

Parte 4C

Liste ordinate



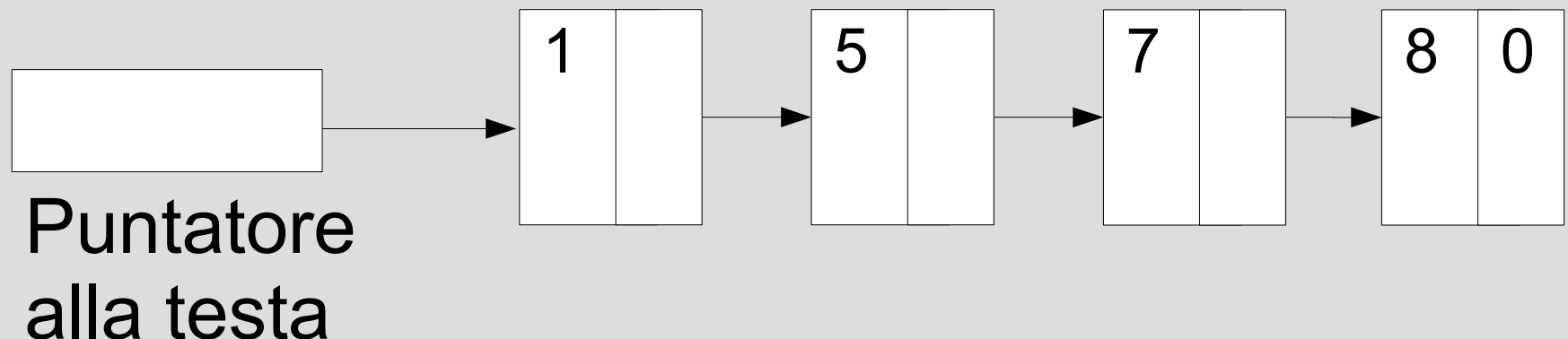
H. Matisse – Dance I, 1909



H. Matisse – Dance, 1910

Lista ordinata

- **Una lista è ordinata** se l'ordine con cui compaiono gli elementi corrisponde ad un qualche ordinamento tra i valori di uno dei campi informazione (spesso chiamato *chiave*)
- Esempio di lista ordinata in senso crescente dei valori del campo informazione:



Inserimento in ordine

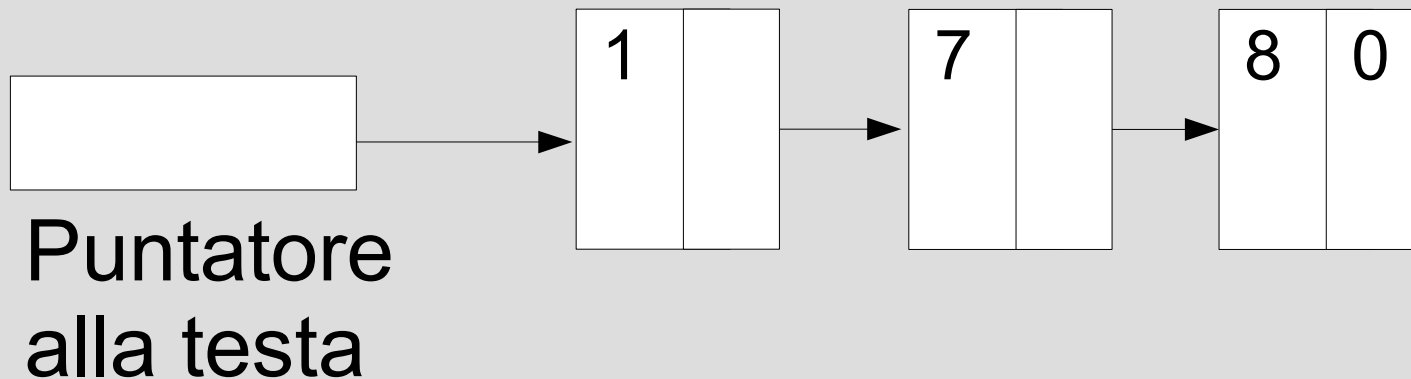
- Algoritmo per inserire un elemento in una lista già ordinata ***preservando l'ordine*** tra gli elementi
- L'inserimento in ordine può ricadere in tre casi distinti a seconda del valore dell'elemento nuovo:
 - ***Inserimento in mezzo***
 - ***Inserimento in testa***
 - ***Inserimento in fondo***

Inserimento in ordine

- 1) ***Ricerca della posizione in cui inserire l'elemento***
- 2) Creazione del nuovo elemento
- 3) Inizializzazione del campo informazione
- 4) Aggancio dell'elemento alla lista
 - 1) Inizializzazione del campo puntatore dell'elemento
 - 2) Intervento sul resto della lista:
 - 1) Se si inserisce in testa, aggiornamento del puntatore alla testa della lista
 - 2) Se in mezzo, aggiornamento del campo puntatore dell'elemento precedente

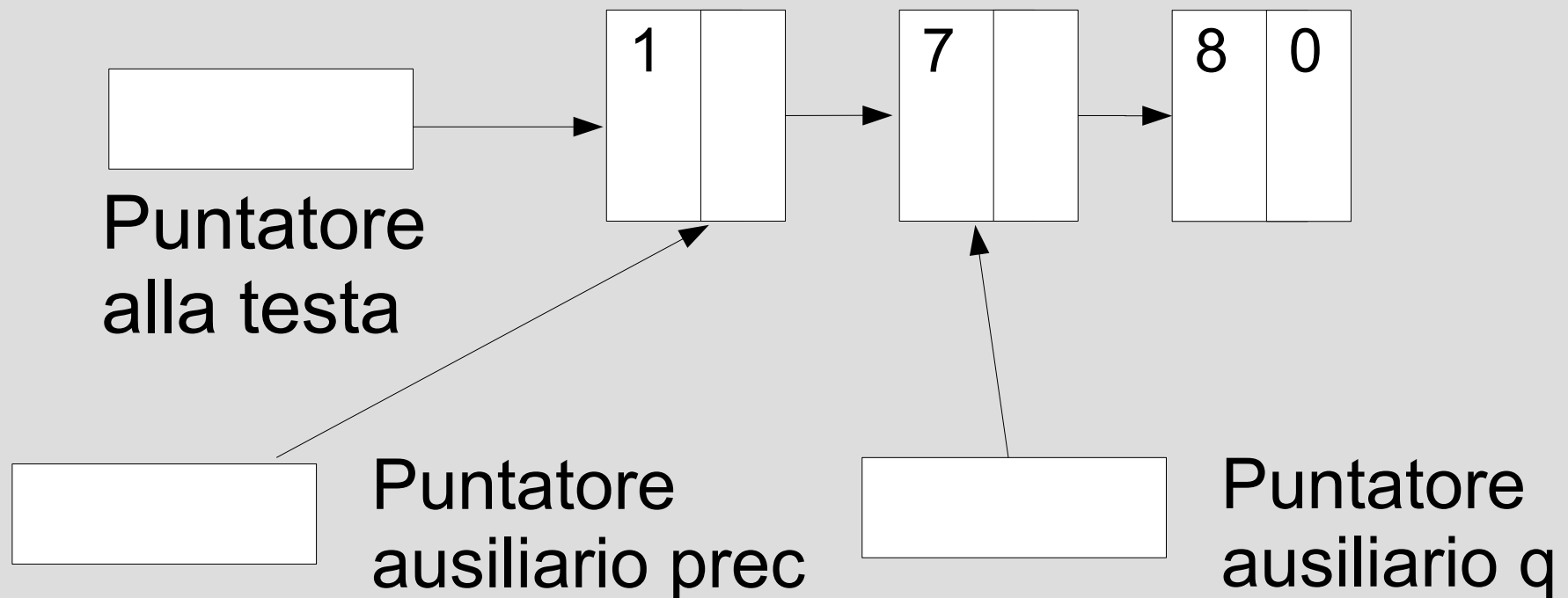
Inserimento in mezzo

- Si supponga di voler inserire un elemento con il valore 5 nella seguente lista ordinata in senso crescente
- Quanti puntatori utilizzare?



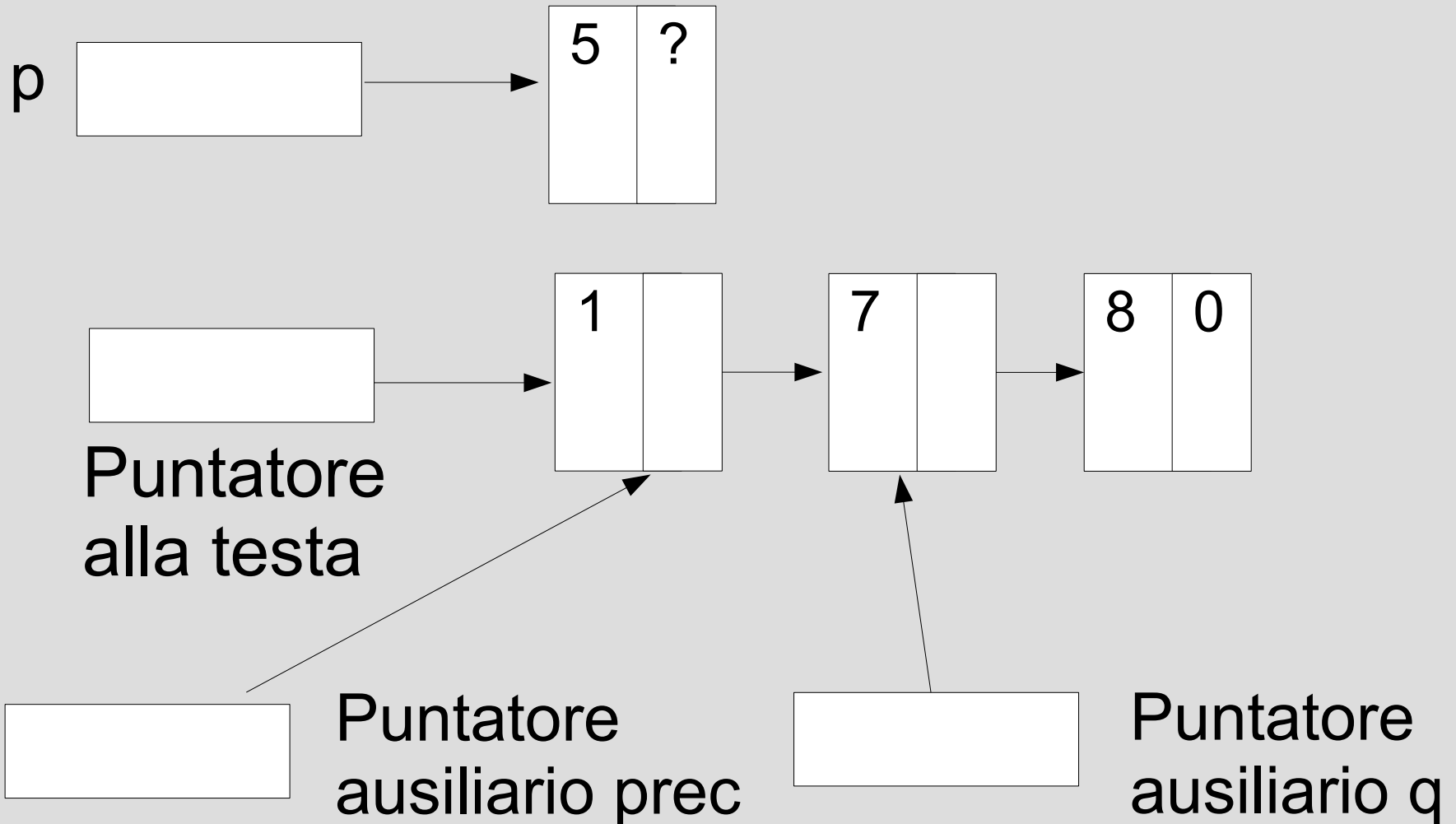
Inserimento in mezzo

- Utilizziamo una coppia di puntatori, q e prec, e posizioniamo q sul **primo elemento di valore maggiore di 5** e prec sull'elemento precedente



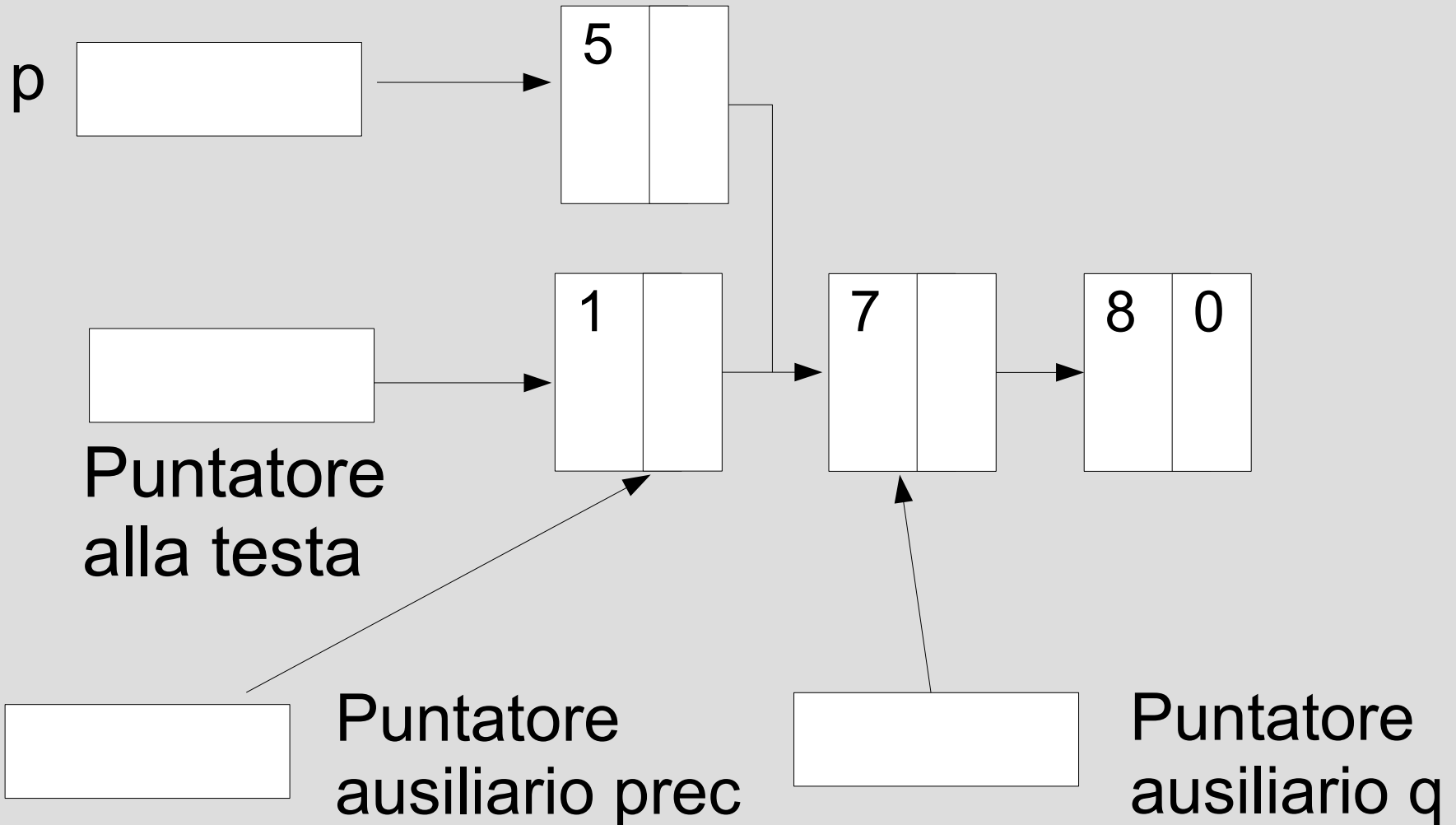
Inserimento in mezzo

- Creazione del nuovo elemento



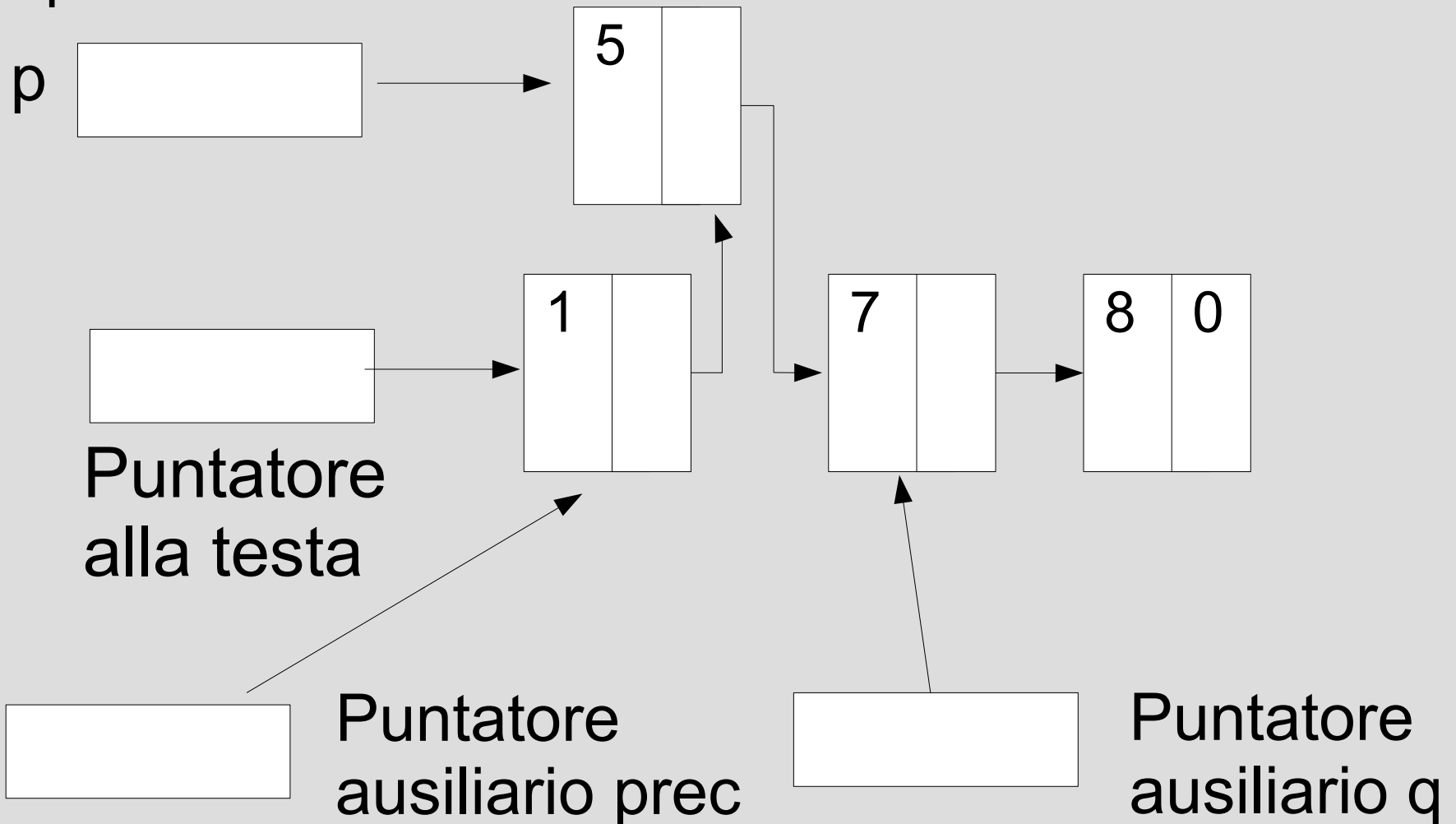
Inserimento in mezzo

- Inizializzazione del campo puntatore



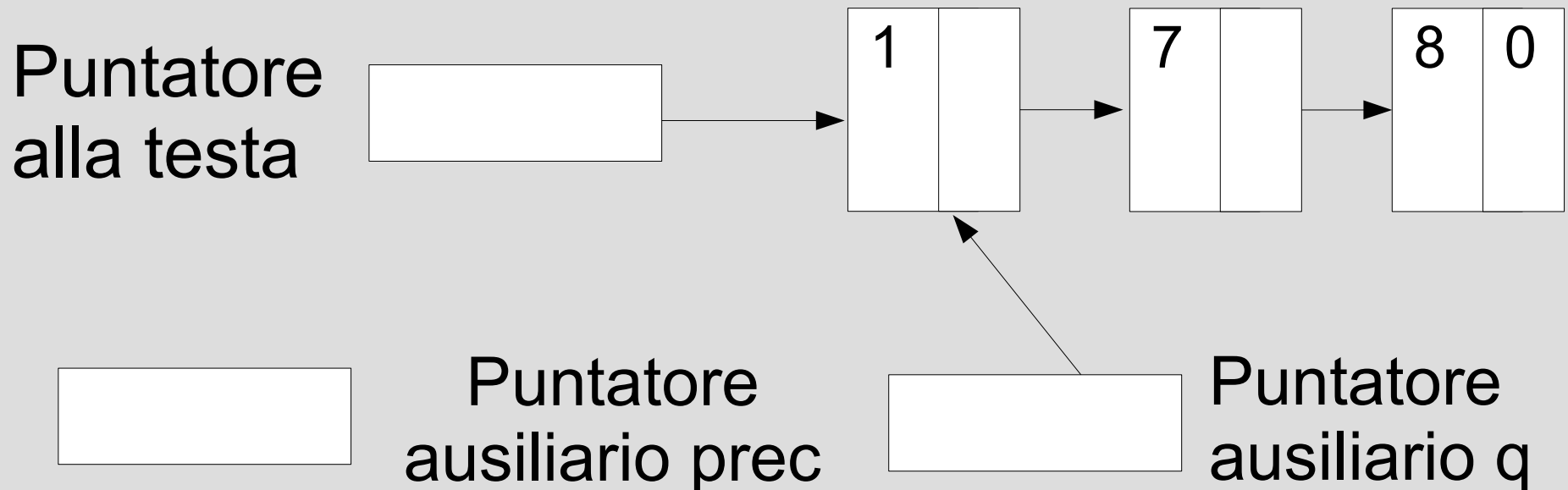
Inserimento in mezzo

- Aggiornamento del campo puntatore dell'elemento precedente



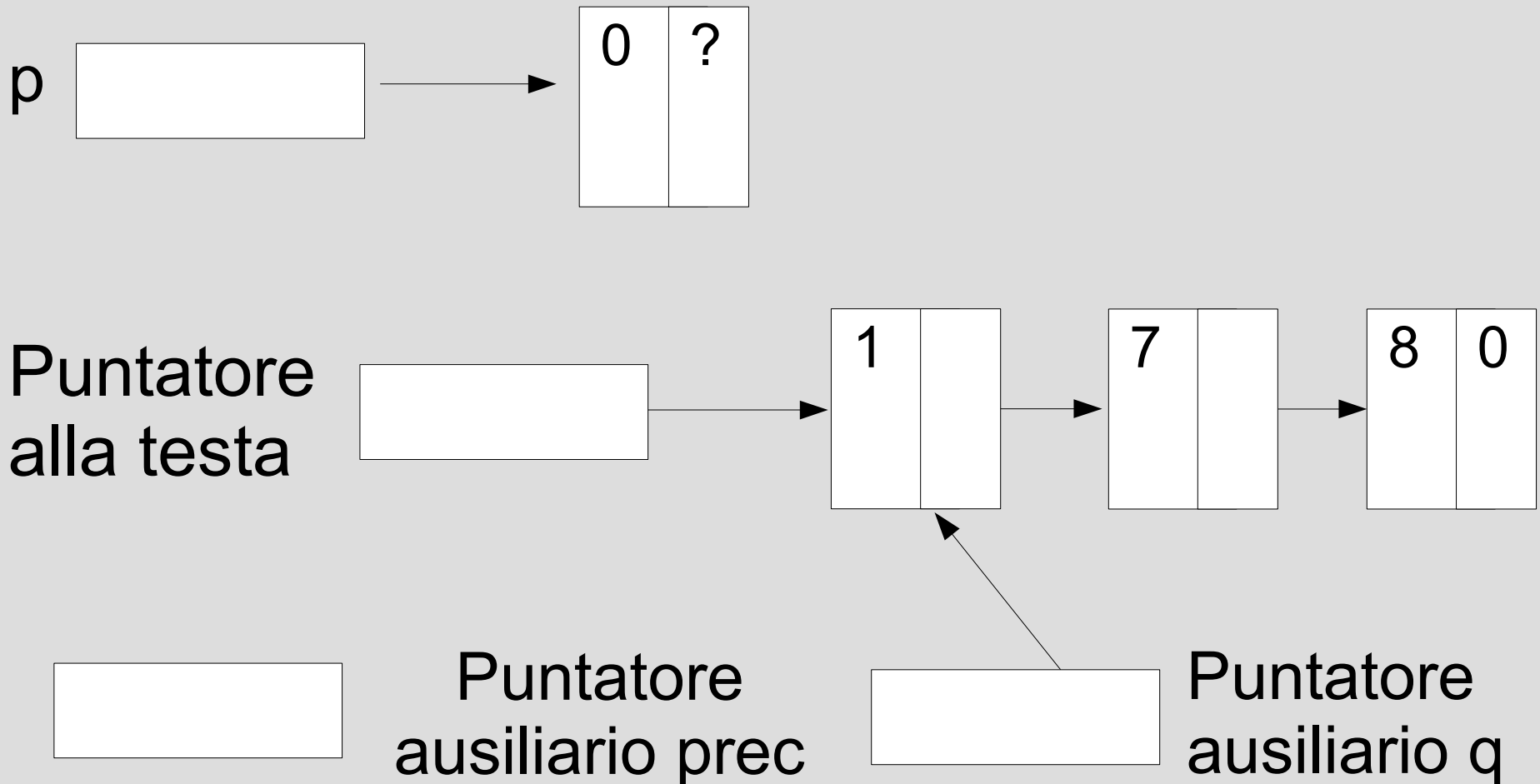
Inserimento in testa

- Si supponga ora di voler inserire un elemento con il valore 0 nella lista ordinata
- Il primo elemento di valore maggiore di 0 è il primo → la situazione dei puntatori in questo caso è la seguente



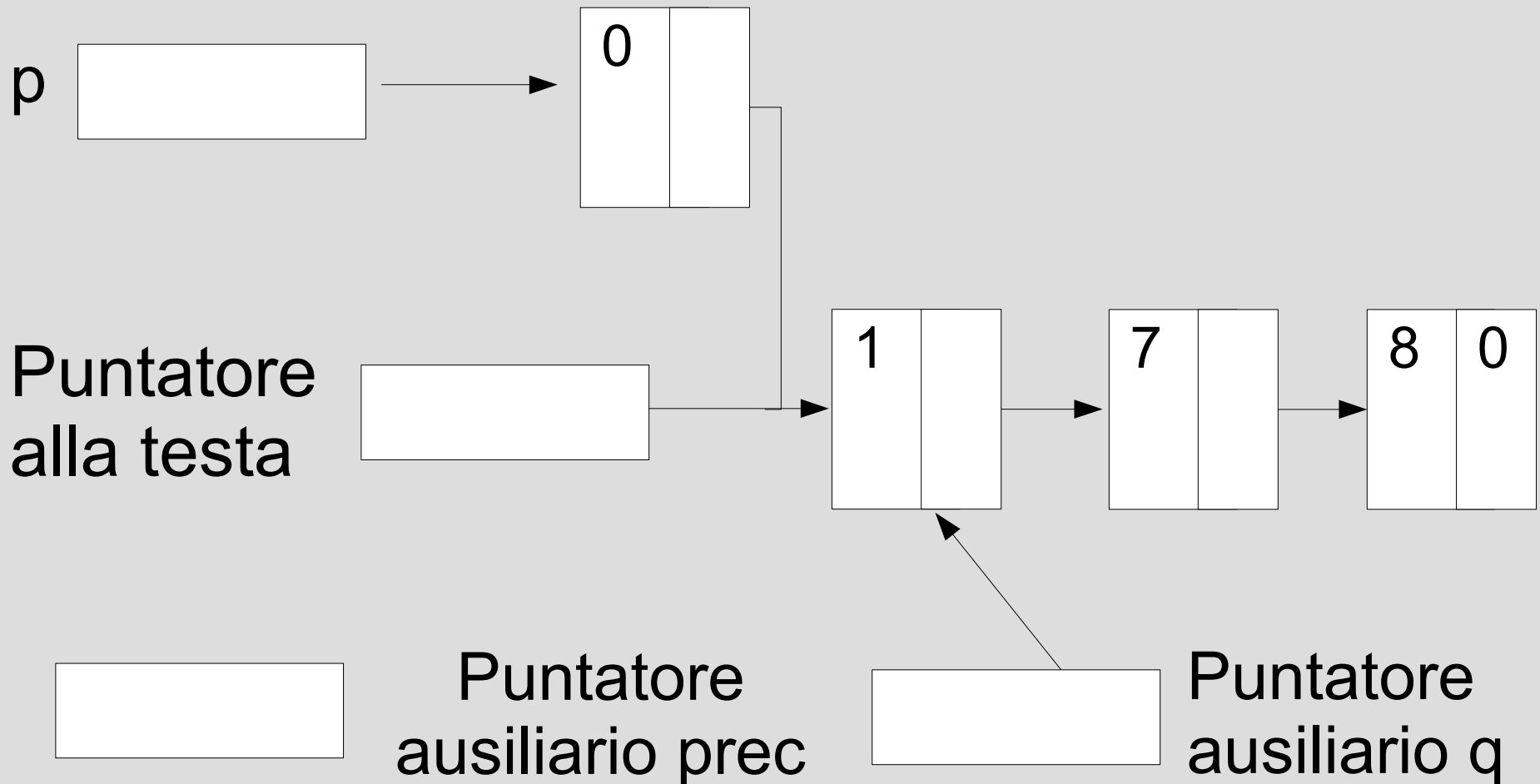
Inserimento in testa

- Creazione del nuovo elemento



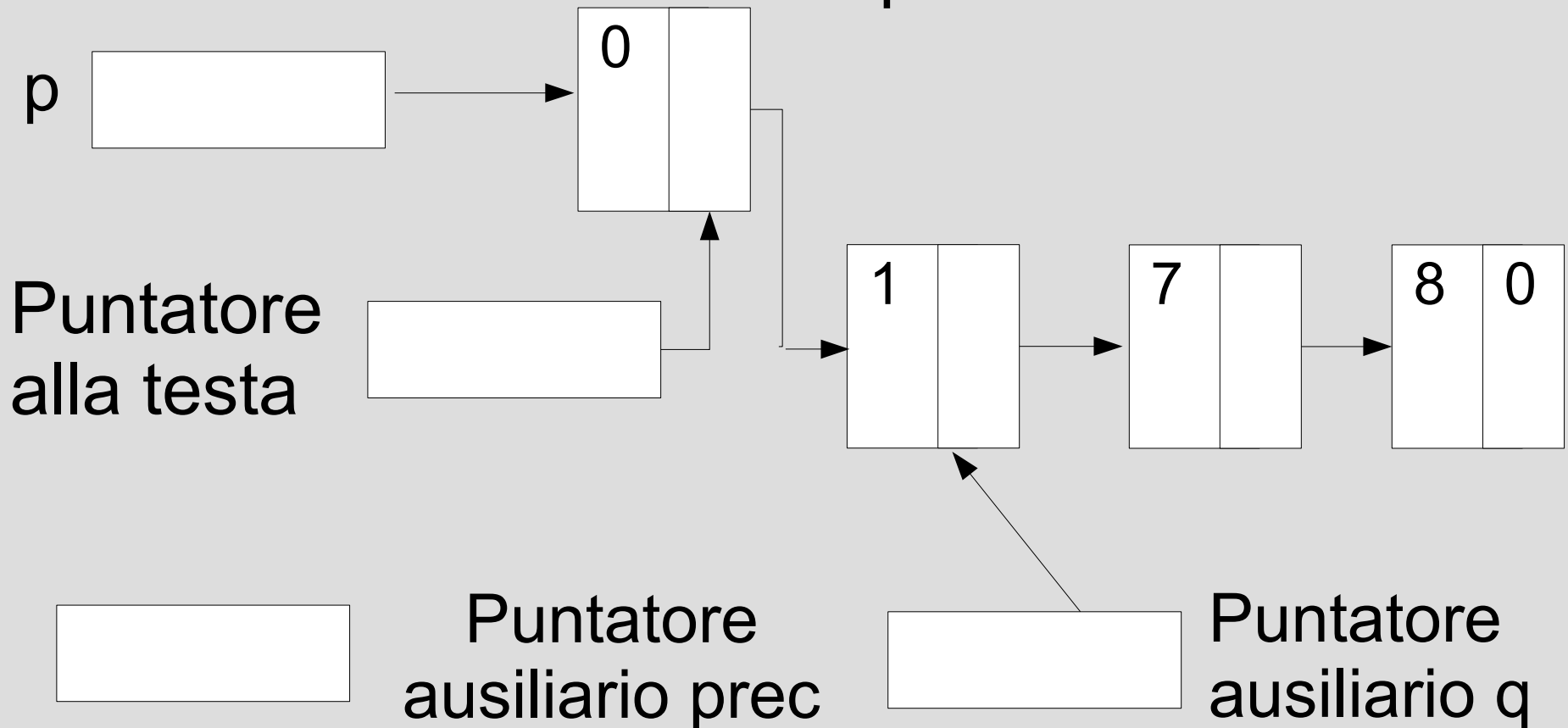
Inserimento in testa

- Aggiornamento del campo puntatore



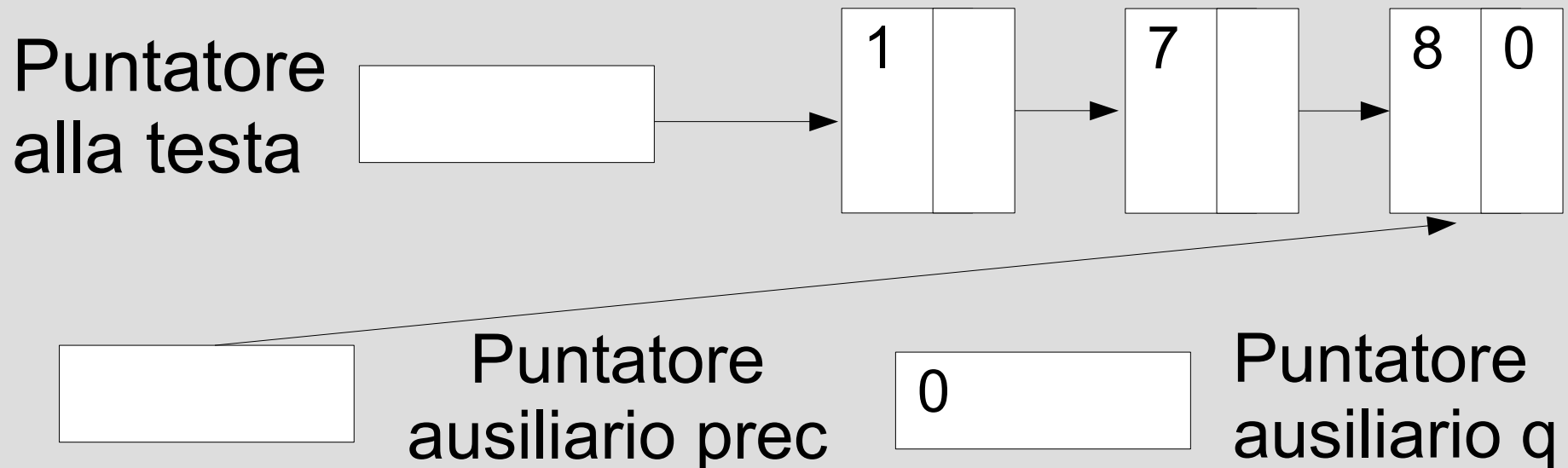
Inserimento in testa

- Dal valore del puntatore q si deduce che si deve inserire in testa \rightarrow aggiornamento del puntatore alla testa



Inserimento in fondo

- Si supponga di voler inserire un elemento con il valore 10 nella seguente lista ordinata
- Siccome non c'è un elemento di valore maggiore di 10, la coppia di puntatori si posiziona come segue



Inserimento in fondo

- Come si completa?
- Il caso di inserimento in fondo va gestito a parte rispetto al caso di inserimento in mezzo ad una lista ordinata?

No

1) Il campo puntatore assume il valore di q (in questo caso NULL)

2) Deve essere aggiornato il campo puntatore dell'elemento cui punta prec

Programma

- Programma *lista_ordinata.cc*
 - Supporre che la funzione *ins_ord*:
 - Abbia come valore di ritorno di tipo *void*
 - Prenda come parametri il puntatore alla testa e il valore intero che si vuole inserire nel campo informazione del nuovo elemento
 - Si può partire da *lista_ordinata_solo_main.cc*

Strutture dati auto-organizzanti

- Strutture che *si adattano alla sequenza delle operazioni effettuate (per migliorare le prestazioni degli accessi futuri)*
- Esempio: lista non ordinata i cui elementi sono etichettati con un identificatore
- Quanti passi sono necessari per accedere ad un elemento?
 - Numero proporzionale alla posizione nella lista dell'elemento da accedere

Strategie di riorganizzazione

- Si possono adottare **strategie di riorganizzazione delle strutture dati** per minimizzare il più possibile il costo di ciascun accesso futuro
- Diverse strategie possibili, tra cui l'algoritmo **“sposta in testa”**
 - **Ipotesi di base**: l'ultimo elemento acceduto è quello a cui si accederà nel futuro con maggior probabilità (località temporale)
 - **Algoritmo che sposta in testa l'ultimo elemento acceduto**

Sposta in testa

- Traccia *stesta*: *traccia_stesta.txt*
- Consideriamo una ***lista non ordinata*** in cui ogni elemento contenga ***2 campi informazione***:
 - uno di ***tipo intero*** detto ***'identificatore'***
 - l'altro di ***tipo stringa*** detto ***'messaggio'***
- Vedere la traccia per le funzionalità da implementare