Talked about: Interfaces and abstract classes (mostly the former). There are ones that you make up yourself (Scoring interface – text & lecture example) and examples from the Java libraries. For example Comparable (but also MouseListener mentioned).

Comparable captures the notion of (natural) order on objects from a class. A class implements Comparable, which means it must provide a body for

    int compareTo(Object other);

What gets compared?  Ans: the calling object and the parameter object.

How do you interpret <, >, =?  Ans: if a is less than b, then a.compareTo(b) < 0. etc.

The String class implements Comparable; ordering realized is lexicographic (alphabetical).

1. If a = “ball”, b = “bat”, then what does a.compareTo(b)? And what does the answer mean? And where (in what class) is all this happening?

2. Suppose you want to compare Infant objects according to the rule: one infant should precede another if the first has a shorter name. How would you expand the Infant definition to realize this?

3. Suppose you want to compare Infant objects according to the rule: one Infant should precede another if the first is younger; and if they are the same age, then use alphabetical order to decide. So if Anna is 5 and Brett is 3, then Brett comes first; but if Brett is also 5, then Anna comes first, since alphabetically her name comes first.

4. Suppose you want to compare Infant objects according to the rule: length of name is the ordering, but with age a tie breaker. So the ordering of two kids name Dana is decided by: younger Dana comes first.

5. How would you sort an array of Infants that have been compared by the rule in #3, above?

6. How would you sort an array of names (strings) of people, putting them in alphabetical order.

7. More on inheritance: Extend Echo: read in a text file, print one character of the first line, two of the second line, and so forth. If the length of a line is longer than its line number, just print the full line.

8. Use inheritance, write a complete program that reads a text file, reports profile of lines with lengths in these groups: 0-10, 10-20, ..90-100, 100+. How would you represent the bins?
Here is the Cipher class (for program #6). Make sure you understand how it works. Write a program that reads in a line of text and an int, then prints the line’s encoding according to that shift.

How would you do this: read in a line of text and an int shift, then encode the first word in the line using shift, the second line using shift+1, the third word using shift + 2, and so forth.

```java
public class Cipher{
    private int shift;

    public Cipher(int s){
        shift = s;
    }

    public String encrypt(String plain){
        plain = plain.toUpperCase();
        String str = "";
        for(int j = 0; j < plain.length(); j++){
            str = str + letterShift(plain.charAt(j));
        }
        return str;
    }

    private char letterShift(char c){
        if (Character.isLetter(c)){
            int cPos = (c - 'A'); // a pos from 0 to 25
            cPos = (cPos + shift) % 26; // deal with overflow
            return ((char)('A' + cPos)); // remake as caps
        }
        else return c;
    }
}
```