Problem Solving with Data Structures using Java: A Multimedia Approach

Chapter 1: Objects for Modeling a World
Today’s story

- What’s the point of this book?
- What’s a model?
- What are data structures?
- Why Java?
- Details on the course
- Getting set up for the course
The Point of this Book

- Real computer-based media developers rarely work in terms of pixels and samples
  - Computer musicians deal in terms of notes, instruments, patches, and other structures.
  - Computer animators deal in terms of characters, movement, scenes, and other structures.
- Bottom-line: They *structure* their media.
Driving Questions of the Book

• How did the wildebeest’s charge over the ridge in *The Lion King*?
• How did the villages wave in *The Hunchback of Notre Dame*?
The Wildebeests in
The Lion King
The Villagers in *The Hunchback of Notre Dame*
The answer: Modeling and Simulation

• These are two of the (rare) times that Disney stepped away from their traditional drawn cel animation.
• Instead they:
  • Modeled the structure of wildebeests and villagers,
  • Modeled the behavior of wildebeests (how they stampede) and villagers (how they wave),
  • Then started a computer *simulation* that *executed* the models…and basically filmed the screen.
What’s “modeling”?

• Describing things in the world in terms of their **structure** and **behavior**.
  - \( F = ma \) (Force=mass \(*\) acceleration) is part of a *model* of the world that describes what happens when one thing hits another.
  - Maps *model* physical spaces and their physical relationships

• On a computer, we can *execute* these models: Make them work, plug values into equations, move things in space, see what happens.
  - That’s *simulation*: Executing a model
What’s a data structure?

• A way of organizing information.
• Different *physical* structures organize space differently.
  • Skyscrapers vs. ranch homes.
  • Trees vs. snail shells
• Data structures organize the information we use in our programs in different ways.
### Data structures you know

<table>
<thead>
<tr>
<th></th>
<th>Age</th>
<th>Eye Color</th>
</tr>
</thead>
<tbody>
<tr>
<td>Matt</td>
<td>13</td>
<td>Brown</td>
</tr>
<tr>
<td>Jenny</td>
<td>7</td>
<td>Blue</td>
</tr>
</tbody>
</table>

[Organizational chart]

[Map of locations]
Data structures that you’ll come to know

Note in Kitchen

Note in Living Room

Note in Study

Note in Bedroom
Data structures have different properties

- Arrays and tables keep things organized right next to one another.
  - Makes it easy to find something in the array or table
  - But if you want to insert something new, you have to move everything over.

- Linked lists and trees keep track of relationships with links (or edges)
  - Easier to insert new things
Thought experiment: Adding a second of silence into a sound

- Assuming that there’s room for another second in the sound…
- We copy samples from the insertion point to the end of sound down one second:
  
  ```
  setSampleValueAt(sound, soundIndex+oneSec, 
  getSampleValueAt(sound, soundIndex))
  ```
- Then we can insert oneSec’s worth of 0’s into the insertion point
How that looks visually

Here is some sound

Here is some sound

One second

00000000000000000000

Here is some sound

One second
Inserting into a table

<table>
<thead>
<tr>
<th></th>
<th>Age</th>
<th>Eye Color</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Matt</td>
<td>13</td>
</tr>
<tr>
<td></td>
<td>Jenny</td>
<td>7</td>
</tr>
<tr>
<td>2</td>
<td>Matt</td>
<td>13</td>
</tr>
<tr>
<td></td>
<td>Jenny</td>
<td>7</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Matt</td>
<td>13</td>
</tr>
<tr>
<td></td>
<td>Jenny</td>
<td>7</td>
</tr>
<tr>
<td>4</td>
<td>Katie</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td>Matt</td>
<td>13</td>
</tr>
<tr>
<td></td>
<td>Jenny</td>
<td>7</td>
</tr>
</tbody>
</table>
Inserting into a linked list

Note in Kitchen

Note in Living Room

Note in Study

Note in Bedroom

Note in Den
Modeling and Simulations are about data structures

- The visual structure of villagers and wildebeests (e.g., how legs and arms attach to bodies) is a *tree* or *graph*.
- Tracking which villager does something next is a *queue*.
- All of the wildebeests to stampede are stored in a *list*.
- The images to be used in making the villagers wave or wildebeests run are usually stored in a *list*. 
Learning objectives in the book

Computer Science Learning Objectives

• Students will be able to program Java classes and methods based on modification.
• Students will learn how to use and manipulate several core data structures: Arrays, linked lists, trees, stacks, and queues.

Media Learning Objectives

• Students will be able to explain the role of data structures in structuring and manipulating data, especially multimedia.
• Students will be able to explain key issues of modern animations, such as sound synchronization and moving objects in layers.
• Students will be able to discuss the properties, strengths, and weaknesses of the different structuring approaches for media.
• Students will be able to design, define, and implement some simulations.
• Students will be able to explain the value of computation for modeling and simulation.
Why are we using Java? (Why aren’t we using Python?)

• Java is faster than Python
  • We can do more operations in less time, so we can do more complicated media in less time.

• Java is more well-known than Python.
  • So there’s more “resume value” than Python.

• If you take more CS, it’ll probably be in Java.
  • More CS classes are being taught now in Java than in other programming languages.
General flow of book

- Introduction to Java
  - Manipulation of pictures and sounds
  - Manipulating music and turtles
- Using arrays, linked lists, and trees
  - With music, pictures, and sounds
  - Creating animations using arrays, lists, and trees
  - Generalized linked lists and trees
- Creating simulations
  - Predator/prey, disease propagation models, movement of people
  - Different kinds of random
    - Sorting our events
  - Simulations with resources
    - Resource queues
  - Creating animations with simulations
Things to do to get started

- Download and install JDK (Java Development Kit)
  - http://www.java.sun.com
- Download and install DrJava
  - http://www.drjava.org
- Download Jmusic
  
  You need jmusic.jar and the inst instrument files
- Download the Java source files for class
  - http://www.mediacomputation.org
- Then, tell Java where to find the JMusic and Java source files.
Open DrJava Preferences
Telling DrJava where to find Jmusic files
Adding in java-source
Everything that has to be there

- Java-source
- Jmusic: inst folder, music.jar
- All jars in java-source
Parts of DrJava

- List of class files that you have open
- Text of your class file (.java)
- Where you interact with Java