Announcements

- Finish Chapter 6, Sec 2 for next time
- Next assignment handed out after spring break – enjoy your break!
- Pixar Talk – Monday, March 21, 7:30pm
  - Computers Don’t Make Movies
  - LSRC B101

What we will do today

- Lecture on Chap 6, Sec 2 through page 170
  - Execution control with if/else statements and Boolean functions
- Classwork

Thinking about More Advanced Worlds

- How do you build animations like simulations and video games?
- Need to write code that involves decisions

Note: thanks to Wanda Dann and Steve Cooper for slide ideas
Examples of Decisions

- A car-race simulation – the driver steers the car around curves and past mile markers
  - If the car stays on the road the score increases
  - If the car goes off the road into the stands, the car crashes
  - If the driver gets the car over the finish line, the time is posted and the driver wins!

Logical Expressions

- Decision is made based on current conditions.
- Condition is checked in a logical expression that evaluates to true or false (Boolean) value.
  - car on road → true
  - car over finish line → false

If/Else

- In Alice, a logical expression is used as the condition in an If/Else control structure
- Decisions (using If/Else) are used in
  - Functions
  - Methods

Example: Boolean Functions

- Suppose we build a simulation system used to train flight controllers
- One of the tasks of a flight controller is to be alert for possible collisions in flight space
Storyboard

- Two aircraft – biplane and helicopter
- As the biplane moves towards the helicopter we want to make sure they do not collide
- If they are too close, they need to adjust their altitude (height)

Storyboard (cont)

- Two factors in determining whether two aircraft are in danger of collision
  - Total distance between them
  - Vertical distance between them
- Both functions return true if aircraft are too close, otherwise false

isTooCloseByDistance

isTooCloseByDistance:

Parameters: aircraft1, aircraft2, minDistance

If distance between aircraft1 and aircraft2 is less than minDistance
return true
Else
return false

Using a Relational Operator

- Use the < relational operator from the World’s built-in functions to check the distance against the minimum
Implementing the Function

**World.IsTooCloseByDistance**

```plaintext
<table>
<thead>
<tr>
<th>aircraft1</th>
<th>aircraft2</th>
<th>minDistance</th>
</tr>
</thead>
</table>
```

No variables

- If:
  - `aircraft1 < distance to aircraft2 < minDistance`
  - Return `true`

- Else:
  - Return `false`

Return `true`

Vertical Distance Function

- To find the difference in altitude, use the built-in *distance above* function
  - Don’t know which aircraft is above the other
  - To avoid a possible negative value, use *absolute value* of the distance

**istooCloseByVertical**

```plaintext
<table>
<thead>
<tr>
<th>aircraft1</th>
<th>aircraft2</th>
<th>minDistance</th>
</tr>
</thead>
</table>
```

No variables

- If:
  - `absolute value of aircraft1 - distance above aircraft2 - more - < minDistance`
  - Return `true`

- Else:
  - Return `false`

Return `true`

Storyboard

**forwardAndCheckCollision**

Parameters: `aircraft1`, `aircraft2`, `distance`

- `aircraft1` move forward `distance`
  - If `aircraft1` and `aircraft2` are closer than twice `distance`
    - Avoid collision if they are too close heightwise
    - Move `aircraft1` forward twice the `distance`
Implementation and Calling Function

**checkForHeightCollision**

Avoid Collision

Demo and Testing

- Create several events
  - Move plane backward
  - Move plane forward slowly and check for collisions
  - Try with planes at different heights
Classwork today

• Create