Today’s topics

Java
Implementing Decision Trees

Upcoming
More formal treatment of grammars

Reading
Great Ideas, Chapter 2

Implementing a Decision Tree

- Start with a very very simple tree
  - Have just one level of decision
  - Need only one if statement
  - Have already done this kind of thing before
  - No new challenges

Code for Simple Tree

```java
public class SimpBook extends java.applet.Applet
implements ActionListener
{
  TextField mQuery, mAnswer;
  Button bYes, bNo;
  public void init()
  {
    mQuery = new TextField(70);
    mQuery.setText("Do you wish a mathematical approach?");
    mQuery.setHint("Yes");
    mQuery.setHint("No");
    mAnswer = new TextField(70);
    mAnswer.addActionListener(this);
    bYes = new Button("Yes");
    bNo = new Button("No");
    add(mQuery); add(bYes); add(bNo); add(mAnswer);
  }
}
```
Code for Simple Tree (part 2)

```java
public void actionPerformed(ActionEvent event) {
    Object cause = event.getSource();
    if (cause == bYes) {
        mAnswer.setText("Books by Harel or Cooper are nice.");
    } else { // must have been the No button
        mAnswer.setText("Books by Pattis or Biermann should do.");
    }
}
```

The Full Decision Tree

- Now have more levels to worry about
  - Have picked up an additional problem
  - Need to keep track of where we are (or have been)
  - (Almost like exploring a cave -- )
  - (~ or putting book marks in a book)
- Add a variable which is used to record where we’ve been
  - Use the numbers on our diagram for reference
    - (numbers are arbitrary; must be unique)
  - Named the variable `myLocation` to suggest use

Code for the Decision Tree

```java
public class BookPick extends java.applet.Applet implements ActionListener {
    TextField mQuery, mAnswer;
    Button bYes, bNo;
    int myLocation;
    public void init() {
        mQuery.setText("Do you wish a mathematical approach?");
        bYes = new Button("Yes");
        bNo = new Button("No");
        myLocation = 0;
        mAnswer = new TextField(70);
        bYes.addActionListener(this);
        bNo.addActionListener(this);
        add(mQuery); add(bYes); add(bNo); add(mAnswer);
    }
}
```

Code for the Decision Tree (p. 2)

```java
public void actionPerformed(ActionEvent event) {
    Object cause = event.getSource();
    if (myLocation == 0) {
        if (cause == bYes) {
            myLocation = 1;
            mQuery.setText("A programming focus instead of theory?");
        } else if (cause == bNo) {
            myLocation = 2;
            mQuery.setText("Narrow focus instead of overview of CS?");
        }
    }
}
```
Code for the Decision Tree (p. 3)

```java
else if (myLocation == 1)
{
    if (cause == bYes)
    {
        myLocation = 3;
        mAnswer.setText("I recommend 'Oh! Pascal' by D. Cooper.");
    }
    if (cause == bNo)
    {
        myLocation = 4;
        mAnswer.setText("'Algorithmics' by D. Harel is a fine book.");
    }
}
```

Code for the Decision Tree (p. 4)

```java
else if (myLocation == 2)
{
    if (cause == bYes)
    {
        myLocation = 5;
        mAnswer.setText("Try 'Karel the Robot' by R. Pattis.");
    }
    if (cause == bNo)
    {
        myLocation = 6;
        mAnswer.setText("Enjoy A. Biermann's 'Great Ideas in CS'");
    }
}
```

General Decision Trees

- How can we extend this to any size tree?
- Assume we can use yes or no answers all the way through
  - Notice that at each on the tree we have another tree
  - At each step we use code similar to our simple tree
  - Must keep track of where we've been
- For more general branching (not just yes/no) need a bit more
  - Not hard to adapt
  - General ideas the same