Welcome to CS 5!

**Introduction to CS**

*Wally Wart, a protrusive advocate of concrete computing*

*We don't have words strong enough to describe this class.*
  - US News and Course Report

*Everyone will get out of this course – a lot!*
  - NYTimes Review of Courses

*We give this course two thumbs...*
  - Metametacritic

**1 handout... slides & syllabus**
A word on 5 spots…

negotiations afoot… *this section may fill*
Introductions...

Zach Dodds
Olin B163 (HMC)
dodds@cs.hmc.edu

pursuer of low-level AI

taker of low-quality selfies

fan of low-tech games

Speaking of introductions
How I spend my summers ...?

actually, this "I" is not quite accurate...

Robots

Chairs?

Outreach
Algorithmic improvisation
Start-up ideas...
... to formal pitches
CS Staff: rising sophomores, unite!
CS Staff: Rising sophomores, unite!

Lots of opportunities surrounding computing... (at the 5Cs and beyond)
Today in CS5

2) How CS 5 runs...

3) Python?!

1) What is CS?

CS is just programming, right?

Whatever it is, it's definitely alien!

I'm not so sure...

Shouldn't there be an alien in this game?
CS vs. programming?
A minute of cs5 programming...

Python source code, a plain-text file (here, edited by the VS Code text editor)

Demo

lab and hw instructions

get everything running on your own machine
Spot the difference here?

\textcolor{green}{\texttt{print('hi')}} \quad \textcolor{orange}{\texttt{print 'hi'}}

I could not – \textit{for days}!
Spot the difference here?

```
print('hi')  # python 3

print 'hi'  # python 2
```

We will be using python 3 this term...
Spot the difference here?

**Syntax!**

We will be using python 3 this term...
CS != programming

"not equal to"
CS != programming

programming : CS ::

longboards : HMC  maybe 5Cs?
capital : business venture
equations : mathematics
language : ideas
web search : knowledge
Tesla : Google

programs are a vehicle, not the destination
CS != programming

So, what is CS?
What is CS a science of?

the study of complexity:

How can it be done?
How well can it be done?
Can it be done at all?

We'll look at 3 examples – each of which you'll construct in CS 5 ...at least to some extent!

3 examples?
That's it for me!
What is CS?

Can you solve the problem?

Can you create a process to solve such problems?

How can it be done?

How well can it be done?

Can it be done at all?

What is the Longest Common Subsequence between 2 strings?

biology's string-matching problem, "LCS"

'HUMAN'

'CHIMPANZEE'

'CGCTGAGCTAGGCC...'

'ATCCTAGGTAACTG...'

Eye oneder if this haz othur aplications?
What is CS?

How can *it* be done?

How well can *it* be done?

Can *it* be done at all?

How quickly can you find a solution?

Is your solution the "*best*" possible?

*How much work is needed to simulate N stars?*

*chemistry's + physics's "N-body" problem*

what if N is a million-and-one...?
What is CS?

How can it be done?
How well can it be done?
Can it be done at all?

Is your problem solvable?
How can you tell !?

many problems are uncomputable...
... and you'll prove this!

Can we build a 3d model from one 2d image?

Andrew Ng’s "Make3d"

All three eyes tell me that Make3d has just failed ~ epically!
What is CS?

CS is the study of complexity

How can it be done?
How well can it be done?
Can it be done at all?

CS's 6 big questions

But only one is programming. Do you see which?

Can you solve this problem?

Can you create a process to solve such problems?

How quickly can you find solutions?

Do you have the “best” solution?

Is every problem solvable?

Is there a way to tell?
There isn’t always!
What is CS?

CS is the study of **complexity**

How can **it** be done?
How well can **it** be done?
Can **it** be done at all?

**CS's 6 big questions**

But only **one** is **programming**.
Do you see which?

---

Can you solve this problem?
Can you create a process to solve such problems?

How quickly can you find solutions?
Do you have the "best" solution?

Is every problem solvable?
Is there a way to tell?
There isn’t always!
CS’s – and CS5's – philosophy:

*Whatever you are, be a good one.*

- Abraham Lincoln

More and more, CS can help!
Take-home message...

CS 5: Welcome!

Homework Assignments and Labs

<table>
<thead>
<tr>
<th>Labs</th>
<th>GOLD</th>
<th>BLACK</th>
</tr>
</thead>
<tbody>
<tr>
<td>Week 0</td>
<td>Lab 0</td>
<td>Homework 0</td>
</tr>
</tbody>
</table>

Lecture Slides

<table>
<thead>
<tr>
<th>Week 0</th>
<th>GOLD</th>
<th>BLACK</th>
</tr>
</thead>
<tbody>
<tr>
<td>10/30/16</td>
<td>Lecture 0: Introduction</td>
<td>Lecture 0: Introduction</td>
</tr>
<tr>
<td>9/1/16</td>
<td>Lecture 1: Pico-fun!</td>
<td>Lecture 1: Map and Reduce</td>
</tr>
</tbody>
</table>

Acknowledgments and thanks...

Yay! in 2017
Just Google for hmc cs5

www.cs.hmc.edu/cs5
You're here ~ what's next?

2) How CS 5 runs...

3) Python?!

the first Python HW is *choice*!

Shouldn't there be an alien in this game?

1) What *is* CS?

CS is just programming, right?

I'm not so sure...

Whatever it is, it's definitely *alien*!
Logically, I've got game!

http://www.youtube.com/watch?v=fqJDc2VICZ0 start at about :28

Let's play! Maybe two out of three?
Soundbite Syllabus

**Lectures**

T and Th: 8:10-9:25 am

Key takeaways: theory, algorithms, data structures

Insight into the HW problems (what, why, how)

We'd like to see you! Let us know if you'll be sick...

**Lab**

T or W: 2:45 - 4:45pm or 6-8 pm

Guided progress on the week's hw

Not required, but recommended by 4 out of 5 CS5 alums!

Come to Labs!

Will SAVE you time and effort in CS 5

**Office hrs**

F: 3-6:00 pm - Linda Activities Center lab

or, come to any of the many tutoring hrs!

Lots of help is available!

**HW**

Monday nights: due by 11:59 pm

Hw is due on Monday nights...
Syllabus, briefly

Lectures

T and Th: 8:10-9:25 am
Key skills, topics, and their motivation
Insight into the HW problems (what, why, how)
We’d like to see you! Let me know if you’ll be sick...

Lab

T or W: 2:45 - 4:45pm or 6-8 pm
Guided progress on the week's hw
Not required, but encouraged: full credit for lab
Will SAVE you time and effort in CS 5

Office hrs

F: 2:30-4:30 pm, Linde Activities Center lab
feel free to work on HW, to just stop by,
or, come to any of the many tutoring hrs!

HW

Monday nights: due by 11:59 pm
Each week's lab...

0) Find the lab! *Sign in...*

1) Get Python running...

2) Edit, run, + submit a file...

Encouraged: *bring your laptop*
Each week's lab...

Labs are **optional**, but *incentivized*.

If you come to lab, give a good-faith effort, and sign in, you'll receive **full credit for the lab problems** even if you don't finish

*(you do need to submit by the usual hwk due date)*

Encouraged: *bring your laptop*
Evening lab? Enter through Olin building through the SE door to Beckman B102, B105, B126. Olin's Southeast door is open! We're here!
Edwards Macalister Pryne

coffee

cool machines - drills, lathes, etc.

other keyboard-free machines

Physicists, chemists & other parenthesis-needing individuals,

CS Hallway and Labs

B102

B100

B105

Map to CS Lab

Galileo

Beckman

Biologists, bees, spiders and other arachnophiles

Laptop? Bring it!

Shan

coffee
Submissions: *GradeScope*

![GradeScope screenshot](https://www.gradescope.com)

**MPVY6M**

**course code**

Happy Instructors at Over 300 Schools
This week: Lab 0

- Running a file!
  - To run your file, go back over to the terminal.
  - Type `ipthon` if you're not yet running it.
  - Type `ls` (windows or mac) to see the files in the current directory.
  - Make sure your `hw0pr1.py` file is there!
    - If not, use `cd ..` or `cd Desktop` or other combinations to get to the correct directory. Ask for help!
  - At the ipthon prompt, type `run hw0pr1.py`
  - This should run the file `hw0pr1.py`
  - If all goes well, the program should run and you should see the output
  - If not, please ask!
  - Now, you can edit your file, save it, and hit up-arrow to re-run it. Awesome!

**Your task: four fours**

- The four fours challenge! Now, add several more lines similar to this one so that you compute 16 of the 21 values from 0 through 20 using exactly four fours. You should use Python's arithmetic operations:
  - `+` addition
  - `-` subtraction or negation
  - `*` multiplication
  - `/` division
  - `( )` parentheses for grouping
  - `**` power
- You may also use `44` or `4.4`, which count as two fours,
- or `.4`, which counts as one four.
- See help for two more allowable operations, `math` and `factorial` both in the `math` library.
- See help for two more allowable operations, `math` and `factorial` both in the `math` library.
- See help for two more allowable operations, `math` and `factorial` both in the `math` library.
- See help for two more allowable operations, `math` and `factorial` both in the `math` library.
- You need only 16 of the 21:

```
In [1]:
In [2]:
In [3]:
In [4]:
In [5]:
```

get everything running **on your own machine**

Python source code, a plain-text file
(here, edited by the VS Code text editor)

shell or command-line or terminal
(the execution environment)
Homework

Assignments  ~ 5 problems/week

Due **Monday** evenings by 11:59 pm.

Extra credit is usually available...

You have 3  **CS 5 Euros** to use...
"Late Days"

**Eur-allowed to use one Euro for up to three hwks.**
No need to let us know, even.

Collaborate!

Some problems are specified “individual-only.”
Others offer the option of working as pairs/partners:

- You don't have to work in pairs/partners (that said, it's fun!)
- If you do, you must share the work equally - typing and coaching
- Be sure to indicate who your partner was at the submission site!
Pairs

one computer
tradeoff typing/debugging ~
about every 20 minutes

Partners

two computers
both partners type/debug ~
provide help as needed

Standard is the same either way:
After finishing the hw, (a) each person has contributed equally and (b) both could complete the problems on their own

Submit with a partner as full co-owners of the work.
Honor Code

• You're *encouraged* to **discuss** problems with other students – or tutors - or any instructors.

• You may **not** share written, electronic or verbal solutions with other students, present or past:

  Please *do* use the internet for Python language references.

  Pleas *do* use other's eyes for finding syntax erorrs.

  Do *not* use the internet (or intranet) to (try to) find solutions...

  If you work as a pair/partners, the rules apply for the duo.

**Sign & submit** CS's honesty policy **online** in this week's lab.
Grading

~ 65% Assignments

~ 30% Exams

~ 5% Participation/“quizzes”

Exams

Midterm
Th, Nov. 2, in-class

Final
W or Th, Dec 13th (7pm) or 14th (2pm)

using a page of notes is OK on exams

the exams are written, not coded

the problems are modeled on the in-class "quizzes"

if perc > .95:
    print('A')
elif perc > .90:
    print('A-')
elif perc > .70:
    print('Pass')

Midterm? This feels more like a 2/3-term!

many take cs5 P/NC

see online syllabus for the full grade list...
Choices, choices!

Let's set the value of `perc` to 0.91...

```python
def set_perc_value(perc):
    perc = 0.91
    if perc > 0.95:
        print('A')
    elif perc > 0.90:
        print('A-')
    elif perc > 0.70:
        print('Pass')
    else:
        print('Aargh!')
```

What's here?

First – who sees the syntax errors here!?
Choices, choices!

Let's set the value of `perc` to 0.91...

\[ \text{perc} = 0.91 \]

```python
if perc > 0.95:
    print('A')
elif perc > 0.90:
    print('A-')
elif perc > 0.70:
    print('Pass')
else:
    print('Aargh!')
```

What will this program print, if `perc` is 0.91?

Aargh! ;-)
Let's set the value of `perc` to 0.91...

\[
\downarrow
\]

```python
perc = 0.91

if perc > 0.95:
    print('A')
elif perc > 0.90:
    print('A-')
elif perc > 0.70:
    print('Pass')
else:
    print('Aargh!')
```

What will this program print, if `perc` is 0.91?
Choices, choices!

perc = 0.80

if perc > 0.95:
    print('A')
elif perc > 0.90:
    print('A-')
elif perc > 0.70:
    print('Pass')
else:
    print('Aargh!')

perc = 0.80

if perc > 0.00:
    print('Aargh!')
elif perc > 0.70:
    print('Pass')
elif perc > 0.90:
    print('A-')
else:
    print('A')

What does each of these programs print out, if \texttt{perc} is 0.8? What value of \texttt{perc} gives an \texttt{A-} on the right? How can you get a \texttt{better} grade on the right than the left?
Exclusive Choices

\[
\begin{align*}
\text{if } \text{perc} & > 0.95: \\
& \quad \text{print('A')} \\
\text{elif } \text{perc} & > 0.90: \\
& \quad \text{print('A-')} \\
\text{elif } \text{perc} & > 0.70: \\
& \quad \text{print('Pass')} \\
\text{else:} \\
& \quad \text{print('Aargh!')} \\
\end{align*}
\]

4 mutually exclusive blocks in a single control structure

When using \text{if} . \text{elif} . . . . \text{else} at most one block will run: the first whose test is \text{True}. If all fail, the \text{else} will run

\text{elif} and \text{else} are optional
Exclusive Choices

Every **if** starts a new control structure.

Every **elif** and **else** continues an existing control structure. The first whose test is **True**. If all fail, the **else** will run.

```python
if perc > 0.95:
    print('A')
elif perc > 0.90:
    print('A-')
elif perc > 0.70:
    print('Pass')
else:
    print('Aargh!')
```
What's the difference?

**mutually exclusive blocks**

```
perc

if perc > .95:
    print('A')

elif perc > .90:
    print('A-')

elif perc > .70:
    print('Pass')
```

**non exclusive blocks**

```
perc

if perc > .95:
    print('A')

if perc > .90:
    print('A-')

if perc > .70:
    print('Pass')
```

What if `perc == .99`? (How would we set it?)

How many separate **control structures** does each side have?
What's the difference?

**mutually exclusive blocks**

```python
perc

if perc > .95:
    print('A')
elif perc > .90:
    print('A-')
elif perc > .70:
    print('Pass')
```

**non exclusive blocks**

```python
perc

if perc > .95:
    print('A')

if perc > .90:
    print('A-')

if perc > .70:
    print('Pass')
```

What if `perc == .99`? (How would we set it?)

How many separate *control structures* does each side have?
Nesting for decision-making, we now have it all...
Nesting for decision-making, we now have it all...
Nesting for decision-making, we now have it all...

So, let's catch 'em all...
comp = 'rock'
user = 'paper'

if comp == 'paper' and user == 'paper':
    print('We tie. Try again?')

elif comp == 'rock':

    if user == 'scissors':
        print('I win! *_*')
    else:
        print('You win. Aargh!')

# Blocks ?
# Tests ?
# C. Structures ?
"Quiz"

Pair up with someone nearby – answer these questions together...

Name ______________________
Your favorite __________ is ____________.
Your least favorite __________ is ____________.

Name ______________________
Your favorite __________ is ____________.
Your least favorite __________ is ____________.

What is something non-Claremont-collegey you have in common?

Then, try these Python q's:

(0) Find the 3 tests and 4 blocks here.

(1) What does this code print?

```python
comp = 'rock'
user = 'rock'

if comp == 'rock':
    if user == 'paper':
        print('I win *_*!')
    if user == 'scissors':
        print('You win.')
else:
    print('Tie.')
```

(2) As written, what output does this print?

```python
comp = 'rock'
user = 'rock'

if comp == 'rock':
    print('I win *_*!')
if user == 'paper':
    print('You win.')
else:
    print('Tie.')
```

(3) Change these inputs to produce a completely correct RPS output here.

(4) How many of the 9 RPS input cases are fully correctly handled here?

(5) What is the smallest number of blocks and tests you'd need for a full game of RPS?

(Extra) What if it were RPS-5, which includes Lizard and Spock? How about RPS-101?
Pair up with someone nearby – answer these questions together...

Name ______________________
Your favorite __________ is ____________.
Your least favorite __________ is ____________.

Name ______________________
Your favorite __________ is ____________.
Your least favorite __________ is ____________.

What is something non-Claremont-collegey you have in common?

Then, try these Python q's:

(0) Find the 3 tests and 4 blocks here.

(1) What does this code print?

```python
comp = 'rock'
user = 'rock'
if comp == 'rock':
    if user == 'paper':
        print('I win *_!*!')
    elif user == 'scissors':
        print('You win.')
    else:
        print('Tie.')
else:
    print('Tie.')
```

(2) As written, what output does this print?

(3) **Change** these inputs to produce a completely correct RPS output here.

(4) How many of the 9 RPS input cases are fully correctly handled here?

(5) What is the **smallest** number of blocks and tests you'd need for a full game of RPS?

(Extra) What if it were RPS-5, which includes Lizard and Spock? How about RPS-101?
Pair up with someone nearby – answer these questions together...

Name ______________________

Your favorite __________ is ____________.

Your least favorite ____________ is ____________.

Name ______________________

Your favorite __________ is ____________.

Your least favorite ____________ is ____________.

What is something non-Claremont-collegey you have in common?

Then, try these Python q’s:

(0) Find the 3 tests and 4 blocks here.

(1) What does this code print?

```python
comp = 'rock'
user = 'rock'

if comp == 'rock':
    if user == 'paper':
        print('I win *_*!')
    elif user == 'scissors':
        print('You win.')
else:
    print('Tie.')
```

(2) As written, what output does this print?

```python
comp = 'rock'
user = 'rock'

if comp == 'rock':
    print('I win *_*!')
if user == 'paper':
    print('You win.')
else:
    print('Tie.')
```

(3) Change these inputs to produce a completely correct RPS output here.

(4) How many of the 9 RPS input cases are fully correctly handled here?

(5) What is the smallest number of blocks and tests you’d need for a full game of RPS?

(Extra) What if it were RPS-5, which includes Lizard and Spock? How about RPS-101?
• Name  Zach Dodds
  • Your favorite **tv show** is **Modern Family** + Dr. Who
  • Your least favorite **coffee** is **decaffeinated**

• Name  T. E. Alien
  • Your favorite **canned-meat food product** is **spam**
  • Your least favorite **#** is **41.999**

Something in common?  Our taste in hats!
Pair up with someone nearby – answer these questions together...

Name ______________________
Your favorite __________ is ____________.
Your least favorite ____________ is ____________.

Name    ______________________
Your favorite __________ is ____________.
Your least favorite ____________ is ____________.

What is something non-Claremont-collegey you have in common?

The computer is your opponent in this RPS game.

```python
comp = 'rock'
user = 'rock'
if comp == 'rock':  
    if user == 'paper':  
        print('I win *_*!')  
    else:  
        print('Tie.')  
else:  
    print('The computer wins!')
```

(1) What does this code print?

(2) As written, what output does this print?

(3) Change these inputs to produce a completely correct RPS output here.

(4) How many of the 9 RPS input cases are fully correctly handled here?

(5) What is the smallest number of blocks and tests you'd need for a full game of RPS?

(Extra) What if it were RPS-5, which includes Lizard and Spock? How about RPS-101?
comp = 'rock'
user = 'rock'

if comp == 'rock':
    if user == 'paper':
        print('I win *_*!*')
    elif user == 'scissors':
        print('You win.')
else:
    print('Tie.')
    print('Ties go to the runner.')
    print(' - and I am running!')

"Quiz" ~ problems 1+2

... what if this else block were indented?
What does this program print?

```python
comp = 'rock'
user = 'rock'

if comp == 'rock':
    print('I win *_!*!')

if user == 'paper':
    print('You win.')

else:
    print('An awful tie')
```

"Quiz" ~ problems 3-5
```
comp = 'rock'
user = 'rock'

if comp == 'rock':
    print('I win *_*!')

if user == 'paper':
    print('You win.')

else:
    print('An awful tie')
```
comp = 'rock'
user = 'rock'

if comp == 'rock':
    print 'I win *_*!'

if user == 'paper':
    print 'You win.'
else:
    print 'An awful tie'

A correct RPS is possible with only if ... elif ... else!

How many possible "input cases" are there?

How efficient can we be?
For how many is this program correct?
For RPS-3? RPS-5? RPS-101?
Remember ~ **Lab this week**

Tue. or Wed. ~ afternoon or evening  
Bring your laptop to Beckman B126 (here)  
- *or* use one of the CS machines in B105  
Get started with Python/text editor/cmdline...

---

**See you in lab!**  
*perhaps at 2:44:44 today...?*

though it's more than a few bits early!

---

Alien defeats everything –  
*even Alien*

How about a sneak peek at this week's HW... ?  
... you must mean sneak *Pic*!