Announcements

- Read the class website DAILY.
- Java programs used in class as examples are in executable form on directory: 
  /usr/local/class/cs111/src/ryder/*
  other subdirectories of .../src contain examples from other lectures
- help@remus - for systems problems only
Announcements

• Printing my lecture notes
  – open up lecture notes in netscape under X windows (this puts you in Adobe Acroread)
  – go to FILE menu in Adobe Acroread program
  – select PRINT command and in box displayed, replace what’s there by:
    mpage -4 -P<printer_name>
    you need to specify the specific printer that you are using by name in this command, i.e.,
    mpage -4 -Pcore3
  – then click on OK button
  – slides will be printed 4-up on a page
Boolean Values

- Program design
- Boolean values
  - Operators &&, ||, !
    - Precedence
    - Use of parentheses
  - Boolean algebra
    - Truth tables
    - DeMorgan’s laws
  - Precedence of operators
- Comparisons of primitive type values
Good Program Design

- Design algorithm before coding
  - Optimize the design, not the code
- How will you test your algorithm?
  - Successful tests cause the program to FAIL!
    (Tests find bugs, not hide them)
  - Test extremal values (e.g., 12 o’clock, minutes values of 00 and 59)
- Use comments and meaningful names
Good Program Design

• Test methods one-by-one, as you write them
  – Create test driver main method
  – Natural modularity of O-O codes

• Use indentation and blank lines to aid understanding of program structure

• Standardize your class definitions
  – Attributes before methods in alphabetic order
  – List constructors first
  – Always add 1 line comment on method functionality
Boolean Type

- **Values**: true, false
- **Operators**: && (and), || (or), !(not)
  - (a && b) true if both a and b are true
  - (a || b) true if either a or b are true
  - !a true if a is false
  - Don’t use bitwise operations: & | ^
- **Used with comparison operators on primitive types** (e.g., z != b)
  
  ==  !=  <  <=  >  >=
**Precedence (highest to lowest)**

`!   -(unary)`

`*   /   %`

`+   -`

`<   <=   >   >=`

`==   !=`

`&&`

`||`
Boolean Operators

Assume $x$ is 1 and $y$ is -1. Then evaluate

$x > y \ or \ x == y \ and \ y > 0$

Repeat: find operator(s) of highest precedence and evaluate.

$x > y \ or \ x == y \ and \ y > 0$

true \ or \ false \ and \ false

true \ or \ false

true.
Boolean Operators

If we guess the precedences incorrectly,

\[ x > y \lor x == y \land \land y > 0 \]

becomes

\[ x > y \lor x == y \land \land y > 0 \]

\[ true \lor false \land \land false \]

\[ true \land false \]

false

Can always use parentheses to insure outcome.

\[ ( x > y \lor x == y ) \land \land y > 0 \]

forces the 2nd evaluation.
## Boolean Operators

Evaluate \(! (x < y) \) || y != x && y == 2*x

\[
\begin{align*}
! \text{false} || \text{true} && \text{false} \\
\text{true} || \text{true} && \text{false} \\
\text{true} || \text{false} \\
\text{true} && \text{false} \\
\text{true} \\
\text{false}
\end{align*}
\]

note difference between != and ! in expression

With wrong precedence yields

\[
\begin{align*}
! \text{false} || \text{true} && \text{false} \\
\text{true} || \text{true} && \text{false} \\
\text{true} && \text{false} \\
\text{false}
\end{align*}
\]
Boolean Operators

- Use parentheses liberally in Boolean expressions to avoid confusion

- **Lazy evaluation** (short circuited evaluation)
  - $(a && x.slowmethod())$ if $a$ is false, then slowmethod never called
  - $(a || x.slowmethod())$ if $a$ is true, then slowmethod never called
  - Used to prevent problems
    
    $(x != 0) && (y == 1/x)$
Equivalent Expressions

• Rules for simplification, given a, b are of boolean type
  – ! (a && b) equivalent to !a || !b
  – ! (a || b) equivalent to !a && !b

• How know these correct?
  – a=T, b=T, !(T && T) = F, ( !T || !T ) = F
  – a=T, b=F, !(T && F) = T, ( !T || !F ) = T
  – a=F, b=T, !(F && T) = T, ( !F || !T ) = T
  – a=F, b=F, !(F && F) = T, ( !F || !F ) = T
### Boolean Identities (full evaluation)

- **Commutative**
  - $a \land b = b \land a$
  - $a \lor b = b \lor a$

- **Associative**
  - $((a \land b) \land c) = (a \land (b \land c))$
  - $((a \lor b) \lor c) = (a \lor (b \lor c))$

- **Absorption**
  - $(a \lor (a \land b)) = a$
  - $(a \land (a \lor b)) = a$
More Identities

• Distributive
  \[ a \land ( b \lor c ) = ( a \land b ) \lor ( a \land c ) \]
  \[ a \lor ( b \land c ) = ( a \lor b ) \land ( a \lor c ) \]

• DeMorgan’s Laws
  \[ \lnot ( a \land b ) = ( \lnot a \lor \lnot b ) \]
  \[ \lnot ( a \lor b ) = ( \lnot a \land \lnot b ) \]
More Identities

- Analogy to min/max
  - $\text{min} \ (a,b) = \text{max} \ (-a, -b)$
  - because if $a < b$ for $a$ and $b$ integers,
    - $\text{min}(a,b)$ is $-a$ and $\text{max}(-a, -b)$ is $-a$
  - since $a < b$ implies $-a > -b$
  - e.g., $\text{min} \ (2, -3) = \text{max} \ (-2, 3) = 3$
Truth Tables - Absorption

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>b</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>a &amp;&amp; b</td>
<td>T</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>a</td>
<td></td>
<td>b</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>!a</td>
<td>F</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>a</td>
<td></td>
<td>(a &amp;&amp; b)</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>a &amp;&amp; (a</td>
<td></td>
<td>b)</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
</tbody>
</table>

Barbara G. Ryder © Spring 1998
## Truth Tables - DeMorgan’s

<table>
<thead>
<tr>
<th>a</th>
<th>T</th>
<th>T</th>
<th>F</th>
<th>F</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>!(a &amp;&amp; b)</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>!a</td>
<td></td>
<td>!b</td>
<td>F</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>!(a</td>
<td></td>
<td>b)</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>!a &amp;&amp; !b</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>T</td>
</tr>
</tbody>
</table>
### Truth Tables - Distributivity

<table>
<thead>
<tr>
<th></th>
<th>T</th>
<th>T</th>
<th>T</th>
<th>T</th>
<th>F</th>
<th>F</th>
<th>F</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>b</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>c</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td>b &amp;&amp; c</td>
<td>T</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>T</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>a</td>
<td></td>
<td>(b&amp;&amp;c)</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td>a</td>
<td></td>
<td>b</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td>a</td>
<td></td>
<td>c</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td>a</td>
<td></td>
<td>b &amp;&amp; a</td>
<td></td>
<td>c</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
</tbody>
</table>