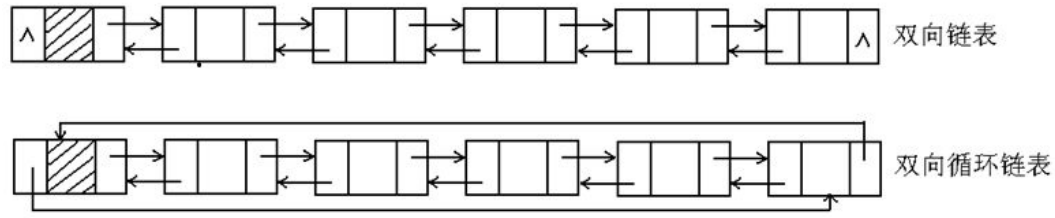
双链表

双链表结构



特点

> this.head.prev = this.lastEle
>
> this.lastEle.next = this.head

双节点类实现

```
public class DoubleNode<T> {
    public T data;
    public DoubleNode<T> prev,next;
    public DoubleNode(){
        this.data = null;
        this.prev = null;
        this.next = null;
    }
    public DoubleNode(T data,DoubleNode<T> prev,DoubleNode<T> next){
        this.data = data;
        this.prev = prev;
        this.next = next;
    }
    public DoubleNode(T data){
        this.data = data;
        this.prev = null;
        this.next = null;
    }
    public String toString(){
        return this.data.toString();
    }
}

```

循环双链表类实现

```
public class CirDoubltList<T> {
    public DoubleNode<T> head;
    public CirDoubltList(){
        this.head = new DoubleNode<T>();
        this.head.prev = this.head;
        this.head.next = this.head;
    }
    public CirDoubltList(T[] values){
```

```
        this();
        DoubleNode<T> p = this.head;
        for(int i=0;i<values.length;i++){
            p.next = new DoubleNode<T>(values[i],p,null);
            p = p.next;
        }
        p.next = this.head;
        this.head.prev = p;
    }
    public boolean isEmpty(){
        return  this.head.next == this.head;
    }
    public DoubleNode<T> insert(T x,int i){
        if(x == null){
            throw new NullPointerException();
        }
        DoubleNode<T> p = this.head;
        for(int j = 0;p.next!=this.head&&j<=i;j++){
                            //这里运用了双链表的特点。
            p = p.next;
        }
        DoubleNode<T> ins = new DoubleNode<T>(x,p.prev,p);
        p.prev.next = ins;
        p.prev = ins;
        return ins;
    }
    public DoubleNode<T> remove(int i){
        if(this.head.next == this.head){
            return null;
        }
        DoubleNode<T> unluckyEle = this.head.next;
        for(int j=0;j<i;j++){
            unluckyEle = unluckyEle.next;
        }

        unluckyEle.prev.next = unluckyEle.next;
        unluckyEle.next.prev = unluckyEle.prev;

        return unluckyEle;
    }
    public String printAll(){
        String str = this.getClass().getName()+"(";
        DoubleNode<T> p = this.head;
        while(p.next!=this.head){
            p=p.next;
            str+=p.data;
        }
        str+=")";
        return str;
    }
}

```