```java
// ******************************************************************************
//
// File:    UndirectedEdge.java
// Package: ---
// Unit:    Class UndirectedEdge
//
// ******************************************************************************

/**
 * Class UndirectedEdge represents an edge in a graph that connects two
 * vertices. It's important to note that the edge does not have a direction nor
 * weight.
 *
 * @author Jimi Ford
 * @version 2-15-2015
 */
public class UndirectedEdge {

    // private data members
    private Cricket a, b;

    // future projects may rely on a unique identifier for an edge
    private final int id;

    /**
     * Construct an undirected edge
     * @param id a unique identifier to distinguish between other edges
     * @param a one vertex in the graph
     * @param b another vertex in the graph not equal to <I>a</I>
     */
    public UndirectedEdge(int id, Cricket a, Cricket b) {
        this.id = id;
        // enforce that a.n is always less than b.n
        if(a.n < b.n) {
            this.a = a;
            this.b = b;
        } else if(b.n < a.n) {
            this.a = b;
            this.b = a;
        } else {
//          System.out.println(a.n + ", " + b.n +", "+ (a==b));
            throw new IllegalArgumentException("Cannot have self loop");
        }
        this.a.addEdge(this);
        this.b.addEdge(this);
    }

    /**
     * Get the <I>other</I> vertex given a certain vertex connected to
     * this edge
     *
     * @param current the current vertex
     * @return the other vertex connected to this edge
     */
    public Cricket other(Cricket current) {
        if(current == null) return null;
        return current.n == a.n ? b : a;
    }
}
```