Welcome to CS 5!

xkcd, CS's id
Welcome to CS 5!

Grab these lecture notes...

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

Julie Medero
Olin 1269 (HMC)
julie@cs.hmc.edu

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

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

Sustainability

Reading

Outreach
Sustainability

Traffic and Transportation Commission
The Traffic and Transportation Commission reviews and comments on the traffic impacts of major development proposals. This includes traffic circulation plans, public transportation, and similar items referred to the commission by the City Council or other commissions.

Meeting Schedule:
Fourth Thursday of each month, 7 p.m.
City Council Chamber, 225 West Second Street, Claremont

Members (year after name indicates end of current term):
Jack Blair (2020); Zachary Courser (2019); Justine Garcia (2022), Julie Medero, Chair (2020); Robert Miletich, Vice Chair (2022); Evan Rutter (2020).
There was an old woman who lived in a shoe. She had so many children, and loved them all, too.

Pick a word:
- Big
- Highlight
- Shadow
- Small
- Underline
- Color

Go!
Outreach
How we spend our summers!

Learn more:
Thursday @ 4:15 in Shan 1430!
Today in CS5

2) How CS 5 runs...

3) Python?!

1) What is CS?

CS is just programming, right?

I'm not so sure...

Whatever it is, it's definitely alien!
CS vs. programming?
A *minute* of cs5 programming...

Lab 1: getting 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)

lab and hw instructions
Lab 1: *Happiness Suggestion*

Download the software BEFORE coming to lab:

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

**Schedule**

(Slides will be uploaded shortly after class.)

<table>
<thead>
<tr>
<th>Lecture</th>
<th>Lab</th>
<th>Homework</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Week 1</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1/22/19: Introduction (Section 1) (Section 2)</td>
<td>Lab 1</td>
<td>Homework 1</td>
</tr>
<tr>
<td>1/24/19: Pico-fun! (Section 1) (Section 2)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Spot the difference here?

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

I \textit{still} confuse these!
Spot the difference here?

We'll be using python 3 this term...

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

```
print 'hi'
```
Spot the difference here?

```
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?*

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...

3 examples?

That's *it* for me!

...at least to some extent!
What is CS?

Can you solve the problem?

How can it be done?

How well can it be done?

Can it be done at all?

Can you create a process to solve such problems?

What is the Longest Common Subsequence between 2 strings?

biology's string-matching problem, "LCS"

'HUMAN'

'CHIMPANZEE'

'CGCTGAGCTAGGGCC...'

'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!?

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
1) What *is* CS?

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

2) How CS 5 runs...

3) Python?!

the first Python HW is *choice*!

Shouldn't there be an alien in this game?

CS is just programming, right?

I'm not so sure...
Logically, I've got game!

rock – paper – scissors – lizard – Spock!

Let's play! Maybe two out of three?

http://www.youtube.com/watch?v=fqIdc2YICZD start at about :28
Soundbite Syllabus

Lectures

**T and Th:** 8:10-9:25 am or 9:35-10:50 am

Key skills, topics, and their motivation (why, how)

Come to Lectures!

Lab

**T:** 1:10-3:10 pm or 3:15-5:15 pm

Will SAVE you time and effort in CS 5

Come to Labs!

Office hrs

Lots of help is available!

or, come to any of the many tutoring hrs!

HW

Hw is due on Monday nights...
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 usualhwk due date)

Encouraged: bring your laptop
Finding lab?

Enter through Olin building through the SE door to Beckman B102 & B105.
Laptop? *Bring it!*
Diversity of Backgrounds in Spring CS5

Key points:

• I want *everyone* to grow as a computer scientist!

• Talk to me first!

• You “owe” CS 150 minutes/week

• Lots of flexibility to make the semester work for you.
This week: Lab 1

- **Running a file!**
  - To run your file, go back over to the terminal.
  - Type `python` 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 ipython prompt, type `run hw0pr1` (tab completion will work)
  - 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 `4.0`, which counts as one four.
- See below for two more allowable operations, `sqrt` and `factorial` both in the math library
- `21` is so that you can choose a few to skip!
- Please write the results, but not the source code, will look like.
- I need only 16 of the 21:

**shell or command-line or terminal**

(Here, edited by the VS Code text editor)

Python source code, a plain-text file

(Here, edited by the VS Code text editor)

lab and hw instructions

getting everything running **on your own machine**
Lab 1: *Happiness Suggestion*

Download the software BEFORE coming to lab:

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

<table>
<thead>
<tr>
<th>Schedule</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Slides will be uploaded shortly after class.)</td>
</tr>
<tr>
<td>Lecture</td>
</tr>
<tr>
<td>----------</td>
</tr>
<tr>
<td><strong>Week 1</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>Week 2</strong></td>
</tr>
</tbody>
</table>
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-ollowed 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.
  
  Please *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”

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

see online syllabus for the full grade list...

Exams
Midterm  Tues., Apr. 2, in-class
Final  Tues.  May 14th, 2pm or 7pm

using a page of notes is OK on exams
the exams are written, not coded
the problems are modeled on the in-class "quizzes"

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...

```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?

Aargh! ;-)
Choices, choices!

Let's set the value of \texttt{perc} to 0.91...

\[
\downarrow
\]

\texttt{perc} = 0.91

What will this program print, if \texttt{perc} is 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's here?

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

how many tests are executed?
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 `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 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.

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

**perc**

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

non exclusive blocks

**perc**

```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...
Does this program print the correct RPS result this time? Does it always?

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

Name ______________________  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':
    if user == 'paper':
        print('I win *_*!')
    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 ______________________

Your favorite __________ is ____________.

Your least favorite ____________ is ____________.

Pair up with someone near by – answer these questions together…

What is something non-Claremont you have in common?

(1) What does this code print?

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

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

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

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

(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?

Please pass in and up the aisles…

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

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

Then, try these Python q's:

... 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?
comp = 'rock'
user = 'rock'

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

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

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

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

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

else:
    print('An awful tie')

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?
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

This afternoon!
Bring your laptop
- or use one of the CS machines in B105/B102
Get started with Python/text editor/cmdline...

See you in lab!

Alien defeats everything – even Alien

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