVIDUARE LA POSIZIONE IN CUI OPERARE NELLA SEQUENZA (MODI DI ACCEDERE)

b) I MODI DI AGIRE NELLA POSIZIONE INDIVIDUATA

a) MODI DI ACCEDERE

- ACCESSO DIRETTO
- ACCESSO PER SCANSIONE
- ACCESSO AGLI ESTREMI

IL VETTORE (ARRAY) E' IL TIPICO ESEMPIO DI ACCESSO DIRETTO: CONSENTE DI ACCEDERE AL SINGOLO COMPONENTE MEDIANTE IL NOME E IL MECCANISMO DELL'INDICE.

LA LISTA E' UN ESEMPIO DI ACCESSO PER SCANSIONE: CONSENTE L'ACCESSO ALL'ELEMENTO GENERICO SOLO DOPO AVER SCANDITO GLI ELEMENTI CHE LO PRECEDONO.

PILA E CODA SONO ESEMPI DI ACCESSO AGLI ESTREMI:

PILA (LIFO LAST IN FIRST OUT)

CODA (FIFO FIRST IN FIRST OUT)

b) RELATIVAMENTE AI MODI DI AGIRE NELLE POSIZIONI POSSIAMO AVERE I SEGUENTI TIPI DI OPERAZIONE:

- LETTURA DEL VALORE DI UN COMPONENTE
- AGGIORNAMENTO DEL VALORE DI UN COMPONENTE (CAMBIO DI VALORE)
- INSERIMENTO DI UN NUOVO COMPONENTE
- RIMOZIONE DI UN COMPONENTE (CANCELLAZIONE)

SONO POSSIBILI DIVERSE COMBINAZIONI TRA I MODI DI ACCEDERE E I MODI D'OPERARE. L'UNICO PROBLEMA RIGUARDA LA COMBINAZIONE TRA *ACCESSO DIRETTO* A TUTTA LA STRUTTURA E POSSIBILITÀ DI INSERZIONE E RIMOZIONE DI COMPONENTI .

GENERALMENTE QUESTE OPERAZIONI ALTERANO LA NUMERAZIONE DEI COMPONENTI E CIÒ È IN CONTRASTO COL PRINCIPIO CHE I COMPONENTI SIANO INDIVIDUATI DAL NUMERO D'ORDINE DELLA LORO POSIZIONE.

TUTTE LE STRUTTURE LINEARI, TRANNE IL VETTORE, SONO DOTATE DI OPERATORI DI INSERZIONE E RIMOZIONE DI COMPONENTI.

LISTE

UNA LISTA E' UNA SEQUENZA FINITA, ANCHE VUOTA, DI ELEMENTI DELLO STESSO TIPO.

LA DIFFERENZA COL CONCETTO DI "INSIEME" E' CHE MENTRE IN UN INSIEME UN ELEMENTO NON PUO' COMPARIRE PIU' DI UNA VOLTA, NELLA LISTA UNO STESSO ELEMENTO PUO' COMPARIRE PIU' VOLTE, IN POSIZIONI DIVERSE.

GLI ELEMENTI DELLA LISTA SONO DEFINITI ATOMI O NODI, CUI SONO ASSOCIATE DELLE INFORMAZIONI.

INDICHIAMO LA LISTA CON LA NOTAZIONE

$$L = < a_1, a_2, \ldots, a_n >$$

A CIASCUN ELEMENTO DI UNA LISTA VIENE ASSOCIATA

UNA POSIZIONE pos(i)

E UN VALORE a(i)

L'ACCESSO

- SI PUO' ACCEDERE **DIRETTAMENTE** SOLO AL **PRIMO** ELEMENTO DELLA SEQUENZA.
- PER ACCEDERE AL **GENERICO** ELEMENTO OCCORRE **SCANDIRE SEQUENZIALMENTE** GLI ELEMENTI DELLA LISTA CHE LO PRECEDONO

LE OPERAZIONI

- E' POSSIBILE AGGIUNGERE **(INSERIRE)** E TOGLIERE **(CANCELLARE)** ELEMENTI.
- *POICHE' LA LISTA E' A **DIMENSIONE VARIABILE** QUESTE OPERAZIONI CHE ALTERANO LA DIMENSIONE SONO FONDAMENTALI (NEL VETTORE NON SONO CONCESSE).*

LA LISTA E' DUNQUE UNA STRUTTURA DATI DINAMICA.

LA **LUNGHEZZA** DI UNA LISTA E' IL NUMERO DEI SUOI ELEMENTI. SE IL NUMERO DI ELEMENTI E' ZERO LA LISTA SI DICE **VUOTA**.

LA LUNGHEZZA CONTA LE POSIZIONI, NON I SIMBOLI DISTINTI, COSI' UN SIMBOLO CHE COMPARE K VOLTE CONTRIBUISCE CON K UNITA' ALLA LUNGHEZZA DELLA LISTA.

SE $L = < a_1, a_2, \dots, a_n >$ E' UNA LISTA

ALLORA PER OGNI i E j TALI CHE

$1 \le i \le j \le n$

$< a_i, a_{i+1}, \dots, a_j >$ E' UNA SOTTOLISTA DI L, OTTENUTA PARTENDO DALLA POSIZIONE i E PRENDENDO TUTTI GLI ELEMENTI FINO ALLA POSIZIONE j.

LA LISTA VUOTA $< >$ E' SOTTOLISTA DI QUALSIASI LISTA.

LA SPECIFICA SINTATTICA

TIPI: LISTA, POSIZIONE, BOOLEAN, TIPOELEM

OPERATORI:

CREALISTA : ( ) → LISTA

LISTAVUOTA : (LISTA) → BOOLEAN

LEGGILISTA : (POSIZIONE, LISTA) → TIPOELEM

SCRIVILISTA : (TIPOELEM,POSIZIONE,LISTA) → LISTA

PRIMOLISTA : (LISTA) → POSIZIONE

FINELISTA : (POSIZIONE, LISTA) → BOOLEAN

SUCCLISTA : (POSIZIONE, LISTA) → POSIZIONE

PREDLISTA : (POSIZIONE, LISTA) → POSIZIONE

INSLISTA : (TIPOELEM,POSIZIONE,LISTA) → LISTA

CANCLISTA : (POSIZIONE, LISTA) → LISTA

SPECIFICA SEMANTICA

TIPI:

LISTA: INSIEME DELLE SEQUENZE $L = < a_1, a_2, \dots, a_n >$ DI ELEMENTI DI TIPO TIPOELEM DOVE L'ELEMENTO i-ESIMO HA VALORE a_i E POSIZIONE pos(i)

BOOLEAN: INSIEME DEI VALORI DI VERITA'

OPERATORI:

CREALISTA = L'

POST: L' = < >

LISTAVUOTA(L) = b

POST: b = TRUE SE L= < >

b = FALSE ALTRIMENTI

LEGGILISTA(p,L) = a

PRE: $L = < a_1, a_2, \dots, a_n >$ $p = pos(i)$ $1 \le i \le n$

POST: $a = a_i$

SCRIVILISTA(a,p,L) = L'

PRE: $L = <a_1, a_2, \dots, a_n >$; $p = pos(i)$ $1 \le i \le n$

POST: $L' = < a_1, a_2, \dots, a_{i-1}, a, a_{i+1}, \dots, a_n >$

PRIMOLISTA(L) = p

POST: p = pos(1)

FINELISTA(p,L) = b

PRE:$L = <a_1, a_2, \dots, a_n >$;p = pos(i) $1 \le i \le n$

POST: b= FALSE SE p =pos(i) $1 \le i \le n$

b = TRUE ALTRIMENTI

SUCCLISTA(p,L) = q

PRE: $L = <a_1, a_2, \dots, a_n >$; $p = pos(i)$ $1 \le i < n$

POST: q = pos(i+1)

PREDLISTA(p,L) = q

PRE: $L = <a_1, a_2, \dots, a_n >$; $p = pos(i)$ $2 \le i \le n$

POST: q = pos(i-1)

INSLISTA(a,p,L) = L'

PRE: $L = <a_1, a_2, \dots, a_n >$; $p = pos(i)$ $1 \le i \le n+1$

POST: $L' = < a_1, a_2, \dots, a_{i-1}, a, a_i, a_{i+1}, \dots, a_n >$, se $1 \le i \le n$

$L' = <a_1, a_2, \dots, a_n, a >$, se i = n+1

(e quindi $L' = < a >$ se $i = 1$ e $L = < >$)

CANCLISTA(p,L) = L'

PRE: $L = <a_1, a_2, \dots, a_n >$; $p = pos(i)$ $1 \le i \le n$

POST: $L' = < a_1, a_2, \dots, a_{i-1}, a_{i+1}, \dots, a_n >$

DALLA SPECIFICA SEMANTICA EMERGE CHE **PER ACCEDERE A UN ELEMENTO OCCORRE CONOSCERNE LA POSIZIONE.**

QUESTO NELLE LISTE **E' POSSIBILE SOLO PER IL PRIMO ELEMENTO.** L'UNICO OPERATORE CHE DA' PER RISULTATO LA POSIZIONE (SENZA ALTRE INFORMAZIONI) E' **PRIMOLISTA.**

PER GLI ALTRI LA POSIZIONE SI OTTIENE CONOSCENDO A PRIORI LA POSIZIONE DELL'ELEMENTO PRECEDENTE (O SEGUENTE) E APPLICANDO L'OPERAZIONE SUCCLISTA (O PREDLISTA). DUNQUE, PER ACCEDERE AD UN GENERICO ELEMENTO OCCORRE

SCANDIRE LA LISTA A PARTIRE DAL PRIMO ELEMENTO

L'OPERATORE LISTAVUOTA E' RIDONDANTE VISTO CHE PUO' ESSERE SOSTITUITO DA

FINELISTA (PRIMOLISTA (L) , L)

NOTA: *IN ALCUNI TESTI SI TROVA UNA DIVERSA SPECIFICA DI PRIMOLISTA(L) CHE RESTITUISCE L'ELEMENTO PRIMO PIUTTOSTO CHE LA POSIZIONE. QUESTO CORRISPONDE ALLA SPECIFICA (LISTA) ⟶ TIPOELEM*

UN ESEMPIO DI COME SIA POSSIBILE, AVENDO A DISPOSIZIONE GLI OPERATORI DEFINITI NELLA ALGEBRA PER IL TRATTAMENTO DI UNA LISTA, RISOLVERE UN GENERICO PROBLEMA, EVITANDO DI ENTRARE NEI DETTAGLI RELATIVI A COME LA LISTA SI E' IMPLEMENTATA, E' DATO DI SEGUITO.

SUPPONIAMO DI UTILIZZARE COME LINGUAGGIO ALGORITMICO DI RIFERIMENTO IL PASCAL E SUPPONIAMO DI DISPORRE, IN PASCAL, DEGLI OPERATORI SUL TIPO LISTA.

CI PONIAMO IL SEGUENTE PROBLEMA

ELIMINAZIONE DI DUPLICATI:

DATA UNA LISTA L, I CUI ELEMENTI SIANO INTERI, ELIMINARE DA L GLI ELEMENTI CHE SONO DUPLICATI.

```
procedure EPURAZIONE (var L : lista);
var p,q : posizione;
begin
        p:=PRIMOLISTA(L);
        while not FINELISTA(p,L) do
        begin
                q:=SUCCLISTA(p,L);
                while not FINELISTA(q,L) do
                if LEGGILISTA(p,L) = LEGGILISTA(q,L)
                then CANCLISTA(q,L)
                else q:=SUCCLISTA(q,L);
                p:=SUCCLISTA(p,L);
        end;
end;
```

COME E' EVIDENTE SI E' POTUTO FARE RIFERIMENTO ALLA STRUTTURA LISTA SENZA ENTRARE NEI DETTAGLI REALIZZATIVI.

PER REALIZZARE UNA LISTA **DISTINGUIAMO DUE RAPPRESENTAZIONI FONDAMENTALI:** RAPPRESENTAZIONE SEQUENZIALE E RAPPRESENTAZIONE COLLEGATA.

REALIZZAZIONE SEQUENZIALE CON VETTORE

UNA LISTA PUO' ESSERE RAPPRESENTATA USANDO UN VETTORE (ARRAY MONODIMENSIONALE).

POICHE' IL NUMERO DI ELEMENTI CHE COMPONGONO LA LISTA PUO' VARIARE, SI UTILIZZA UNA VARIABILE **PRIMO** PER IL VALORE DELL'INDICE DELLA COMPONENTE DEL VETTORE IN CUI E' MEMORIZZATO IL PRIMO ELEMENTO DELLA LISTA.

SI UTILIZZA UN'ALTRA VARIABILE **LUNGHEZZA** PER INDICARE IL NUMERO DI ELEMENTI DI CUI E' COMPOSTA LA LISTA RAPPRESENTATA.

LISTA

(4 5 1 21 45)

1	4
2	5
3	1
4	21
5	45
6	78
7	12
8	1
9	-5
10	0
11	-2
12	61

PRIMO [1]

LUNGHEZZA [5]

QUESTA RAPPRESENTAZIONE CONSENTE DI REALIZZARE MOLTO SEMPLICEMENTE ALCUNE DELLE OPERAZIONI DEFINITE PER LA LISTA.

IL VERO PROBLEMA RIGUARDA L'INSERZIONE E LA RIMOZIONE DI ELEMENTI

LISTA

(4 5 1 11 21 45)

1	4
2	5
3	1
4	11
5	21
6	45
7	12
8	1
9	-5
10	0
11	-2
12	61

PRIMO [1]

LUNGHEZZA [6]

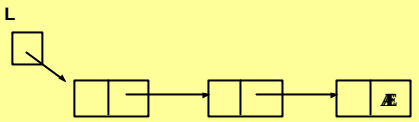
INFATTI, LA BANALE INSERZIONE IN QUARTA POSIZIONE DI UN NUOVO ELEMENTO 11 CAUSA LO SPOSTAMENTO VERSO IL BASSO DEL 4° E DEL 5° ELEMENTO DELL'ARRAY

LA RAPPRESENTAZIONE COLLEGATA

L'IDEA FONDAMENTALE DELLA RAPPRESENTAZIONE COLLEGATA DI UNA LISTA E' QUELLA DI MEMORIZZARE I SUOI ELEMENTI ASSOCIANDO AD OGNUNO DI ESSI UNA PARTICOLARE INFORMAZIONE (RIFERIMENTO) CHE PERMETTA DI INDIVIDUARE LA LOCAZIONE IN CUI E' MEMORIZZATO L'ELEMENTO SUCCESSIVO.

PER VISUALIZZARE TALE RAPPRESENTAZIONE SI USA UNA NOTAZIONE GRAFICA IN CUI

- GLI ELEMENTI SONO RAPPRESENTATI MEDIANTE NODI
- I RIFERIMENTI MEDIANTE ARCHI CHE COLLEGANO NODI



SI PUO' NOTARE CHE SI USA UN RIFERIMENTO AL PRIMO ELEMENTO DELLA LISTA (RIFERIMENTO INIZIALE) E UN SIMBOLO SPECIALE Æ COME RIFERIMENTO ASSOCIATO ALL'ULTIMO NODO (NEL CASO LA LISTA SIA VUOTA, TALE SIMBOLO COMPARE DIRETTAMENTE NEL RIFERIMENTO INIZIALE).



REALIZZAZIONE CON CURSORI

E' SEMPRE UTILIZZATO UN VETTORE (ARRAY MONODIMENSIONALE) PER L'IMPLEMENTAZIONE DELLA LISTA, MA SI RIESCE A SUPERARE, ATTRAVERSO I RIFERIMENTI, IL PROBLEMA DELL'AGGIORNAMENTO (INSERIMENTO O CANCELLAZIONE DI UN ELEMENTO)

SI REALIZZANO I RIFERIMENTI MEDIANTE CURSORI, CIOE' VARIABILI INTERE O ENUMERATIVE, IL CUI VALORE E' INTERPRETATO COME INDICE DI UN VETTORE.

SI DEFINISCE UN VETTORE SPAZIO CHE :

1) CONTIENE TUTTE LE LISTE, OGNUNA INDIVIDUATA DA UN PROPRIO CURSORE INIZIALE
2) CONTIENE TUTTE LE CELLE LIBERE, ORGANIZZATE IN UNA LISTA, DETTA "LISTALIBERA"

Esempio: Disponiamo di tre diverse liste L, M, S

L = 7, 2 M = 4, 13 S = 13, 4, 9

POSSIAMO USARE UN UNICO VETTORE SPAZIO PER RAPPRESENTARE LE TRE LISTE.

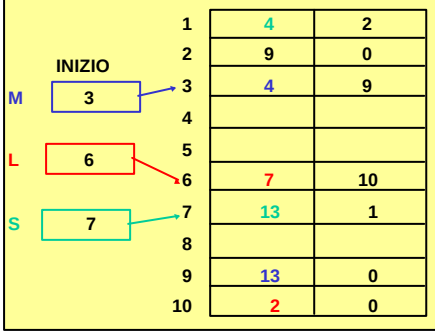
LA COMPONENTE DI SPAZIO HA DUE CAMPI: NEL CAMPO "ELEMENTO" E' MEMORIZZATO IL CONTENUTO DEL NODO, NEL CAMPO "SUCCESSIVO" E' IL RIFERIMENTO AL PROSSIMO NODO.

LA SEQUENZA DEGLI ELEMENTI CHE FORMANO LA LISTA E' RICOSTRUIBILE INIZIANDO DALLA COMPONENTE DELL'ARRAY CORRISPONDENTE AL VALORE DI INIZIO, NEL CUI CAMPO "ELEMENTO" E' MEMORIZZATO IL VALORE DEL PRIMO ELEMENTO. SEGUENDO I CURSORI SI TROVANO GLI ELEMENTI SUCCESSIVI DELLA LISTA.

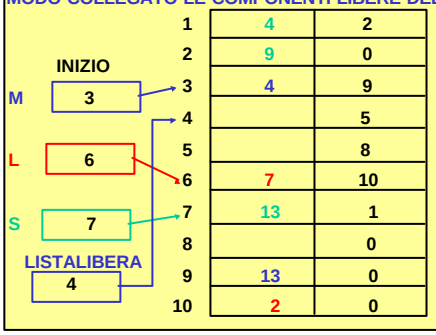
INFATTI, UTILIZZANDO I CURSORI, SI PUO' DEFINIRE LA POSIZIONE pos(i) DELL'ELEMENTO i-ESIMO DI L UGUALE AL VALORE DEL CURSORE ALLA CELLA DEL VETTORE SPAZIO CHE CONTIENE L'ELEMENTO (i-1) – ESIMO, SE 2 £ i £ n.

ANALOGAMENTE, SI DEFINISCE pos(n+1) UGUALE AL CURSORE ALLA CELLA DI SPAZIO CHE CONTIENE L'ELEMENTO n – ESIMO SE n ³ 1, O UGUALE A 0 ALTRIMENTI. LA LISTA VUOTA E' L=0.

ESEMPIO: L = 7, 2 M = 4, 13 S = 13, 4, 9



PER POTER AGGIORNARE UNA LISTA COSI' REALIZZATA SORGE IL PROBLEMA DI INDIVIDUARE LA POSIZIONE DI UNA COMPONENTE LIBERA NELL'ARRAY. SI USA UNA LISTALIBERA MEMORIZZATA NELLO STESSO ARRAY PER RACCOGLIERE IN MODO COLLEGATO LE COMPONENTI LIBERE DELL'ARRAY.



QUANDO SI DEVE INSERIRE UN NUOVO ELEMENTO NELLA LISTA, SI PRELEVA UNA COMPONENTE DELLA LISTALIBERA E LA SI UTILIZZA PER MEMORIZZARE IL NUOVO ELEMENTO, COLLEGANDOLO IN MODO OPPORTUNO AGLI ALTRI ELEMENTI DELLA LISTA.

UNA POSSIBILE DICHIARATIVA PASCAL:

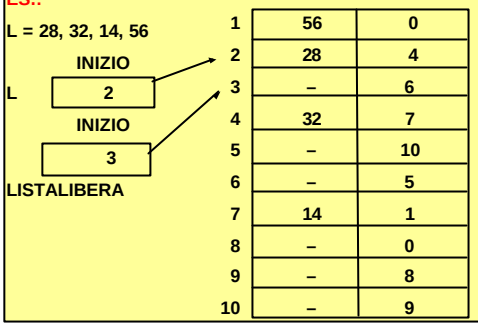
```
CONST
        maxlung = ... ; (constante intera positiva)

TYPE
        posizione = 0..maxlung;
        Lista = posizione;
        tipoelem = integer;
        componente =        RECORD
                            elemento:tipoelem;
                            successivo:posizione;
                            END;
VAR
        listalibera : lista;
        SPAZIO; ARRAY (1..maxlung) OF componente;
```

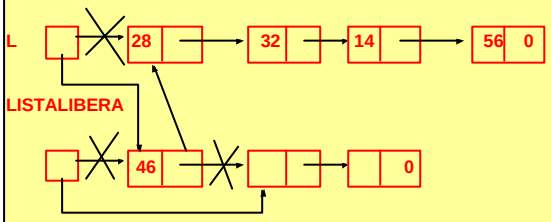
LA LISTALIBERA RAPPRESENTA UN SERBATOIO DA CUI PRELEVARE COMPONENTI LIBERE DELL'ARRAY E IN CUI RIVERSARE LE COMPONENTI DELL'ARRAY CHE NON SONO PIU' UTILIZZATE PER LA LISTA

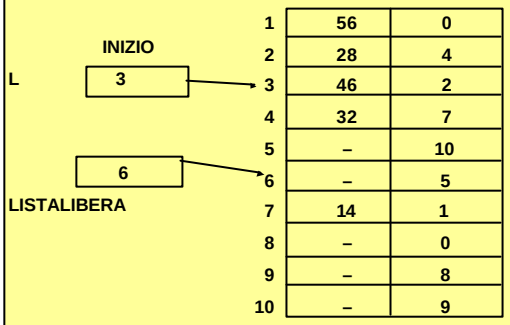
ES.:

L = 28, 32, 14, 56



SE SI VUOLE INSERIRE UN NUOVO ELEMENTO (= 46) IN TESTA ALLA LISTA SI FA USO DELLA PRIMA COMPONENTE DELLA LISTA LIBERA





LE OPERAZIONI ESEGUIBILI SULLA LISTALIBERA SONO:

1. LA INIZIALIZZAZIONE, CHE SERVE A COLLEGARE TRA LORO LE VARIE COMPONENTI DELL'ARRAY CHE FANNO PARTE INIZIALMENTE DELLA LISTALIBERA;
2. IL PRELIEVO DI UNA COMPONENTE DELLA LISTALIBERA SE ESISTE;
3. LA RESTITUZIONE DI UNA COMPONENTE ALLA LISTALIBERA.

```
PROCEDURE INIZIALIZZA_LISTA_LIBERA;
VAR i:integer;
BEGIN
listalibera:=1;
FOR i:=1 TO maxlung – 1 DO
        BEGIN
                SPAZIO[ i ].successivo:=i+1;
        END;
        SPAZIO [ maxlung ].successivo:=0;
END;
```

E' IMMEDIATO VERIFICARE, PRIMA DI DEFINIRE LE OPERAZIONI DI INSLISTA E CANCLISTA, CHE CON QUESTA RAPPRESENTAZIONE L'INSERIMENTO E L'ELIMINAZIONE DI UN ELEMENTO NON RICHIEDONO LO SPOSTAMENTO DI ALTRI ELEMENTI DELLA LISTA, GRAZIE ALLA LISTALIBERA.

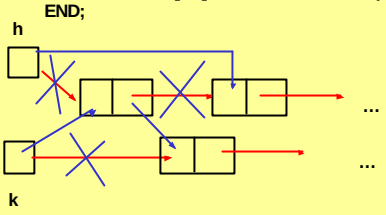
RIMANGONO I PROBLEMI CONNESSI ALL'USO DELL'ARRAY CIOE' ALL'ESIGENZA DI DEFINIRE UNA DIMENSIONE.

CONSIDERAZIONI SULLA REALIZZAZIONE DI UNA RAPPRESENTAZIONE COLLEGATA MEDIANTE ARRAY

- UNO DEGLI SVANTAGGI DELLA RAPPRESENTAZIONE SEQUENZIALE E' SUPERATO: L'INSERIMENTO E LA ELIMINAZIONE DI UN ELEMENTO NON RICHIEDONO LO SPOSTAMENTO DI ALTRI ELEMENTI NELLA LISTA
- LA COMPLICAZIONE DATA DALLA NECESSITA' DI GESTIRE LA LISTA LIBERA E' COMPENSATA DAL FATTO CHE GLI AGGIORNAMENTI RICHIEDONO UN NUMERO LIMITATO DI OPERAZIONI
- RIMANGONO GLI SVANTAGGI CONNESSI ALL'USO DELL'ARRAY: LA DIMENSIONE DELL'ARRAY RAPPRESENTA UN LIMITE ALLA CRESCITA DELLA LISTA E LA QUANTITA' IN MEMORIA UTILIZZATA NON DIPENDE DALLA LUNGHEZZA EFFETTIVA DELLA LISTA
- RISPETTO ALLA RAPPRESENTAZIONE SEQUENZIALE VI E' UN'ULTERIORE OCCUPAZIONE DI MEMORIA, VISTA LA NECESSITA' DI MEMORIZZARE I RIFERIMENTI.

E' FONDAMENTALE DISPORRE DELLA PROCEDURA SPOSTA(h,k) CHE TRASCFERISCE LA CELLA PUNTATA DA h SPOSTANDOLA PRIMA DELLA CELLA PUNTATA DA k

```
PROCEDURE SPOSTA (var h,k:posizione);
var temp : posizione;
BEGIN
        temp:=k;
        k:=h;
        h:=SPAZIO [ h ].SUCCESSIVO;
        SPAZIO [ k ].SUCCESSIVO:=temp;
END;
```




```
PROCEDURE INSLISTA (a: tipoelem; p:posizione; VAR L:lista);
BEGIN
        IF listalibera = 0 THEN WRITE ('ERRORE')
        ELSE IF p = 0 THEN
                BEGIN
                SPOSTA(listalibera,L);
                SPAZIO [ L ].elemento:=a;
                END
        ELSE
                BEGIN
                SPOSTA(listalibera,SPAZIO [ p ].successivo);
                SPAZIO [ SPAZIO [ p ].successivo ].elemento:=a;
                END;
END;
```

```
PROCEDURE CANCLISTA (p:posizione; VAR L:lista);
BEGIN
        IF p = 0 THEN
        SPOSTA(L,listalibera)
        ELSE
        SPOSTA(SPAZIO [ p ].successivo,listalibera);
        END;
```

NOTA:

BENCHE' SI SIANO DATE LE REALIZZAZIONI USANDO IL LINGUAGGIO PASCAL, E' EVIDENTE CHE LA REALIZZAZIONE CON CURSORI E' IMPLEMENTABILE IN TUTTI I LINGUAGGI CHE DISPONGONO DI VETTORI E CONSENTONO L'ASTRAZIONE FUNZIONALE MEDIANTE PROCEDURE O FUNZIONI (FORTRAN, ALGOL)

UN' ALTRA POSSIBILE REALIZZAZIONE DI UNA LISTA E' QUELLA MEDIANTE L'USO CONGIUNTO DEL TIPO PUNTATORE E DEL TIPO RECORD.

QUESTA E' SICURAMENTE LA REALIZZAZIONE PIU' EFFICACE DELLA RAPPRESENTAZIONE COLLEGATA.

IL TIPO PUNTATORE (in PASCAL)

E' UN TIPO DI DATO I CUI VALORI RAPPRESENTANO INDIRIZZI DI LOCAZIONI DI MEMORIA. LE OPERAZIONI USUALMENTE DISPONIBILI SU UNA VARIABILE P SONO:

- L'ACCESSO ALLA LOCAZIONE IL CUI INDIRIZZO E' MEMORIZZATO IN P
- LA RICHIESTA DI UNA NUOVA LOCAZIONE DI MEMORIA E LA MEMORIZZAZIONE DELL'INDIRIZZO IN P (new)
- IL RILASCIO DELLA LOCAZIONE DI MEMORIA IL CUI INDIRIZZO E' MEMORIZZATO IN P (dispose)

Esempi:

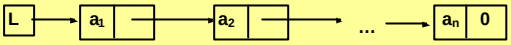
```
type punta–intero = ^integer;
var     x : integer;
        y : punta–intero;
```

x INDICA UN INTERO

y INDICA IL RIFERIMENTO AD UNA LOCAZIONE CHE CONTIENE UN INTERO (SE IN ESECUZIONE y = 1031, NELLA CELLA CON QUESTO INDIRIZZO VI SARA' UN INTERO; SE L'INTERO CONTENUTO E' 5 PER ACCEDERVI INDICHERO' y^ CHE DARA' 5)

RAPPRESENTAZIONE COLLEGATA REALIZZATA MEDIANTE PUNTATORI

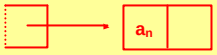
UNA POSSIBILE REALIZZAZIONE E' QUELLA DI UNA LISTA MONODIREZIONALE SEMPLIFICATA. VI E' UNA STRUTTURA DI n ELEMENTI O "CELLE", TALE CHE L'i – ESIMA CELLA CONTIENE L'i – ESIMO ELEMENTO DELLA LISTA E L'INDIRIZZO DELLA CELLA CHE CONTIENE L'ELEMENTO SUCCESSIVO



- LA PRIMA CELLA E' INDIRIZZATA DA UNA VARIABILE L DI TIPO PUNTATORE
- L'ULTIMA CELLA CONTIENE UN VALORE CONVENZIONALE NIL
- GLI INDIRIZZI SONO NOTI ALLA MACCHINA MA NON AL PROGRAMMATORE
- LA POSIZIONE pos(i) E' UGUALE AL VALORE DEL PUNTATORE ALLA CELLA CHE CONTIENE IL PUNTATORE ALL'i – ESIMO ELEMENTO (ATOMO) 2 £ i £ n

DUNQUE IN GENERALE

pos (n + 1)



PER n > 1

MENTRE PER n = 0



IN PASCAL NIL E' UN PARTICOLARE VALORE CHE SI PUO' ASSEGNARE AD UN PUNTATORE PER INDICARE CHE NON PUNTA A NIENTE

UNA POSSIBILE DICHIARAZIONE DI TIPO

```
TYPE
        cella = RECORD
                elemento : tipoelem;
                successivo : ^cella;
                END;
        lista = ^cella;
        posizione = ^cella;
```

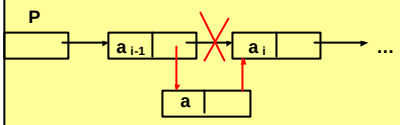
INOLTRE LA PRIMITIVA PASCAL

new(L)

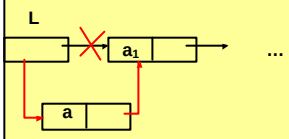
ALLOCA UNA VARIABILE DEL TIPO PUNTATO DA L, CIOE' UNA "CELLA", E FORNISCE IN L IL SUO INDIRIZZO.

L^ INDICA IL VALORE DELLA CELLA PUNTATA DA L.

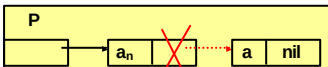
L'AGGIORNAMENTO DEI PUNTATORI



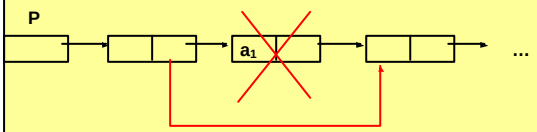
INSLISTA p = pos (i) i ³ 2 (a)



INSLISTA p = pos (1), n ³ 1 (b)



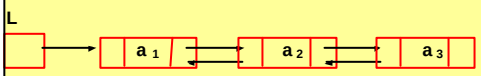
INSLISTA p = pos (n+1), n ³ 1 (c)



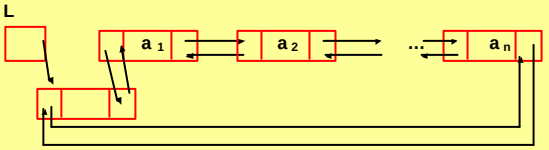
CANCLISTA p = pos (i), i ³ 2 (d)

UNA VARIANTE DELLA RAPPRESENTAZIONE COLLEGATA (A DOPPI PUNTATORI O SIMMETRICA)

OGNI ELEMENTO CONTIENE, OLTRE AL RIFERIMENTO AL NODO SUCCESSIVO, ANCHE IL RIFERIMENTO AL PRECEDENTE.



LA LISTA PUO' ESSERE "ESPANSA" CON UNA CELLA IN PIU' PER LA REALIZZAZIONE CIRCOLARE



CON QUESTA RAPPRESENTAZIONE SI HA IL VANTAGGIO DI:

- **POTER SCANDIRE LA LISTA IN ENTRAMBE LE DIREZIONI**
- **POTER INDIVIDUARE FACILMENTE L'ELEMENTO CHE PRECEDE**
- **POTER REALIZZARE LE OPERAZIONI DI INSERIMENTO SENZA DOVER USARE VARIABILI AGGIUNTIVE**

ESERCIZIO SU LISTE

ESERCIZIO SU LISTE

- Si vogliono gestire le prenotazioni dei testi in una biblioteca. Si dispone delle richieste dei testi da parte degli utenti e dell'elenco dei testi disponibili in biblioteca. Si desidera un programma che, sulla base delle richieste, verifichi l'esistenza del testo come disponibile al prestito e definisca l'elenco dei libri da prestare, completo del nome della persona cui prestarlo.

PER LA SOLUZIONE DEL PROBLEMA PROCEDERE SECONDO LE FASI SOTTOELENCATE, DOCUMENTANDOLE OPPORTUNAMENTE:

- ***ANALISI DEL PROBLEMA*: CHIARIRE LA TRACCIA RIGUARDO ALL'OBIETTIVO DA PERSEGUIRE, EVIDENZIANDO OUTPUT, INPUT, IPOTESI FATTE ED ERRORI;**
- ***DEFINIZIONE DEI DATI*: DEFINIRE LE STRUTTURE DEI DATI CHE SI UTILIZZANO CON LA RELATIVA ALGEBRA. PROPORRE UNA REALIZZAZIONE EFFICIENTE;**
- ***FORMALIZZAZIONE DELLA STRATEGIA DI SOLUZIONE*: DESCRIVERE GRAFICAMENTE O LINEARMENTE PRIMA L'ALGORITMO PRINCIPALE E DETTAGLIARE, POI, I SINGOLI BLOCCHI. DOCUMENTARE LA STRUTTURA DEL PROGRAMMA (LIVELLI DI INNESTO DELLA PROCEDURE);**
- ***CODIFICA IN UN LINGUAGGIO DI PROGRAMMAZIONE*.**

ESERCIZIO SU LISTA GENERALIZZATA

SI ORGANIZZINO I DATI RELATIVI AGLI INSEGNAMENTI DEL CL IN INFORMATICA IN UNA STRUTTURA DINAMICA. PER OGNI INSEGNAMENTO SI MEMORIZZINO NUMERO DI MATRICOLA, COGNOME E NOME DEGLI STUDENTI CHE SEGUONO L'INSEGNAMENTO. QUESTI, ESSENDO IN NUMERO VARIABILE, DEVONO ESSERE GESTITI DINAMICAMENTE.

PER OGNI INSEGNAMENTO STAMPARE:

- **IL NOME DELL'INSEGNAMENTO;**
- **IL NUMERO DEGLI STUDENTI CHE SEGUONO TALE INSEGNAMENTO;**
- **MATRICOLA, COGNOME E NOME DEGLI STUDENTI LA CUI MATRICOLA È MAGGIORE DI UN VALORE ASSEGNATO.**