

# CSC 221: Computer Programming I

Fall 2006

See online syllabus (also accessible via Blackboard):

<http://www.dave-reed.com/csc221>

Course goals:

- To develop problem solving and programming skills to enable the student to design solutions to non-trivial problems and implement those solutions in Java.
- To master the fundamental programming constructs of Java, including variables, expressions, classes, methods, control structures, and arrays.
- To build a foundation for more advanced programming techniques, including object-oriented design and the use of standard data structures (as taught in CSC 222).

1

## What is programming?

programming is *applied problem-solving*

1. understand a problem
2. identify relevant characteristics
3. design an algorithm (step-by-step sequence of instructions to carry out a task)
4. implement the algorithm as a computer program
5. test the program by repeated (and carefully planned) executions
6. GO BACK AND REPEAT AS NECESSARY

in short: *programming* is the process of designing, writing, testing and debugging algorithms that can be carried out by a computer

we encounter algorithms everyday: directions to dorm, instruction manual, recipe

- people are smart, so spoken languages can be vague
- computers are not smart, so programming languages are extremely picky

2

## Problem-solving example

Sudoku is a popular puzzle craze

given a partially filled in 9x9 grid, place numbers in the grid so that

- each row contains 1..9
- each column contains 1..9
- each 3x3 subsquare contains 1..9

		1			2			
	3	7	8					2
2	4						7	3
4						7	1	
			6	8				
	8	2						9
9	5						3	7
6					5	4	8	
			4			5		

how do people solve these puzzles?

if we wanted to write a program to solve Sudoku puzzles, must/should it use the same strategies?

3

## Object-oriented programming

the dominant approach to software development is *object-oriented design*

- solve problems by modeling real-world objects  
e.g., if designing a Sudoku solver, model board (rows/cols/subsquares), numbers  
e.g., if designing a banking system, model clients, accounts, deposits, ...
- a program is a collection of interacting objects
- emphasizes code reuse (important in industry not to keep reinventing the wheel)

*when designing a program, first focus on the data objects involved, understand and model their interactions*

QUESTION: what is a die?

this course will emphasize object-oriented programming techniques

- utilize the Java programming language (1995, Sun Microsystems)
- we will also use the BlueJ IDE, designed for teaching/visualizing OOP
- good news: **both are free**

4

## Programming is a means to an end

### important point: programming is a tool for solving problems

- computers allow people in many disciplines to solve problems they couldn't solve without them
  - natural sciences, mathematics, medicine, business, ...
- to model this, many exercises will involve writing a program, then using it to collect data & analyze results

**PAPER FOLDING PUZZLE:** if you started with a regular sheet of paper and repeatedly fold it in half, how many folds would it take for the thickness of the paper to reach the sun?

- what information do you need (e.g., distance of sun)?
- what data values do you need to store and update?
- what is the basic algorithm?

5

```
public class PaperSheet
{
    private double thickness; // thickness in inches
    private int numFolds; // the number of folds so far

    /**
     * Constructs the PaperSheet object
     * @param initial the initial thickness (in inches) of the paper
     */
    public PaperSheet(double initial)
    {
        this.thickness = initial;
        this.numFolds = 0;
    }

    /**
     * Folds the sheet, doubling its thickness as a result
     */
    public void fold()
    {
        this.thickness *= 2;
        this.numFolds++;
    }

    /**
     * Repeatedly folds the sheet until the desired thickness is reached
     * @param goalDistance the desired thickness (in inches)
     */
    public void foldUntil(double goalDistance)
    {
        while (this.thickness < goalDistance) {
            this.fold();
        }
    }

    /**
     * Accessor method for determining folds
     * @return the number of times the paper has been folded
     */
    public int getNumFolds()
    {
        return this.numFolds;
    }
}
```

Java solution

note: only the black text is necessary code

(comments, which document the code, are shown here in blue)

6

## Next week...

### review basic computer terminology and history

- hardware vs. software
- generations of computer technology
- evolution of programming



### explore object-oriented concepts using Alice

- a fun, interactive environment for creating animations



### finally, segue into Java programming

- we will go over the details,  
so NO PREVIOUS EXPERIENCE NECESSARY
- classes will mix lecture and hands-on experimentation,  
so be prepared to DO THINGS

