Chapter 3: Decisions and Loops

Programming with Alice and Java
First Edition

by
John Lewis
and
Peter DePasquale

Copyright © 2009 Pearson Education, Inc. Publishing as Pearson Addison-Wesley

Objectives

- Make decisions using an If/Else statement.
- Base decisions on conditions that use equality, relational operators, and Boolean functions.
- Use logical operators to create complex conditions.
- Nest If/Else statements and loops.
- Execute a set of statements repeatedly using While and Loop statements.
- Use a loop control variable to tailor loop processing.

Copyright © 2009 Pearson Education, Inc. Publishing as Pearson Addison-Wesley

The If/Else Statement

- Control statements allow you to control the flow of a program's logic.
- Control statements are based on the result of a condition.
- The condition produces a Boolean (true or false) value, which determines which statements are executed next.
- An If/Else statement determines which of two sets of statements are executed.

The default structure of an If/Else statement

Copyright © 2009 Pearson Education, Inc. Publishing as Pearson Addison-Wesley

Using the If/Else Statement

The choose true condition of the If/Else statement returns a true or false result determined randomly

Copyright © 2009 Pearson Education, Inc. Publishing as Pearson Addison-Wesley

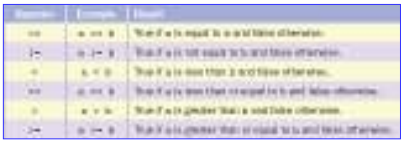
Using the If/Else Statement (continued)

There can be as many statements as needed in either section of If/Else

Copyright © 2009 Pearson Education, Inc. Publishing as Pearson Addison-Wesley

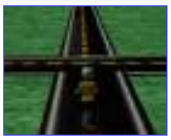
The Equality and Relational Operators

- The following six *equality and relational operators* can be used to compare numeric data.



Copyright © 2009 Pearson Education, Inc. Publishing as Pearson Addison-Wesley

Nested If/Else Statements



- A statement inside an If/Else statement could itself be an If/Else statement – *nested if statement*.

This If/Else is nested within Else section

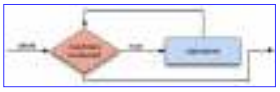
Copyright © 2009 Pearson Education, Inc. Publishing as Pearson Addison-Wesley

Repetition – The While Statement

- *Repetition statements* allow the programmer to repeat one or more statements a number of times.
- Another term for repetition statement is *loop*.
- There are two repetition statements in Alice:
  - the While statement;
  - the Loop statement.
- A While statement is based on a Boolean condition, and repeatedly executes the statement it contains (body of the *loop*) as long as its condition remains true.

Copyright © 2009 Pearson Education, Inc. Publishing as Pearson Addison-Wesley

More about the While Statement



The default form of a While statement

- The condition of a While loop is used to decide if its statements should be executed yet again, and may be evaluated many times.
- The condition in a If/Else statement is evaluated once.
- An *infinite loop* is a loop whose condition never becomes false, and this loop never ends.

Copyright © 2009 Pearson Education, Inc. Publishing as Pearson Addison-Wesley

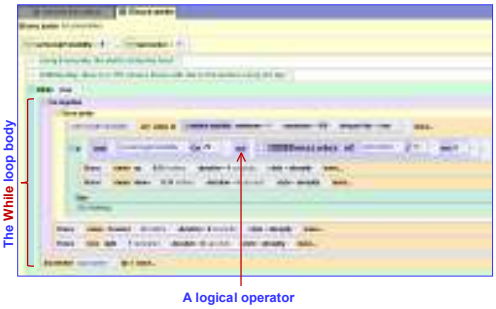
Using the While Statement

This is an example of infinite loop that contains the statements that cause the shark to swim in a circle

The While loop body

Copyright © 2009 Pearson Education, Inc. Publishing as Pearson Addison-Wesley

Using the While Statement (continued)



The While loop body

A logical operator

Copyright © 2009 Pearson Education, Inc. Publishing as Pearson Addison-Wesley

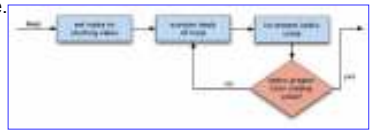
The Logical Operators

Operator	Truth Table			Description
not a	a true false		not a false true	True if a is false, and false if a is true
both a and b	a true true false false	b true false true false	a and b true false false false	True if a and b are both true, and false otherwise
either a or b, or both	a true true false false	b true false true false	a or b true true true false	True if a or b or both are true, and false otherwise

Copyright © 2009 Pearson Education, Inc. Publishing as Pearson Addison-Wesley

The Loop Statement

- The Loop statement allows you to control the exact number of repetitions.
- After dragging a new Loop statement into a method, the number of repetitions needs to be specified.
- The Loop statement uses a condition that tests the value of an integer variable—the *loop control variable (index)*—and terminates when this value reaches a specified end value.



Copyright © 2009 Pearson Education, Inc. Publishing as Pearson Addison-Wesley

Using the Loop Statement

Copyright © 2009 Pearson Education, Inc. Publishing as Pearson Addison-Wesley

Nested Repetition Statements

- Repetition statements can be nested – the body of a loop can contain another loop.
- Each iteration of the outer loop causes complete execution of the inner loop

Copyright © 2009 Pearson Education, Inc. Publishing as Pearson Addison-Wesley

Nested Repetition Example

Copyright © 2009 Pearson Education, Inc. Publishing as Pearson Addison-Wesley

Summary

- Statements that control the flow of a program are based on Boolean conditions (they are either true or false).
- An If/Else statement determines which of two sets of statements are executed.
- The Else portion of an If/Else statement can be left empty.
- Equality and relational operators produce Boolean results.
- An If/Else statement can be part of another If/Else statement.
- A While loop executes its statements until its condition becomes false.
- An infinite loop is a loop whose condition never becomes false.
- Logical operators allow you to construct complex conditions for decisions and loops.
- A Loop statement executes the loop body a specific number of times.
- The alternate version of the Loop statement provides explicit access to the loop control variable
- When loops are nested, the inner loop executes all of its iterations for each iteration of the outer loop.

Copyright © 2009 Pearson Education, Inc. Publishing as Pearson Addison-Wesley