Java

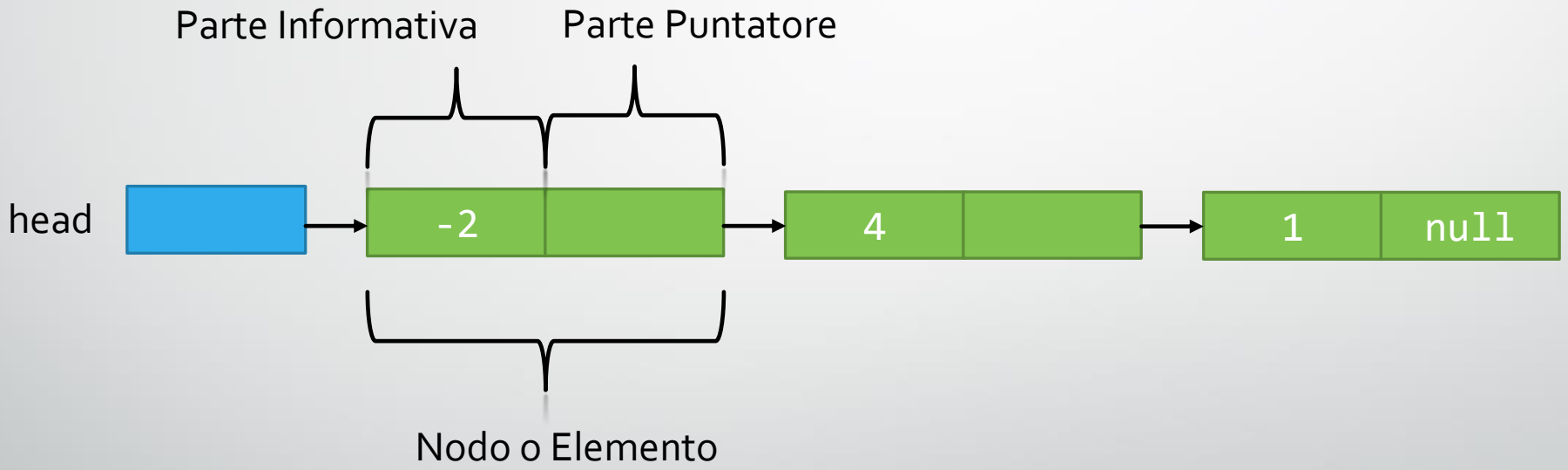
Capitolo 6 – Gestione Lista

Prof. Ivan Gentile

Lista

- Una lista è una struttura dati (dato strutturato)
- In generale gli elementi non occupano posizioni contigue
- Di solito è gestita attraverso puntatori (**lista a puntatori**)

Lista a puntatori



Vari tipi di liste a puntatori

- Esistono vari tipi di liste a puntatori
- **Lista circolare**: l'ultimo elemento punta al primo
- **Lista bidirezionale**: due puntatori per ogni elemento
- *Lista con altri puntatori*: esempio un puntatore di coda e/o in specifice condizioni
- La lista di base è allora detta **lista semplice**

Struttura di una lista a puntatori (semplice)

- Una classe **Nodo**: generico elemento della lista
- Una classe **Lista**: con il solo attributo head
  - Poi ci possono essere altri puntatori a seconda del tipo di lista
  - Metodi per operare sulla lista (inserisci in testa, in coda, in una certa posizione, elimina, toString, etc.)

Confronto tra Lista e array

- Accesso:
  - Diretto per gli array
  - Sequenziale o quasi per le liste
- Ridimensionamento:
  - Semplice per le liste
  - Oneroso per gli array
- Occupazione di memoria:
  - leggermente maggiore per le liste
  - Ma le liste non hanno spreco di locazioni
- Inserimento e cancellazione
  - Oneroso per array
  - Semplice per le liste

Classe Nodo

```java
public class Nodo {
    private int info;
    private Nodo link;

    public Nodo () {
        link = null;
        info = 0;
    }

    public Nodo (int info) {
        link = null;
        this.info = info;
    }
```

```java
    // Metodi Getter e Setter
    public void setInfo(int info) {this.info = info;}
    public int getInfo() {return info;}
    public void setLink(Nodo link) {this.link = link;}
    public Nodo getLink() {return link;}

    public String toString() {
        return Integer.toString(info);
    }
}
```

Possiamo usare i generics

```java
public class Nodo<T> {
    private T info;
    private Nodo<T> link;

    public Nodo() {
        link = null;
        info = null;
    }

    public Nodo(T info) {
        link = null;
        this.info = info;
    }
```

```java
    public void setInfo(T info) {
        this.info = info;
    }

    public T getInfo() {
        return info;
    }

    public void setLink(Nodo<T> link) {
        this.link = link;
    }

    public Nodo<T> getLink() {
        return link;
    }

    public String toString() {
        return info.toString();
    }
}
```

Classe Lista

- Unico attributo obbligatorio head
  - che punta al primo elemento della lista
  - sarà di tipo "*Nodo*"
- Poi ci sono tutti i metodi di inserimento e cancellazione


```
public Lista() { head = null; }
```

Passare da un elemento a un altro

- Se head punta a un elemento
- `head.getLink()` è l'indirizzo del successivo
- Per fare avanzare head: `head = head.getLink();`
- Chiramente head non va modificato!!!
- Quindi mi creo una variabile di appoggio **p che inizialmente è pari a head**
- Scorrimento sarà: **`p = p.getLink();`**

Gestione di una Lista

```
**********************************************************************
GESTIONE DI UNA LISTA
1 - Generazione di una lista di numeri casuali
2 - Cancellazione lista
3 - Inserimento di un elemento in testa
4 - Inserimento di un elemento in coda
5 - Inserimento di un elemento in una specifica posizione
6 - Cancellazione di un elmento in testa
7 - Cancellazione di un elmento in coda
8 - Cancellazione di un elmento in una specifica posizione
9 - Ricerca di un elemento
10 - Stampa della lista
11 - Fine
Scegli l'operazione:
```