Algoritmi e Strutture Dati

Università di Camerino

Corso di Laurea in Informatica

(12 CFU)

I periodo didattico

Emanuela Merelli
email:emanuela.merelli@unicam.it

Argomenti della lezione

- Tipi di Dato Astratto
  - Pila
  - Coda

- *Concetto di*
  - *Struttura dati dinamiche*

Strutture Dati Dinamiche

Una struttura dati si dice essere dinamica se permette di rappresentare insiemi dinamici, cioè insiemi la cui cardinalità varia durante l'esecuzione del programma.

Esamineremo Strutture dati dinamiche che utilizzano *puntatori*.

Puntatore

Permette di definire un tipo di dato Tp i cui valori sono puntatori (riferimenti) a valori di un altro tipo di dato T

Esempio:

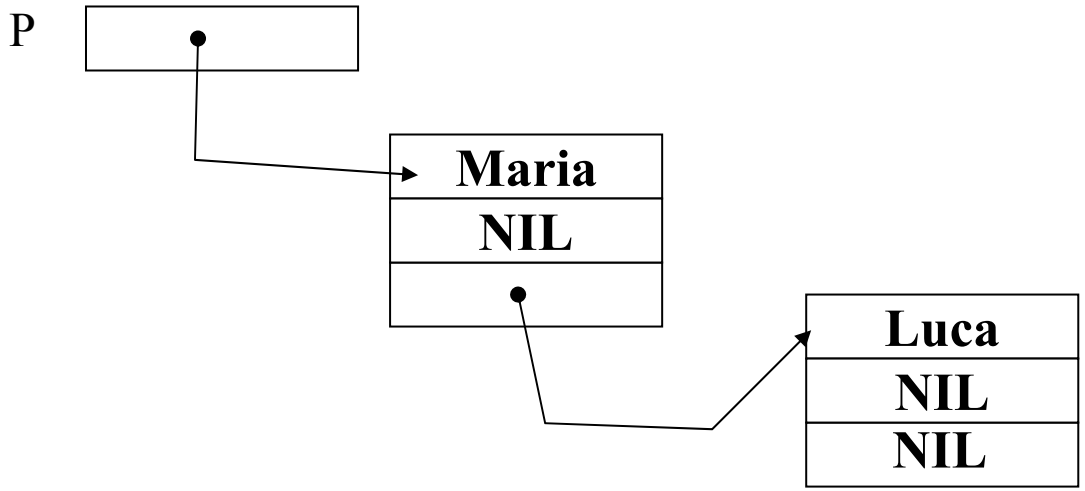
type Tp = ↑ T

Var P :Tp

Begin

New(P)

New(P↑.*next*)



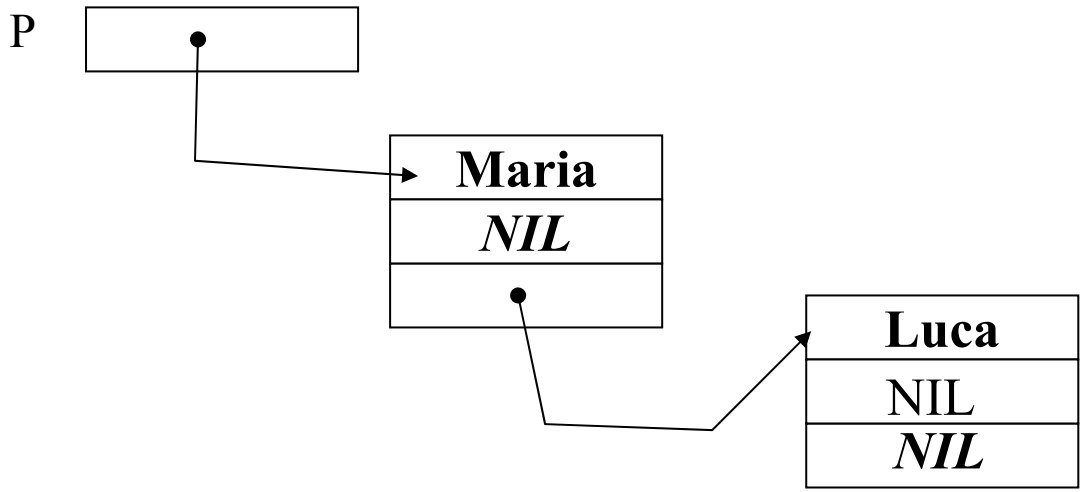
I puntatori

Distinguiamo:

1. Tipo del puntatore T_p definito tramite un altro tipo T
2. Variabile di tipo puntatore $P:T_p$
3. Variabile dinamica new(P)
    1. Non vengono dichiarate esplicitamente
    2. Non vengono identificate esplicitamente tramite un identificatore
    3. Vengono create dinamicamente in base alle necessità che emergono nel corso del programma
    4. Vengono referenziate tramite una variabile di tipo puntatore, che rappresenta indirettamente l'indirizzo della variabile in memoria
    5. La loro esistenza è indipenedente dal blocco in cui sono state create
    6. La dichiarazione di una variabile di tipo puntatore, all'atto della compilazione, non alloca memoria per l'oggetto di tipo T.
    7. Il comando New allocherà memoria per l'oggetto di tipo T
4. NIL è il valore che indica puntatore non definito valido per tutti i tipi di puntatori

Esempio

```
Type
  T = RECORD
            Nome : Alfa;
            Madre : Tp;
            Padre : Tp;
          END;
  Tp = ↑ T;
Var
  P: Tp; A: T;
Begin
  …
New(P)
…
P ↑.Nome:=“Maria”;
…
New(P ↑.Padre);
…
P ↑.Padre ↑.Nome:=“Luca”;
P ↑.Padre ↑.Madre:= NIL
```



Liste

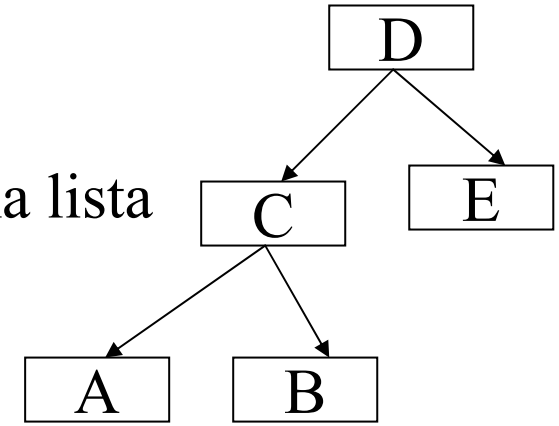
Le Liste possono essere Lineari e NON Lineari

Lista Lineare:

- è un insieme ordinato di *n* elementi, tutti dello stesso tipo,
- ogni elemento ha un predecessore e un successore

Lista Non Lineare:

- è una lista
- ogni elemento può essere a sua volta una lista



Tipo di Dato Astratto

Lo scopo di definire un tipo di dato è quello di fissare il campo dei valori che un dato potrà assumere e definire le operazioni che permettono la manipolazione dei dati di quel tipo.

Un tipo di dato astratto viene rappresentato tramite strutture dati statiche o dinamiche

Lista

Coda

Pila

Lista Lineare

è un insieme ordinato di *n* elementi, tutti dello stesso tipo, ogni elemento ha un predecessore e un successore

<u>Proprietà:</u>

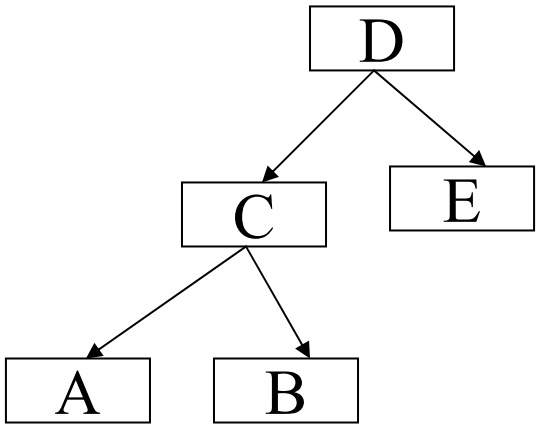
1. X_1 è il primo elemento della lista
2. X_i è preceduto da xi-1 e seguito da xi+1
3. X_n è l'ultimo elemento della lista
4. N= lunghezza della lista
5. N=0 $\Rightarrow$ Lista Vuota

L(A, B, C, D)

Relazione di precedenza: per accedere ad un elemento si deve accedere a tutti quelli che lo precedono

Lista Non Lineare

è una lista, ogni elemento può essere a sua volta una lista



L(D (C (A, B), E))

Lista Lineare come Tipo di Dato Astratto

Dominio dei valori

- Insieme dinamico basato su un elemento omogeneo

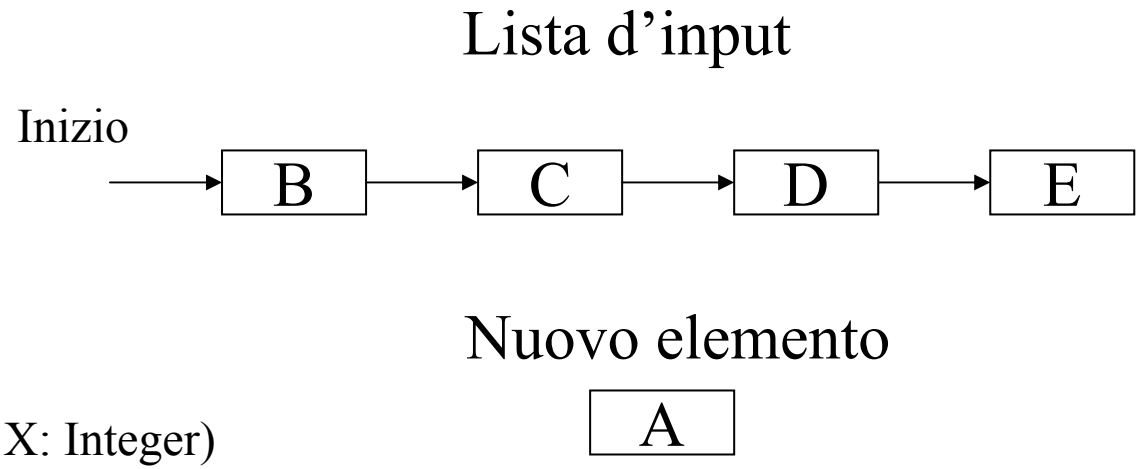
Operazioni su Lista Lineare

- Inserire un elemento in testa
- Inserire un elemento in coda
- Inserire un elemento dopo un elemento puntato da *P*
- Creare una lista con *N* elementi
- ….

Operazioni su Liste Lineari

Inserimento di un elemento in Testa alla Lista

```
Type
  T = RECORD
          Inf : Integer;
          Next : TP;
        END;
  Pun = ↑ T;
Var
  Primo: Pun; A: T;
Procedura Testa (Var Inizio: Pun, X: Integer)
 Var Q: Pun;
Begin
  New(Q);
  Q↑.Inf := X;
  Q↑.Next := Inizio;
  Inizio:= Q;
END;
```



Inserimento di un elemento in testa ad una lista lineare

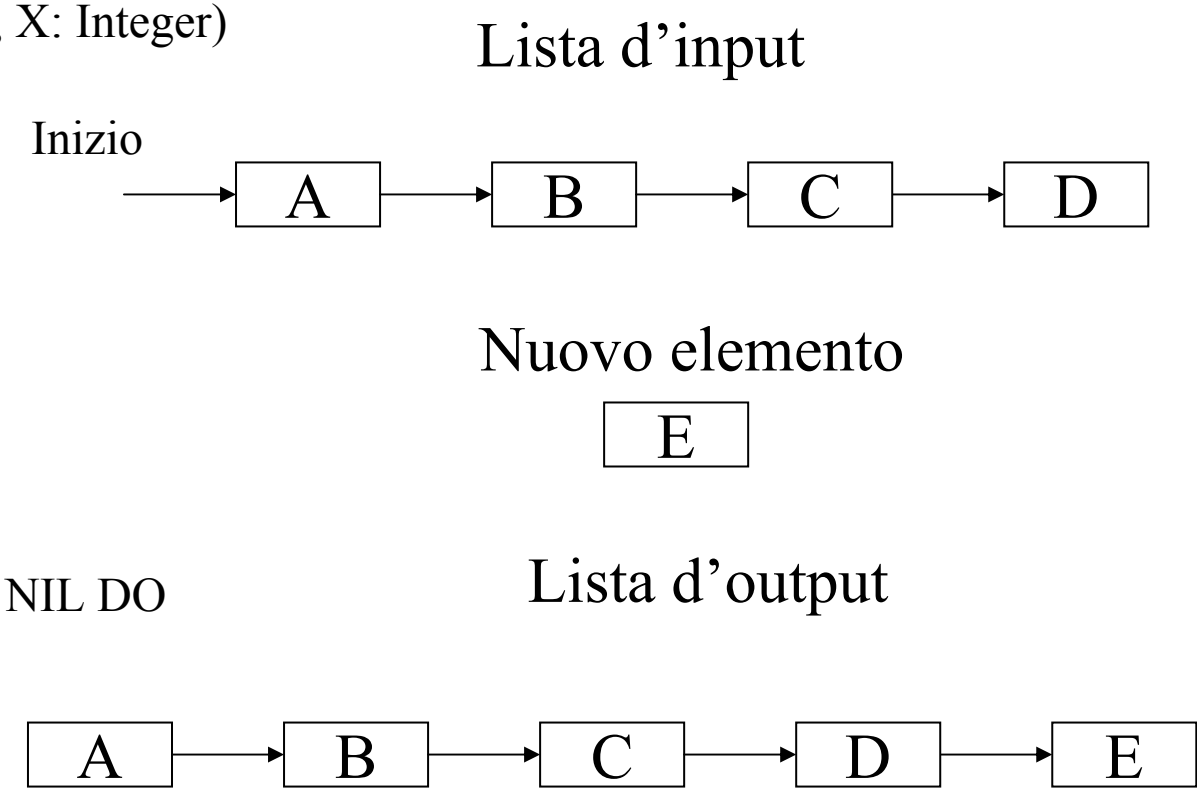
Esercizio

Definire una Funzione Testa di tipo puntatore che prenda in input un valore X e restituisca Testa con X come primo elemento

Inserimento di un elemento in coda

…

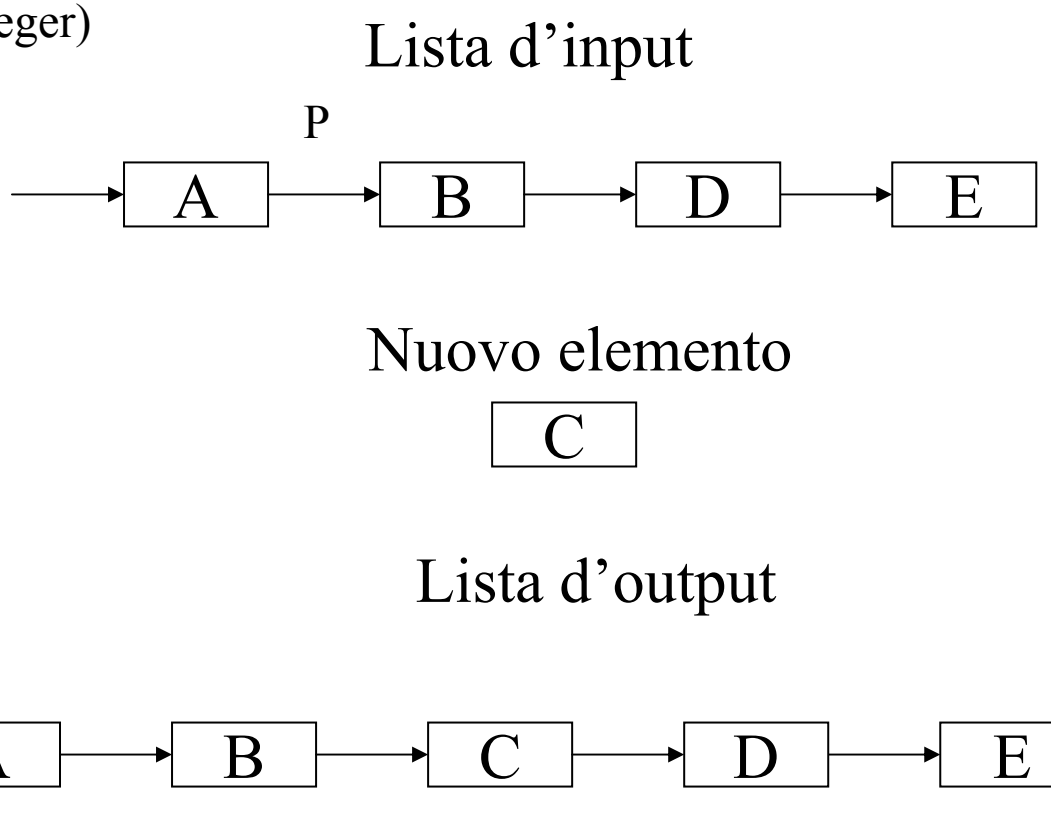
```
Procedura Coda (Var Inizio: Pun, X: Integer)
 Var P, Q: Pun;
Begin
  New(Q);
  Q↑.Next := NIL;
  Q↑.Inf := X;
  P:= Inizio;
  IF Inizio=NIL THEN Inizio:=Q
      ELSE Begin
              WHILE P↑.Next <> NIL DO
                 P:=P↑.Next;
  P↑.Next:=Q;
END;
```



Inserimento di un elemento dopo un elemento puntato da *P*

…

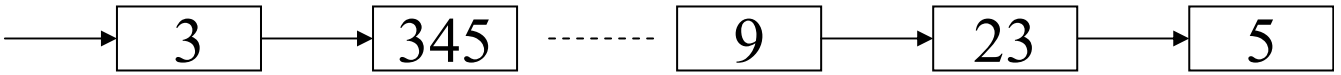
```
Procedura Inserisci (Var P: Pun, X: Integer)
 Var Q: Pun;
Begin
  New(Q);
  Q↑.Inf := X;
  Q↑.Next:= P↑.Next;
 P↑.Next:=Q;
END;
```



Creazione di una lista di *N* elementi

…

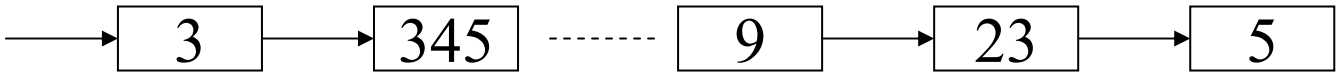
```
Procedura Crea (Var Inizio: Pun, N: Integer)
 Var P, Q: Pun;
Begin
 Inizio:= NIL;
 WHILE N>0  DO
  Begin
     New(Q);
     Q↑.Next := Inizio;
     Inizio:= Q;
     Q↑.Inf:=N;
     N:= N-1
END;
```



Scansione di una lista lineare

…

```
Procedura Scansione (Var Inizio: Pun)
Begin
   WHILE Inizio <> NIL  DO
        Begin
          Writeln(Inizio.Inf);
          Inizio:= Inizio↑.Next;
        End;
End;
```



Esercizio 1

Data un elenco di parole, si vuole costruire una **Lista Ordinata** contenente un elemento per ogni distinta parola. Il generico elemento della Lista conterrà un campo informazione (la parola) e un contatore che indicherà il numero di occorrenze della parola nell'elenco.

Inserimento in Liste Ordinate

```
Type
  Pun  =↑ Parola;
  Parola = RECORD
                Inf : Stringa;
                Conta: Integer
                Next : Pun;
           END;
Var
K: Stringa; Radice:Pun;
Procedura Ricerca (Var Inizio: Pun, X: stringa)
 Var W1,W2 : Pun;
Begin
  W2:= Inizio; W1:=W2 ↑.Next;
  IF W1 = NIL THEN Inserisci (NIL) ELSE
     Begin
       WHILE (W1.Inf < X AND (W1↑.Next <> NIL)
           DO Begin
                  W2:=W1;
                    W1:=W1 ↑.Next
                End;
 IF    W1↑.Inf = X      THEN
                    W1 ↑. Conta:= W1 ↑.Conta+1
             ELSE  Begin
                   IF W1↑.Next:= NIL THEN
                                Begin
                                   W2:=W1;
                                    W1:= NIL
                                End
                 Inserisci (W1)
              End

Procedura Inserisci (W:Pun)
  Var W3: Pun;
  Begin
    New (W3);
    W3↑.Inf:= X;
    W3 ↑.Conta:°= 1;
    W3 ↑.Next:=W
  End ;

Begin
  New(Radice); Radice ↑.Next=NIL; Read(X);
  While K <> "Fine" DO Begin
                                  Ricerca(Radice, K)
                                  Read(K)
                                End;
End.
```