Overview

1. Review relevant material.
2. Discuss each milestone.
3. Q&A
Classes

Define your very own variable type!
What custom variables have you already been using?

Hint: anything not a primitive came from a class!

HashMap
ArrayList
GRect
MouseEvent
Integer

GImage
String
SuperKarel

(the wrapper class)
Classes = Blueprints

A class is a blueprint for a custom variable type.
The blueprint must define three things.

What information does this variable store?
What can you do with this variable type?
How do you create this variable type?

Instance variables
Methods
Constructor
Example: `ArrayList<Integer>`

What information does this variable store?
What can you do with this variable type?
How do you create this variable type?

Each element, size
add, indexOf, contains, ...
new ArrayList<Integer>()
ArrayList Instance Variables

What information does this variable store?

```java
private int size;
private int elements[];
```
ArrayList Methods

What can you do with this variable do?

public void add(int element) {...}
public int get(int index) {...}
public boolean contains(int element) {...}
ArrayList Methods

What can you do with this variable do?

```java
public boolean contains(int element) {
    // something cool
}
```
ArrayList Constructor

How do you create this variable type and initialize instance variables?

```
public ArrayList<Integer>() {...}
public ArrayList<Integer>(int capacity) {...}
```

Sidenote: the array stored in the ArrayList start with capacity 10 if you use the default constructor, and resizes if it needs more than 10 elements, but you start with a different capacity, the initial array will have the given capacity.
ArrayList Constructor

How do you create this variable type and initialize instance variables?

```java
public ArrayList<Integer>() {
    size = 0;  // initialize size
    elements = new int[10];  // initialize elements
}
```
ArrayList Used in Our Program

We can now use this custom variable type!

```java
public void run() {
    ArrayList<Integer> scores = new ArrayList<Integer>(); // constructor
    scores.add(100); // method that uses the instance we constructed
}
```
Instances

An instance is one object you created using the blueprint.
Where did we create a new instance of ArrayList?

```java
public void run() {
    ArrayList<Integer> scores = new ArrayList<Integer>() // constructor
        scores.add(100); // method that uses the instance we constructed
}
```
Interactors!

Don’t press the button!
Main Interactors

JButtons
- Click me!
  (click input)

JTextFields
- Type here!
  (text input)

JLabels
- I'm kinda useless. :(
  (not an interactor)
GraphicsProgram Regions

Canvas goes CENTER - usually not where you put interactors
### JButton Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>new JButton(&quot;text&quot;)</code></td>
<td>Creates new button with given text string</td>
</tr>
<tr>
<td><code>jb.getBackground()</code></td>
<td>get or set background color on button</td>
</tr>
<tr>
<td><code>jb.setBackground(color);</code></td>
<td></td>
</tr>
<tr>
<td><code>jb.isEnabled()</code></td>
<td>get or set whether button is clickable</td>
</tr>
<tr>
<td><code>jb.setEnabled(boolean);</code></td>
<td></td>
</tr>
<tr>
<td><code>jb.getFont()</code></td>
<td>get or set text font used for button text</td>
</tr>
<tr>
<td><code>jb.setFont(font);</code></td>
<td></td>
</tr>
<tr>
<td><code>jb.getForeground()</code></td>
<td>get or set text color on button</td>
</tr>
<tr>
<td><code>jb.setForeground(color);</code></td>
<td></td>
</tr>
<tr>
<td><code>jb.getIcon()</code></td>
<td>get or set icon image showing on button</td>
</tr>
<tr>
<td><code>jb.setIcon(icon);</code></td>
<td></td>
</tr>
<tr>
<td><code>jb.getText()</code></td>
<td>set or return text showing on the button</td>
</tr>
<tr>
<td><code>jb.setText(&quot;text&quot;);</code></td>
<td></td>
</tr>
</tbody>
</table>
## JTextField Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>new JTextField(&quot;text&quot;)</td>
<td>Create new text field of given size</td>
</tr>
<tr>
<td>new JTextField(columns)</td>
<td></td>
</tr>
<tr>
<td>jtf.addActionListener(this);</td>
<td>causes action events to occur when the user presses Enter on the field</td>
</tr>
<tr>
<td>jtf.getActionCommand()</td>
<td>set/return a string to identify the action events that will occur in this field</td>
</tr>
<tr>
<td>jtf.setActionCommand(&quot;cmd&quot;);</td>
<td></td>
</tr>
<tr>
<td>jtf.getText();</td>
<td>set/return text in the field</td>
</tr>
<tr>
<td>jtf.setText(&quot;text&quot;);</td>
<td></td>
</tr>
</tbody>
</table>
# JLabel Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>new JLabel(&quot;text&quot;)</code></td>
<td>Create new label with given text</td>
</tr>
<tr>
<td><code>jl.getFont()</code></td>
<td>get/set text font used for label text</td>
</tr>
<tr>
<td><code>jl.setFont(\textit{font});</code></td>
<td></td>
</tr>
<tr>
<td><code>jl.getForeground()</code></td>
<td>get or set text color on label</td>
</tr>
<tr>
<td><code>jl.setForeground(\textit{color});</code></td>
<td></td>
</tr>
<tr>
<td><code>jl.getHorizontalAlignment()</code></td>
<td>set or return horizontal alignment of text in the label; pass <code>JLabel.LEFT, CENTER, or RIGHT</code></td>
</tr>
<tr>
<td><code>jl.setHorizontalAlignment(\textit{align});</code></td>
<td></td>
</tr>
<tr>
<td><code>jl.getText()</code></td>
<td>set/return text in the label</td>
</tr>
<tr>
<td><code>jl.setText(&quot;text&quot;);</code></td>
<td></td>
</tr>
</tbody>
</table>
## Action Event Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>e.getActionCommand()</td>
<td>a text description of the event <em>(e.g. the text of the button clicked)</em></td>
</tr>
<tr>
<td>e.getSource()</td>
<td>the interactor that generated the event</td>
</tr>
</tbody>
</table>
public void init() {
    JButton button = new JButton("Graph");
    add(button, NORTH);
    addActionListeners(); // Listen for all button clicks
}
How to use interactors

```java
public void actionPerformed(ActionEvent e) {
    if (e.getActionCommand().equals("Graph")) {
        // do something
    }
}
```
public void init() {
    JTextField searchBar = new JTextField(TEXT_FIELD_WIDTH);
    searchBar.setActionCommand("SearchBar");
    searchBar.addActionListener(this); // Listen for "ENTER" in text field
    add(searchBar, SOUTH);
}
public void actionPerformed(ActionEvent e) {
    if (e.getActionCommand().equals("SearchBar")) {
        String text = searchBar.getText(); // gets user input.
    }
}
actionPerformed

Called when a button is pressed or textbox is ENTERED.
public void actionPerformed(ActionEvent e) {
    if (e.getActionCommand().equals("SearchBar")) {
        String text = searchBar.getText(); // gets user input.
    } else if (e.getActionCommand().equals("StopButton")) {
        // stop search
    }
}
getActionCommand

Figure out which action was performed by reading the ActionCommand.
NameSurfer

Due Wednesday, March 7, 2018
NameSurfer Assignment

Putting everything together!

1. Data Structures (arrays, ArrayList, HashMap)
2. Classes (multiple files, custom variables)
3. Graphics and Interactors
NameSurfer Assignment

- NameSurfer.java
- NameSurferEntry.java
- NameSurferDataBase.java
- NameSurferGraph.java
- NameSurferConstants.java (provided)
Overview of NameSurfer

Milestone 2+3: Read from a file, store the data of those files into custom variable types that you design.

Milestone 1+4: Setting up the console, interacting with the user.

Milestone 5: When instructed by user, draw the graph. (i.e. a bunch of GLines!)
Milestone 1: Interactors

Add a bunch of buttons and text boxes to the screen.
Milestone 1: Interactors

- Set up all the interactors
- Test to make sure each interactor responds correctly
Milestone 1: Interactors

- JLabel
- JTextBox (Initially blank)
- JButton
- JButton

If JTextBox entered, or JButton “Graph” pressed, should add Graph.

- Name is case-insensitive
- If no data found, don’t add line.
Milestone 1: Interactors (testing)

Change

```
public class NameSurfer extends Program
```

to

```
public class NameSurfer extends ConsoleProgram
```

You can check if your interactors are working!
Milestone 2: NameSurferEntry

Goal: create a custom variable that stores the information in one NameSurfer entry
NameSurferEntry

One line with name and popularity from 1900 to 2000.

eg. “Sam 58 69 99 131 168 236 278 380 467 408 466”

Need to store parse string and store information.
The blueprint must define three things.

What information does this variable store?
What can you do with this variable type?
How do you create this variable type?

Instance variables
Methods
Constructor
Example: NameSurferEntry

What information does this variable store?
What can you do with this variable type?
How do you create this variable type?

getName, getRank, toString
NameSurferEntry(String dataline)
Instance Variables?

What pieces of information make sense as an instance variable?

What type do you use? What collections do you use?

All instance variables should be *private*. 
Constructor to Implement

You are given “Sam 58 69 99 131 168 236 278 380 467 408 466”

```java
public NameSurferEntry(String dataline) {
    // initialize instance variables using dataline
}
```

Relevant Concepts: String processing

Hint: there’s a useful parsing method `str.split(" ").`
Methods to Implement

Dataline was: Sam 58 69 99 131 168 236 278 380 467 408 466

public String getName() {...} // should return “Sam”

public int getRank(int decadesSince1900) {...} // getRank (3) returns 131

public String toString() {...} // should return “Sam [58, 69, ..., 466]”

Relevant Concepts: String processing, data structures
Other Methods?

You may (and probably should) implement more private methods, but may not add/remove/change the headers of any public methods.
Milestone 3: NameSurferDataBase

Goal: store collection of NameSurferEntries
NameSurferDataBase

Text file contains data for one name/one line.

Need to read file and store a collection of NameSurferEntries (each line should be stored as a NameSurferEntry)
The blueprint must define three things.

What information does this variable store?  
What can you do with this variable type?  
How do you create this variable type?

Instance variables  
Methods  
Constructor
Example: NameSurferDataBase

What information does this variable store? ???
What can you do with this variable type?
findEntry(String name)
How do you create this variable type?
NameSurferDataBase(String filename)
Instance Variables?

What pieces of information make sense as an instance variable?

What type do you use? What collections do you use?

All instance variables should be private.
Constructor to Implement

```java
public NameSurferDataBase (String filename) {
    // initialize instance variables using file here
    // probably expect some file reading here
}
```

Relevant Concepts: File reading
Methods to Implement

```java
public NameSurferEntry findEntry(String name) {
  // returns a NameSurferEntry (custom variable you defined!)
}
```

Relevant Concepts: variables, data structures
Other Methods?

You may (and probably should) implement more private methods, but may not add/remove/change the headers of any public methods.
Milestone 4: NameSurferGraph (part 1)
NameSurferGraph

Responsible for graphing the data for each name.

Should store all entries that are currently graphed, so it can redraw if window is resized.
public class NameSurferGraph extends GCanvas

Therefore, you can call methods like getHeight, add, remove.
You can add GLines to a NameSurferGraph!
(see lecture/section for examples)
Step 1: Grid Lines

Set up the Grid Lines.

(this should later be incorporated into the update method)

Relevant Concepts: calculating coordinates.
Important Details

Year 1900
(at left of screen)

Year 2000
(a bit off from right of screen!)
Important Details

Very common bug: Cannot call getHeight() before adding NameSurferGraph to the canvas.

Can use getHeight(), getWidth()
Step 2: Managing NameSurferEntries

Figure out a way to store the NameSurferEntries that should be drawn.

Relevant Concepts: Classes and instances, data structures.
In NameSurfer.java

```
public void init() {
    graph = new NameSurferGraph(); // new instance of NameSurferGraph
    add(graph); // now our graph is on the canvas - can use getHeight()
}

public void run() {
    // something to create NameSurferEntry entry
    graph.add(entry); // does not add the entry to graph yet
    graph.update(); // now it does!
}
```
The blueprint must define three things.

What information does this variable store?
What can you do with this variable type?
How do you create this variable type?

Instance variables
Methods
Constructor
Example: NameSurferGraph

What information does this variable store?
What can you do with this variable type?
How do you create this variable type?

???
Clear, addEntry, update
NameSurferGraph()
Instance Variables?

What pieces of information make sense as an instance variable? What type do you use? What collections do you use?

All instance variables should be *private*. 
Constructor to Implement

```java
public NameSurferGraph () {
    // initialize instance variables
}
```
Methods to Implement

// does not actually clear graph, just deletes the entries.
public void clear(String name) {...}

// does not actually draw graph, just stores the entry.
public void addEntry(NameSurferEntry entry) {...}

// delete all GObjects and reassemble them all
public void update() {...}
The add and clear method should not add/remove any GObjects from the graph.

The update method will remove everything from the graph, then reassemble everything based on the NameEntries stored.

Why? Only update is called when window is resized - must change everything, even the grid lines.
Other Methods?

You may (and probably should) implement more **private** methods, but may not add/remove/change the headers of any **public** methods.
Milestone 5: NameSurferGraph (part 2)

Goal: Finish implementing update(), and make interactors work!
Important Details

Colors cycle: black, red, blue, magenta, repeat.

Rank 0 (not in top 1000): store at bottom (MAX_RANK), with an asterisk (*).
Important Details

Rank 1 at top
(notice the margin)

Rank MAX_RANK
at the bottom
(also notice margin)
Drawing the GLines/GLabels
Finally...

Revisit the interactors from milestone 1, and change the code so the user can type in a name and click the button.
Fully Functioning Program
Common Pitfalls: Off-by-1

Notice there are
11 GLabels
11 decade lines
BUT
10 connecting GLines
Common Pitfalls: getHeight() is zero!

You can only use getHeight() after graph is added to the canvas.

- Can’t use it in NameSurferGraph constructor
  - Can’t use it in NameSurfer init
Common Pitfalls: Another off by 1

The ranks go from 1 to 1000. How many space divisions do we make? (Be careful!)
Common Pitfalls: Not using constants

```java
public interface NameSurferConstants {

    /** The width of the application window */
    public static final int APPLICATION_WIDTH = 800;

    /** The height of the application window */
    public static final int APPLICATION_HEIGHT = 600;

    /** The name of the file containing the data */
    public static final String NAMES_DATA_FILE = "names-data.txt";

    /** The width of the text field in the NORTH of the window */
    public static final int TEXT_FIELD_WIDTH = 16;

    /** The first decade in the database */
    public static final int START_DECADE = 1900;

    /** The number of decades */
    public static final int NDECADES = 11;

    /** The maximum rank in the database */
    public static final int MAX_RANK = 1000;

    /** The number of pixels to reserve at the top and bottom */
    public static final int GRAPH_MARGIN_SIZE = 20;

    /** The number of pixels between the baseline of the decade labels and the bottom of the window */
    public static final int DECADE_LABEL_MARGIN_SIZE = GRAPH_MARGIN_SIZE / 4;
}
```
Common Pitfalls: Changing Method Names

Do not add/remove/change the headers of any public method.

You may (and should) add private methods.
What is good style?

- All the guidelines from Assignment 1-5.
- Separating responsibilities between classes.
- Using instance variables appropriately.
- Choosing appropriate data structures to use.
NameSurfer Overview

**NameSurferDatabase**
- Loads and manages NameSurferEntries

**NameSurfer**
- Main program. Receives user input, reads from the database, and tells the graph what to display.

**NameSurferGraph**
- Graphs NameSurferEntries and keeps track of which entries need to be displayed.

**NamesData.txt**
- Sam 58 69 99 131 168 236 278 380 467 408 466
- Samantha 0 0 0 0 0 0 272 107 26 57
- Samara 0 0 0 0 0 0 0 0 0 886
- Samir 0 0 0 0 0 0 920 0 798
- Sammie 537 545 351 325 333 396 565 772 930 0 0
- Sammy 0 887 544 299 202 262 321 395 575 639 755
- Samson 0 0 0 0 0 0 0 0 0 915
- Samuel 31 41 46 60 61 71 83 61 52 35 28
- Sandi 0 0 0 0 704 864 421 695 0 0 0
- Sandra 0 942 606 50 6 12 11 39 94 168 257
Ideas for Extensions?
Any Questions?
Good luck on NameSurfer!

Have a good night, and see you at the LaIR.