CS 127
Data structures

- Course policies and procedures
- What we'll cover this term
- Java review

What we'll cover this term

- Efficiency of programs and how our choice of data structures and algorithms dictates this
  - “order-\(n\) notation” or “big-O notation”
- Basic data structures and algorithms of CS
  - Stacks
  - Queues
  - Lists
  - Trees
  - Searching/sorting, ...

Java resources

- Java 1.4.2 API
  http://java.sun.com/j2se/1.4.2/docs/api/
- Java tutorial for C++ programmers
  http://www.cs.wisc.edu/~solomon/cs537/java-tutorial.html
- Javadoc information
  http://java.sun.com/j2se/javadoc/writingdoccomments/
  (Writing Doc Comments > Format of a Doc Comment,
  Descriptions, A Style Guide, Examples of Doc Comments )

Java review

- Java is an object-oriented (OO) language
- Key features of OO languages:
  - Data encapsulation (tight coupling between data and methods)
  - Information hiding
Classes and objects

- Class: defines a type of object
- Object: an instance of a class
- Method: sequence of instructions to perform a given task
- Variable: data value used by the class or object
  - Class variable: shared by all instances of a class
  - Instance variable: specific to one object
- Constructor: special method to create an instance of a class

Visibility modifiers

- **public**: visible to everyone
- **private**: visible only to instances of the class
  - **not** visible to subclasses!
- **protected**: visible to instances of the class and instances of any subclasses
- Default (no label) is *package visibility*
  - visible to any classes within the same package as this class

Control structures

- Conditionals:
  - if-else
  - switch
- Loops:
  - for
  - while
  - do-while

Error handling

- **Done via exception handling**
- **try-catch block**
  ```java
try {
  ...
} catch (NameOfFirstException e1) {
  ...
} catch (NameOfSecondException e2) {
  ...
}
```
Input/Output

• Print to screen:
  System.out.println(...)
• Read from standard in:
  BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
  try {
    String s = br.readLine();
    // optional conversion
    int i = Integer.parseInt(s);
  } catch (IOException e) {...}
  catch (NumberFormatException e) {...}

Memory management

• Handled by the runtime environment
  – garbage collection
• The programmer does not have to handle pointers or other memory tasks
  – Key difference between Java and C++
• Good programming practice to restrict size of arrays, vectors, number of objects created, etc.

Good programming practices

• Comment your code!
  – Use Javadoc-style comments before classes, methods, class-level variables
  – Use comments within methods to describe your logic
• Variables should be declared private
  – Access the data through get and set methods (control access to data)
  – Exception: Class constants may be declared public

Good programming practices

• Class names: start with a capital letter; new words also start with a capital letter
  – e.g. Workbook, DrawingPad, Dialing4Dollars
  – BAD: work book, 2Cool4School
• Method and variable names: start with a lowercase letter; new words start with a capital letter
  – e.g. integrateByParts(), divide(), isValid, base2Number
  – Exception: class constants are typically all upper-case, with words separated by underscores
  – e.g. CONVERSION_FACTOR, PI, MIN_CLASS_SIZE
Java resources

- Java 1.4.2 API
  http://java.sun.com/j2se/1.4.2/docs/api/
- Java tutorial for C++ programmers
  http://www.cs.wisc.edu/~solomon/cs537/java-tutorial.html
- Javadoc information
  http://java.sun.com/j2se/javadoc/writingdoccomments/
    (Writing Doc Comments > Format of a Doc Comment, Descriptions, A Style Guide, Examples of Doc Comments)