Welcome to CS 5!

xkcd, CS's id
Welcome to CS 5!

Introduction to CS

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

Wally Wart, a protrusive advocate of concrete computing
A word on 5 spots...

Welcome, not only to HMC, but to all 5Cs!
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 ...?

Robots

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

Chairs?

Outreach

Who?!??  Dinos!
Start-up ideas...
... to formal pitches
CS Staff: *Rising sophomores, unite!*

Setting up labs ...
CS Staff: *Rising sophomores, unite!*

**Teacher Outreach in S.F.**
CS Staff:  *Rising sophomores, unite!*

Where is this?
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...

Lab 0: getting everything running on your own machine

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

Demo

lab and hw instructions

shell or command-line or terminal (the execution environment)
Lab 0: *Happiness Suggestion*

Download the software BEFORE coming to lab:

https://www.cs.hmc.edu/twiki/bin/view/CS5/OwnMachines

<table>
<thead>
<tr>
<th>Homework Assignments and Labs</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Week 0</strong></td>
</tr>
<tr>
<td>Lab 0</td>
</tr>
</tbody>
</table>
Spot the difference here?

\texttt{print('hi')} \hspace{1cm} \texttt{print 'hi'}

I still confuse these!
Spot the difference here?

```
print('hi')
```

```
print 'hi'
```

python 3

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

```python
print('hi')

print 'hi'
```

**Syntax!**

We'll 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**, but 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?

it ~ information

or, more precisely, a process transforming information from one form to another

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?

How can it be done?

How well can it be done?

Can it be done at all?

Can you solve the problem?

Can you create a process to solve such problems?

What is the Longest Common Subsequence between 2 strings?

biology's string-matching problem, "LCS"

'CGCTGAGCTAGGCC...'

'ATCCTAGGTAACTG...'

+10^9 more

'HUMAN'

'CHIMPANZEE'

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?

Is your problem solvable?
How can you tell!?

Can it be done at all?

How can it be done?
How well can it be done?

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

Andrew Ng’s "Make3d"

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

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?

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 6 big questions

Only one is programming. Which one?
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

Only *one* is programming. *Which one?*

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...

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

2) How CS 5 runs...

3) Python?!

the first Python HW is *choice*!

1) What *is* CS?

Whatever it is, it's definitely *alien*!

CS is just programming, right?

I'm not so sure...

Shouldn't there be an alien in this game?
Logically, I've got game!

rock – paper – scissors – lizard – Spock!

Let's play! Maybe two out of three?

Logically, I've got game!

http://www.youtube.com/watch?v=fq1Dc2V1CZ0 start at about :28
Soundbite Syllabus

Lectures

**T and Th:** 8:10-9:25 am
Keep track of due dates and their respective dates.
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

For 3-4pm, Linda Activities Center, lab
Or, come to any of the many tutoring hrs!

HW

Monday HW is due on Monday nights...

Lots of help is available!

Come to Lectures!

Come to Labs!
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?

Olin's Southeast door is open!

Enter through Olin building through the SE door to Beckman B102, B105, B126
Submissions: GradeScope

MRZZ4Y course code
This week: Lab 0

getting everything running on your own machine

lab and hw instructions

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

shell or command-line or terminal
(the execution environment)
Lab 0: *Happiness Suggestion*

Download the software BEFORE coming to lab:

https://www.cs.hmc.edu/twiki/bin/view/CS5/OwnMachines
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-o**llowed 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 errors.

  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. 8, in-class
        Final  Tue. Dec 17th (7pm) or Wed. 18th (9am)

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')

many take cs5 P/NC

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

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

```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?

First – who sees the **syntax errors** here !?
Let's set the value of `perc` to 0.91...

```
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?

Aargh! ;-)

What's here?

- # of BLOCKS here:
- # of TESTS here:
- # of CONTROL STRUCTURES here:

Lots of Illuminating Solid Parentheses!
Choices, choices!

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

```
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!')

What does each of these programs print out, if perc is 0.8?

What value of perc gives an 'A-' on the right?

How can you get a better grade on the right than the left?
Exclusive Choices

if ... elif ... else

```python
if perc > 0.95:
    print('A')

elif perc > 0.90:
    print('A-')

elif perc > 0.70:
    print('Pass')

else:
    print('Aargh!')
```

4 mutually exclusive blocks
in a single control structure

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

elif and 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')
```

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

**nonexclusive blocks**

```python
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...
Nesting

comp = 'rock'
user = 'paper'

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

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

Does this program print the correct RPS result this time? Does it always?
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?
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 *_*!')
    else:
        print('Tie.')
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':
    if user == 'paper':
        print('I win *_*!')
    else:
        print('Tie.')
elif user == 'scissors':
    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 ______________________  Name ______________________
Your favorite __________ is ____________.  Your favorite __________ is ____________.
Your least 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?
• 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 ____________.

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

comp = 'rock'
user = 'rock'

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

(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?

(0) Find the 3 tests and 4 blocks here. Then, try these Python q's:

Please pass these up the aisles...

(take a picture, if you'd like)

... then, turn back to the notes
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!')

... 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
How many possible “input cases” are there?
For how many is this program correct?

How efficient can we be?
For RPS-3? RPS-5? RPS-101?
How many possible “input cases” are there?
For how many is this program correct?

How efficient can we be?
For RPS-3? RPS-5? RPS-101?
"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'
```

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/B102
Get started with Python/text editor/cmdline...

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

Alien defeats everything –
even Alien

How about a sneak peek at this week's HW... ?

... you must mean sneak *Pic* !