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 || x==y && y > 0

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

x > y || x==y && y > 0

true || false && false

true || false

true.

Boolean Operators

If we guess the precedences incorrectly,

x > y || x==y && y > 0 becomes

x > y || x==y && y > 0

true || false && false

true && false

false

Can always use parentheses to insure outcome.

(x > y || x == y) && y > 0 forces the 2nd evaluation.

Boolean Operators

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

! false || true && false

true || true && false

true || false

true

note difference between != and ! in expression

With wrong precedence yields

!false || true && false

true || true && false

true && false

false

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 && b == b && a

  a || b == b || a

- **Associative**

  ((a && b) && c) == (a && (b && c))

  ((a || b) || c) == (a || (b || c))

- **Absorption**

  (a || (a && b)) == a

  (a && (a || b)) == a

More Identities

- **Distributive**

  a && (b || c) == (a && b) || (a && c)

  a || (b && c) == (a || b) && (a || c)

- **DeMorgan's Laws**

  ! (a && b) == (!a || !b)

  ! (a || b) == (!a && !b)

More Identities

- **Analogy to min/max**

  - min (a,b) == max (-a, -b)

  because if a < b for a and b integers,

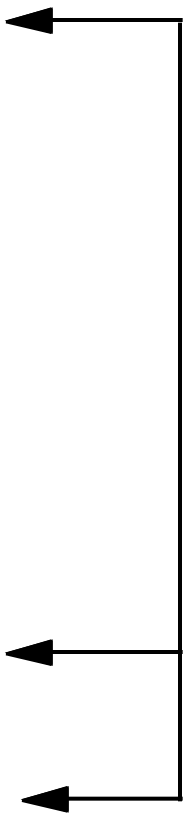
  - min(a,b) is -a and max(-a, -b) is -a

  since a < b implies -a > -b

  e.g., - min (2, -3) == max (-2, 3) == 3

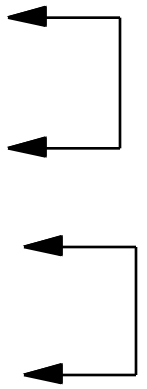
Truth Tables - Absorption

a	T	T	F	F
b	T	F	T	F
a &&b	T	F	F	F
a || b	T	T	T	F
!a	F	F	T	T
a || (a && b)	T	T	F	F
a && (a || b)	T	T	F	F



Truth Tables - DeMorgan's

a	**T**	**T**	**F**	**F**
b	**T**	**F**	**T**	**F**
!(a && b)	**F**	**T**	**T**	**T**
!a || !b	**F**	**T**	**T**	**T**
!(a || b)	**F**	**F**	**F**	**T**
!a && !b	**F**	**F**	**F**	**T**



Truth Tables - Distributivity

a	T	T	T	T	F	F	F	F
b	T	T	F	F	T	T	F	F
c	T	F	T	F	T	F	T	F
b && c	T	F	F	F	T	F	F	F
a || (b&&c)	T	T	T	T	T	F	F	F
a || b	T	T	T	T	T	T	F	F
a || c	T	T	T	T	T	F	T	F
a||b && a||c	T	T	T	T	T	F	F	F