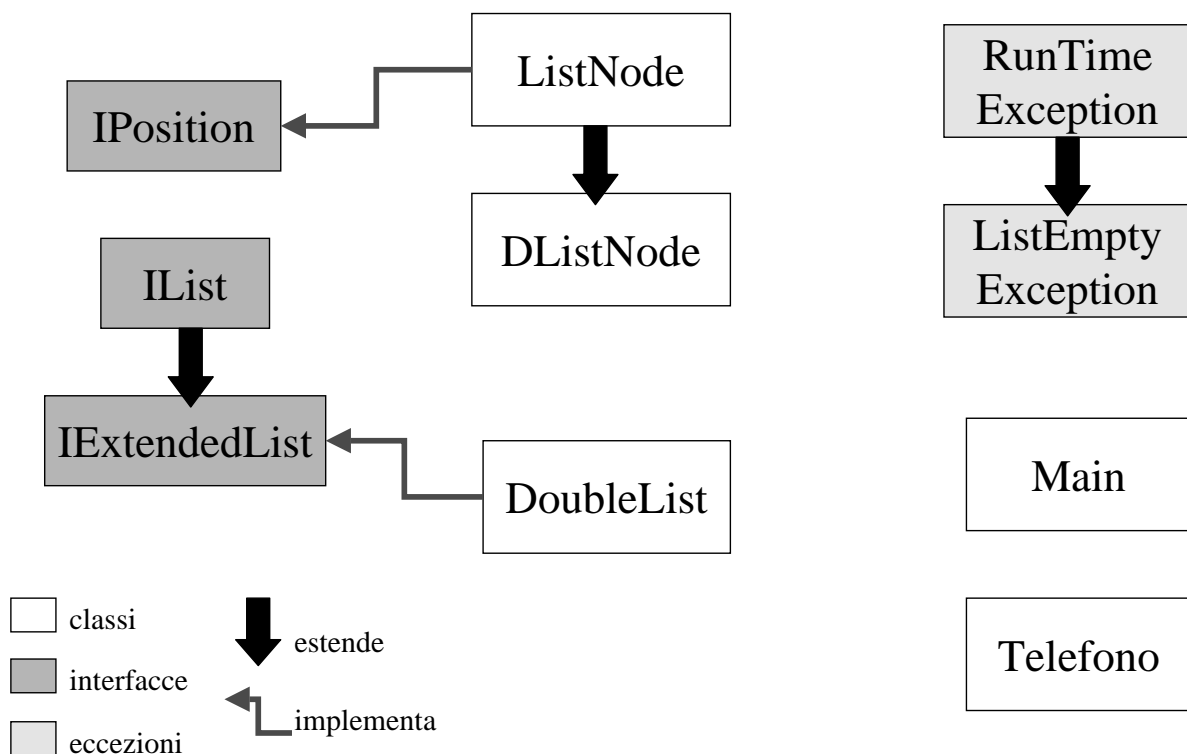


# Esercitazione n° 5

## Obiettivi:

- Capacità di *analisi* e di *estensione* di progetti esistenti
- Strutture dati dinamiche: *Liste doppiamente concatenate*
- Linguaggio Java:
  - Ancora *casting*, *interfacce* ed *eccezioni*
  - Estensione dell'esempio `DoubleList` visto in aula
    - *Inserimento ordinato*
    - Singoli dati come oggetti della classe `Telefono`

## Il package `Esercitazione5`: organizzazione



## Esercizio

---

- Realizzazione di una *rubrica telefonica* i cui singoli elementi fanno parte di una *lista doppiamente concatenata*
- Estensione della classe `DoubleList` vista in aula per aggiungere funzionalità di:
  - *inserimento ordinato* nuovi elementi (contatti telefonici)
  - *stampa* di tutti i contatti il cui cognome comincia per una data lettera, come se sfogliassimo una rubrica (compresi i contatti precedente e successivo all'insieme determinato)

## L'interfaccia `IExtendedList`

---

```
interface IExtendedList extends IList {  
  
    // Inserisce un nuovo elemento in modo ordinato  
    // in una lista ordinata  
    public void insOrd(Object nuovo);  
  
    // Stampa tutti i contatti telefonici che  
    // cominciano per ch, + precedente e successivo  
    public void stampaPerIniziale(char ch);  
}
```

## La classe Telefono (1)

---

```
public class Telefono {  
  
    private String nome, cognome, numero;  
  
    public Telefono(String a, String b, String c) {  
        nome = new String(a);  
        cognome = new String(b);  
        numero = new String(c); }  
  
    public static boolean cominciaPer(Object o, char  
    ch) {  
        Telefono t = (Telefono)o;  
        if (t.cognome.charAt(0)==ch) return true;  
        else return false; }  
}
```

## La classe Telefono (2)

---

```
public static boolean isLess(Object o, char ch) {  
    Telefono t = (Telefono)o;  
    if (t.cognome.charAt(0) < ch) return true;  
    else return false; }  
  
public static boolean isLess(Object o1, Object o2) {  
    Telefono t1 = (Telefono)o1;  
    Telefono t2 = (Telefono)o2;  
    if (t1.cognome.compareTo(t2.cognome) < 0)  
        return true;  
    else if ((t1.cognome.compareTo(t2.cognome)==0)&&  
        ((t1.nome.compareTo(t2.nome) < 0))) return true;  
    else return false; }  
}
```

## La classe Telefono (3)

---

```
public String toString() {  
  
    return (nome + ", " + cognome + ", " + numero);  
}
```

## La classe Main (1)

---

```
public static void main(String[] args) {  
    IExtendedList ifList;  
    Object o; Telefono tel;  
    DoubleList dlist = new DoubleList();  
    ifList = dlist;  
  
    System.out.println("Attuale dimensione della doppia  
        lista: " + ifList.size());  
    tel=new Telefono("Cesare","Stefanelli","0512093087");  
    ifList.insOrd(tel);  
    ifList.insOrd(new  
        Telefono("Michela","Milano","0512093019"));  
    ifList.insOrd(new  
        Telefono("Antonio","Corradi","0512093083"));
```

## La classe `Main` (2)

---

```
while (ifList.isEmpty() != true) {
    o = ifList.removeFirst();
    System.out.println("Estratto " + o + "\nda lista "
        + ifList); }

System.out.println("Attuale dimensione della doppia
    lista: " + ifList.size());
tel = new Telefono("Paolo", "Bellavista", "0512093087");
ifList.insOrd(tel);
ifList.stampaPerIniziale('S'); }
}
```

## La classe `DoubleList` (1)

---

```
public void stampaPerIniziale(char ch)
    throws ListEmptyException {

    DListNode currentNode;
    if (isEmpty()) throw new ListEmptyException
        ("Lista vuota!");
    else { currentNode = first;
        while ((Telefono.isLess(currentNode.getItem(), ch))
            &&(currentNode.getNext()!=null))
            currentNode=(DListNode)currentNode.getNext();

        if (Telefono.cominciaPer(currentNode.getItem(), ch))
            { if (currentNode.getPrev()!=null)
                System.out.println("Elemento precedente: " +
                    ((DListNode)currentNode.getPrev()).getItem());
```

## La classe `DoubleList` (2)

---

```
System.out.println("Elemento nel range: " +
    currentNode.getItem());
while (((currentNode=(DListNode)currentNode.
getNext())!= null) && (Telefono.cominciaPer
(currentNode.getItem(), ch)))
    System.out.println("Elemento nel range: " +
        currentNode.getItem());
if (currentNode != null)
    System.out.println("Elemento successivo: " +
        currentNode.getItem());
    }
}
}
```

## L'interfaccia `IExtendedList`

---

```
interface IExtendedList extends IList {

    // Inserisce un nuovo elemento in modo ordinato
    // in una lista ordinata
public void insOrd(Object nuovo);

    // Stampa tutti i contatti telefonici che
    // cominciano per ch, + precedente e successivo
public void stampaPerIniziale(char ch);
}
```