Figure 13.2 A first look at *Inheritance*.

Let's write a program to read a mixture of numerical and letter-grade marks, store them in a Vector, and then find the average.

We have data like this: `65 A+ C- 76` … (each mark on a separate line).

Without some fairly tricky programming, we can't tell upon input which are numbers and which are letters, so we'll ask the user:

```
Vector markList = new Vector();
while (true) {
    String mark = keyb.readLine();
    if (mark.equals("quit")) break;
    System.out.print("letter grade? (y/n) ");
    if (keyb.readLine().equals("y")) {
        markList.addElement(mark):
    else
        markList.addElement(new Integer(mark));
}
```

(**Note**: (1) the **else** clause adds an **Integer** object to the Vector, and (2) we create that object with **new Integer(mark)**. It appears that we've created an **Integer** object directly from **mark**, which is a **String**! How is this possible?)

So now we have both numerical and letter-grade marks stored in **markList**. As mentioned on the previous slide, the problem is when we read marks back from **markList** during averaging, we can't tell which is which.

A better way of doing this is to first construct a **Mark** class where each object "knows" what kind of mark it's holding:

```
public class Mark {
    private boolean isLetterGrade;   // this is how the object knows
                                     // what kind of mark it's holding!
    private String markString:
    private int markInt:

    public Mark (String mark, boolean isLetter) {
        isLetterGrade = isLetter;
        markString = mark;
        if (! isLetter)
            markInt = Integer.parseInt(mark);
    }

    public int intValue () {
        if (! isLetterGrade) return markInt;
        if (markString.equals("A+")) return 95:
        if (markString.equals("A")) return 88;
        …
    }
}
```