Lectures

- Readings will be posted online together with lecture notes.
- Lecture notes will be online, print them before class and bring them to class.
- Attendance is mandatory.

TF 10:10-11:00 AM, Office 155

Sections

- No permission needed to switch sections.
- Pick one section and attend it.
- Sections may cover material not covered in class. You must show up.
- Each section will be led by someone on staff.
- 10 sections

Section Sheet

http://www.cs.umd.edu/Courses/cs212/2003fa

Overview of CS/ENGR 211

Lecture 1

Course Administrator: Heritage Choir in Uphson 3146

Office hours: TBA online

Consultant in Uphson 304

Know him/her well.

Yr Tu is your main point of contact for the course. Get to

Each Tu will lead one or two recitation sections.

TAs:

Professor David Stewart
Professor Keshav Pingali

Instructor:
CS 211

Strongly advised to take 211 and 212 in same semester

CS 212

1 credit project course

Academic Excellence Workshops

Coursework

7:30pm-10:30pm
Time and place: Uparm 317, September 7/4 (T.R.)

Bootscamp will be taught by Professor Schwartz

Bootscamp

Students with little or no Java knowledge attend Java

method, instance variables

CS 211 assumes basic Java knowledge; classes, objects,
See syllabus and code of academic integrity online.

Indecipherable differences

Strongly discouraged and ignored are penalties in case of

Monogamy is strongly encouraged, polygamy/polyamory is
discouraged.

Finding a partner: post to nextstop or email

You can do them by yourself if you like.

Assignments may be done by teams of two students

Assignments

This is not a course on Java Programming

Software engineering: How to organize large programs

Graphs

Data structures: arrays, lists, stacks, queues, trees, hash-tables

Efficiency of programs

1. Inheritance, Interfaces

2. Classes, objects

1. Recursion, Induction

Core concepts in modern programming languages:

Learn the following:

Objectives of CS 211
Sam Loyd's 8-puzzle

A particular configuration is called a STATE of the puzzle. Each move that leads to a new configuration moves the puzzle or blank 1 step. Given a sequence of moves (NSWE/WSNEW/SEWN/SW), each state is listed and the transition diagram is drawn. The initial and sorted configurations are shown.

Other...

Virtual machine

Game of 8-puzzle

A configuration is a STATE of the puzzle.
Stack operations are used to implement SaM commands.

### SaM Commands

- **STOP**: Terminate execution of program
- **AND**: Boolean values are simulated using 0/1 (false/true)
- **GREATER**: Pushes the integer on stack
- **TIMES**: Subtracts item and pushes result
- **SUB**: Pops two values from top of stack
- **ADD**: Pops two values from top of stack
- **PUSHIMM**: Pushes that integer on stack
  - *some integer*
  - *some integer*

All arithmetic/logical operations pop values from stack.

#### Operations on Stack

- **Pushing 16 on stack**
- **Pop: removes 7 from stack and returns it**

#### Example

```
PUSHIMM 16
```

<table>
<thead>
<tr>
<th>SP</th>
<th>2</th>
<th>16</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>7</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>16</td>
</tr>
</tbody>
</table>
```
Here are two simple SaM programs:

```saM
ADD // should leave 1 on top of stack
  push 7
  push -2
  pop two values from stack (7 and -2)
  add them (5)
  push result

SUB // similar; result would be (7 - 2) = 5
  push 7
  push -2
  pop two values from stack (7 and -2)
  subtract them (5)
  push result
```

Booleans are simulated in SaM with integers:

- True -> 1
- False -> 0

SaM compiles integer values into machines for internal use of SaM.
By the end of CS 211, you will be able to design and write moderately large, well-structured programs to simulate such systems.