Overview

- Java Programming
- Arrays
Goals

- Understand the basics of Java programming
- Control Statements and Arrays
Assignments

■ Brookshearn: Ch 4, Ch 5 (Read)
■ Read linked documents on these slides (slides will be posted in courseworks)
■ http://java.sun.com/docs/books/tutorial/
Objectives:

- Learn about the **boolean** data type
- Learn the syntax for **if-else** statements
- Learn about relational and logical operators, De Morgan’s laws, short-circuit evaluation
- Learn when to use nested **if-else** statements, **if-else-if** sequences, the **switch** statement
Midterm

- History – Know people and the point of their contributions. Be prepared for short answers (2-3 sentences)
- Hardware – Know the main parts of a computer (processor, memory, etc). Understand that programs and data are *both* information and can be stored in some sort of memory.
- Assembly – given a simple assembly language, write a short (very short) program
- Problem solving – identify a problem with a given algorithm
Midterm

- Networking: TCP vs UDP
- Good/bad design features: be able to name a few usability features
- Modern architecture: privacy, centralization (references on review sheet)
- Programming: given program x, what does it output when run?
- Find the error (not syntax errors; logic errors only)
if-else Statement

if ( <condition> )
{
    < statements >
}
else
{
    < other statements >
}
else clause is optional
**boolean Data Type**

- George Boole (1815 - 1864)
- Unified Logic and Mathematics
- \textit{boolean} variables may have only two values, \texttt{true} or \texttt{false}.
- You define \textit{boolean} fields or \textit{boolean} local variables the same way as other variables:

  ```java
  private boolean hasMiddleName;
  boolean isRolling = false;
  ```
Boolean Expressions

- In `if (<condition> ) <condition> ` is a Boolean expression.
- A Boolean expression evaluates to either true or false.
- Boolean expressions are written using boolean variables and relational and logical operators.
Relational Operators

<, >, <=, >=, ==, !=

is equal to

is NOT equal to
Relational Operators (cont’d)

- Apply to numbers or chars:
  
  ```java
  if ( x <= y ) ...
  if ( a != 0 ) ...
  if ( letter == 'Y' ) ...
  ```

- Do not use `==` or `!=` with `doubles` because they may have rounding errors

```java
double x = 7.0;
double y = 3.5;
if (x / y == 2.0)
...
```

May be false!
Relational Operators (cont’d)

- Be careful using `==` and `!=` with **objects** (e.g., **Strings**): they compare references (addresses) rather than values (the contents)

```java
String cmd = console.readLine();
if ( cmd == "Help" ) ...
```

Wrong!
(always false)
Relational Operators (cont’d)

- Use the `equals` method to compare Strings:

```java
String cmd = console.readLine();
if ( cmd.equals("Help") ) ... 
```

or

```java
if ( "Help".equals(cmd) ) ... 
```
Relational Operators (cont’d)

- Use the == or != operator with strings and other objects when you want to know whether or not this is exactly the same object.

- Also use == or != to compare to null:

```java
String text = file.readLine();
if ( text != null ) ...
```
Logical Operators

&&, ||, !

and

or

not
Logical Operators (cont’d)

- $(condition1 \&\& \ condition2)$ is true if and only if both $condition1$ and $condition2$ are true

- $(condition1 \mid \mid condition2)$ is true if and only if $condition1$ or $condition2$ (or both) are true

- $!condition1$ is true if and only if $condition1$ is false
Logical Operators (cont’d)

- `&&`, `||`, and `!` obey the laws of formal logic called *De Morgan's laws*:

  \[
  \neg (p \land q) = (\neg p \lor \neg q) \\
  \neg (p \lor q) = (\neg p \land \neg q)
  \]

- Example:

  ```
  if ( ! ( x >= -10 && x <= 10 ) ) ... \\
  if ( x < -10 || x > 10 ) ... 
  ```

  Easier to read
Ranks of Operators

Easier to read

if ((year % 4) == 0) && (month == 2) ...

if (year % 4 == 0 && month == 2) ...
if (condition1 && condition2) ...  
If condition1 is false, condition2 is not evaluated  
(the result is false anyway)

if (condition1 || condition2) ...  
If condition1 is true, condition2 is not evaluated  
(the result is true anyway)

if ( x >= 0 && Math.sqrt (x) < 15.0) ...  
Always OK: won’t get to sqrt if x < 0
if ("forward".equals(cmd))
{
    if (slide >= numSlides)
        beep.play();
    else
        slide++;
}
else
{
    if (slide <= 1)
        beep.play();
    else
        slide--;
}
if (drinkSize.equals("Large"))
{
    price += 1.39;
}
else if (drinkSize.equals("Medium"))
{
    price += 1.19;
}
else // if "Small" drink size
{
    price += 0.99;
}
Common `if-else` Errors

```
if (...) {
  statements;
  ...
}
```

- **Extra semicolon**

```
if (...) statement1;
  statement2;
  ...
```

- **Missing braces**

```
if (...) if (...)
  statement1;
else
  statement2;
  ...
```

It is safer to always use braces in `if-else`
The **switch** Statement

```
switch (expression)
{
    case value1:
        ...
        break;
        case value2:
        ...
        ...
        ...
        default:
        ...
        break;
}
```

**Reserved words**

- switch
- case
- default
- break
The *Craps* Applet
Review:

- What are the possible values of a `boolean` variable?
- What operators are used to compare values of numbers and `chars`?
- How can you test whether two `Strings` have the same values?
- Which binary operators have higher rank (are executed first), relational or logical?
Review (cont’d):

- Can you have an if statement without an else?
- What are De Morgan’s laws?
- Explain short-circuit evaluation.
- How long can an if-else-if sequence be?
- What are breaks used for in switch? What happens if you forget one?
- Can the same case in a switch have two breaks?