Prof. Susan Rodger
Announcements

• Review for test next time.
  – Hand out Test 1 from last semester
    • Should try it before next class
  – Old Quizzes will be available on Blackboard
  – Study classwork and lecture notes

• Next assignment handed out after fall break

• Today – Chap 6, Sec 2
  – Execution control – if/else & Boolean functions
  – Relational operators
  – Logical Operators
Thinking - More Advanced Worlds

• How do you build animations like simulations and video games?
• Need to write code that involves decisions
• Example car-race simulation
  – 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 $\rightarrow$ true
  - car over finish line $\rightarrow$ 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)

• The biplane will move forward a little, check to see if it is close, move forward more, check again, repeating this over and over
Storyboard (cont)

• Two factors in determining whether two aircraft are in danger of collision
  – Total distance between them
  – Vertical distance between them

• We can write functions to determine these

• Both functions return true if aircraft are too close, otherwise false
Methods to write

• World.myFirstMethod
  – Setup, then biplane continuously move forward a little and check

• ForwardAndCheckCollision
  – move biplane forward once, check to see if planes are too close, and if so adjust

• AvoidCollision
  – Moves aircraft up or down if needed

• AdjustForHeightCollision
  – Checks vertical distance and calls AvoidCollision if needed
Functions to write

- `isTooCloseByDistance`
  - Returns true if two objects are too close by distance

- `isTooCloseByVertical`
  - Returns true if the vertical distance between two objects are two close
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

```
T/F World.IsTooCloseByDistance

World.IsTooCloseByDistance (Obj aircraft1, Obj aircraft2, 123 minDistance)

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
forwardAndCheckCollision

Parameters: \textit{aircraft1, aircraft2, distance}

\textit{aircraft1} move forward \textit{distance}

If \textit{aircraft1} and \textit{aircraft2} are closer than twice \textit{distance} avoid collision if they are too close heightwise

move \textit{aircraft1} forward twice the \textit{distance}
Implementation and Calling Function

```plaintext
World.forwardAndCheckCollision( aircraft1, aircraft2, distance )

No variables

If World.IsTooCloseByDistance
    aircraft1 = aircraft1
    aircraft2 = aircraft2
    minDistance = { distance * 2 }

World.adjustForHeightCollision( aircraft1 = aircraft1, aircraft2 = aircraft2, distance = distance )

Else
    Do Nothing
```
adjustForHeightCollision
Avoid Collision

If aircraftOne is above aircraftTwo
- Do together:
  - aircraftOne move up 5 meters
  - aircraftTwo move down 5 meters

Else:
- Do together:
  - aircraftOne move down 5 meters
  - aircraftTwo move up 5 meters
Putting it All Together - Demo

```plaintext
// Run simulation with different heights for helicopter, - up 5, up 10, the same
helicopter move up 5 meters more...
Start! set isShowing to true more...
Camera move backward 25 meters more...
Start! set isShowing to false more...

// run simulation

Do in order
```
Map of interactions – what calls what

myFirstMethod

\[\Rightarrow\]

ForwardAndCheckCollision (method)

\[\Rightarrow\]

isTooCloseByDistance (function)

\[\Rightarrow\]

adjustForHeightCollision (method)

\[\Rightarrow\]

isTooCloseByVertical (function)

\[\Rightarrow\]

avoidCollision (method)
Demo and Testing

• Try helicopter at different heights
  – Move up 5 meters
  – Move up 10 meters
  – Stay the same
  – Down 5 meters
Problem

• The helicopter may go below the ground!

• How do we fix this?
  – Only move down if above a certain distance!
  – Use nested if’s to check more than one condition
Another Way - Logical Operators

• Use Boolean logic operators to check more than one condition
Check

• Where do you get the if?

• Do you have to fill all the parts of the if?
• Where do you find the relational operators?
• Where do you find the logical operators?
Random Numbers

• Skip, We will cover this later
Classwork today

• Write functions and methods with if/else
The next two slides

• Code is equivalent
• First one uses nested if’s (an if inside another if)
• The second one uses logic and nested ifs
avoidCollisionGroundCheck1

```
If aircraftOne is above aircraftTwo
    Do together
    aircraftOne move up 5 meters
    aircraftTwo move down 5 meters
Else
    aircraftOne move up 10 meters
Else
    if aircraftTwo is equal height or above aircraftOne
        Do together
        aircraftOne move down 5 meters
        aircraftTwo move up 5 meters
    else
        aircraftTwo move up 10 meters
```
avoidCollisionGroundCheck2