Welcome to CS 5 Black!

Professor: Ran Libeskind-Hadas aka “Prof. Ran” (“RON”)
E-mail: ran@cs.hmc.edu
Office: Olin 1253

Three handouts today:
Lecture notes
Syllabus
Worksheet

Official alien of CS 5 Black

Some reviews of CS 5 Black…
• “Of all the courses that we’ve seen, this was definitely one of them!” – Rotten Tomatoes
• “We don’t have strong enough words to describe this course!” – NY Times
Gold and Black

• Same…
  – Overall structure and organization
  – Weekly labs
  – End of semester projects
  – Both prepare you for CS 60

• Different…
  – Emphasis on basic concepts
  – Depth of applications
  – Homework problems (mostly)
  – Grading

How does the workload compare?
CS 5 Gold and Black, One-Stop-Shopping!

CS 5: Welcome!

Homework Assignments and Labs

Lecture Slides

(Before class, the previous term's slides might be posted; shortly after class the current slides will replace them.)

<table>
<thead>
<tr>
<th>Date</th>
<th>Gold</th>
<th>Black</th>
</tr>
</thead>
<tbody>
<tr>
<td>Week 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1/21/20</td>
<td>Lecture 1: Introduction</td>
<td>Lecture 0: Introduction</td>
</tr>
</tbody>
</table>

www.cs.hmc.edu/cs5
Please read the syllabus carefully
(after class)
A few highlights from the

- **Come to class, come to labs**
  - Class attendance is required; if you’re going to miss class, let me know in advance
- **Please no use of electronic devices in class**, but laptops welcome in lab!
- **Grading:**
  - In-class worksheets: 5%
  - Weekly assignments, due Monday nights at 10 PM: 45%
  - Two midterms (12.5% each)
  - Final exam: 25%
- **Late homework policy** (3 Euros in the CS 5 Bank!)
- **Accommodations :^)***
Labs and grutors

- Tuesday afternoon labs: 2:45-4:45, here and B102/105 (Sign in!)
- Evening lab help sessions (totally optional) throughout the week and weekend at 2nd floor of the Linde Activities Center (times TBA on Thursday)
- 11 friendly grutors!
The textbook…

Prof. Ran will give you back the publishers royalties (in dollars or Euros?)
Academic Integrity

• You're encouraged to discuss problems with other students currently in this class – or grutors - or the instructors (Prof. Ran and Prof. Medero)

• You may not share written, electronic or verbal solutions with other students, present or past:
  
  Do not use the internet to (try to) find solutions...

  Do use the internet for Python language reference.

  You may use other's eyes for finding syntax erorrrs (only if they have already submitted their solution).

  If you work as a pair/partners, please read the policy on the syllabus

Sign & submit CS's honesty policy online in this week's lab.
Don’t copy code!

• At the end of the semester, all submissions will automatically be tested using MOSS (comparing against other submissions this semester and in the past as well as materials found online)

• Copying/plagiarism has serious consequences…
  – Failure of the course
  – Likely inability to enroll in future CS courses at Mudd
  – Sanctions from the home college
A note about notes…

• The lecture notes in class will have space for you to write stuff
• The full note sets (including solutions to in-class exercises) will be posted on the course website a few hours after class
Some things *you’ll* do this semester…

- Cryptography
- Graphics
- AI *(Demo!)*
- ... lots more stuff …
- Final project

14 weeks of action-packed adventure!
Picobot!

Goal: whole-environment coverage with only local sensing…

DEMO!
Environment in the NEWS!

Picobot can only sense things directly to the N, E, W, and S

For example, here its surroundings are

Surroundings are always in NEWS order.
Surroundings

How many distinct surroundings are there?

$2^4 = 16$ possible ...

xxxx     Nxxx     xExx     xxWx     xxxS     NExx     NxWx     NxxS

xEWx     xExS     xxWS     NEWx     NExS     NxWS     xEWS     NEWS

(won’t happen)
State

Picobot's memory is a single number, called its state. State is the internal context of computation.

I am in state 0. My surroundings are xxWS.

Picobot always starts in state 0.

State and surroundings represent everything the robot knows about the world.
Picobot moves according to a set of rules:

- **state**
- **surroundings**

If I'm in state **0**
seeing **xxWS**, Then I move **North**, and change to state **0**.

Aha!
I should move **N**.
I should enter state **0**.

A capital “X” here means “Don’t Move”.

I am in state **0**.
My surroundings are **xxWS**.
Wildcards

Asterisks * are wild cards. They match walls or empty space:

I am in state 0. My surroundings are xxWS.

Aha! This matches x***

and EWS may be wall or empty space

N must be empty
**What Will This Set of Rules Do to Picobot?**

<table>
<thead>
<tr>
<th>state</th>
<th>surroundings</th>
<th>direction</th>
<th>new state</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>x***</td>
<td>-&gt;</td>
<td>N</td>
</tr>
<tr>
<td>0</td>
<td>N***</td>
<td>-&gt;</td>
<td>X</td>
</tr>
</tbody>
</table>

Add some code here to make Picobot go up and down in the same column forever!

Picobot checks its rules from the top each time. When it finds a matching rule, that rule runs. Only one rule is allowed per state and surroundings.
This Week!

Write rules that will always cover these two rooms.
(separate sets of rules are encouraged...)

Problem 3

Problem 4

Your “program” can be slow but it should work for any starting location and for any wall-connected maze!

**our best:** 3 states, 7 rules (but Cam Zhou had 6)  
**our best:** 4 states, 8 rules
How about obstacles?

Picobot has 100 states, but the “room” could be arbitrarily big and weird!

Want to try the “pebble” version? Talk to Prof. Ran!
Segue way to Python...

```python
def dbl(x):
    return 2 * x
```

Notice the indentation. This is done using “tab” and Python requires it.

```python
def dbl(myArgument):
    myResult = 2 * myArgument
    return myResult
```
def dbl(x):
    """This function takes a number x and returns 2 * x """
    return 2 * x

Docstrings!

Notice the three leading and ending double quotes (single quotes are OK too, just be sure to be consistent!)
# Doubling program
# Author: Ran Libeskind-Hadas
# Date: January 42, 2020

def dbl(x):
    
    """This function takes a number x and returns 2 * x """
    
    return 2 * x

>>> help(dbl)
Composition of functions

def quad(x):
    return 4 * x

Doubly cool!
def myFunc(x, y):
    """ returns x + 42 * y ""
    return x + 42 * y