CSC 108H: Introduction to Computer Programming

Summer 2011

Marek Janicki

Greetings!

- Be welcome in this lecture hall!
- Please ask questions/let me know if I'm difficult to understand.
- This is an introduction to computer programming using Python.
  - The order matters!
- Intended for people with no experience with programming.

Is CSC 108H for me?

- CSC 148H is offered during this term.
  - Instructor is Dustin Freeman.
  - Assumes knowledge of basic python and object oriented concepts.
  - Does more object oriented stuff and focuses on data structures.
  - Lecture is R:4-6, One 2 hour lab per day.
  - http://www.cdf.toronto.edu/~csc148h/summer/

Well, how can I tell?

- CSC 148H is having a two-day ramp-up workshop.
  - Friday May 20th 1-6 and Saturday May 28th 1-6 in BA3175.
  - Information exists on the course website.
- Intended for people haven't taken CSC 108H but have done some object-oriented programming.
- I encourage you do show up if you're uncertain which course you should be taking.

What will I be doing?

Work	Weight	Comment
Assignments(4)	5%,11%,11%,13%	
Midterm	10%	
Labs(11)	5%	0.5% each, drop the lowest.
Codelab(11)	5%	0.5% each, drop the lowest
Final	40%	Need to get at least 40% to pass the course

Assignments!

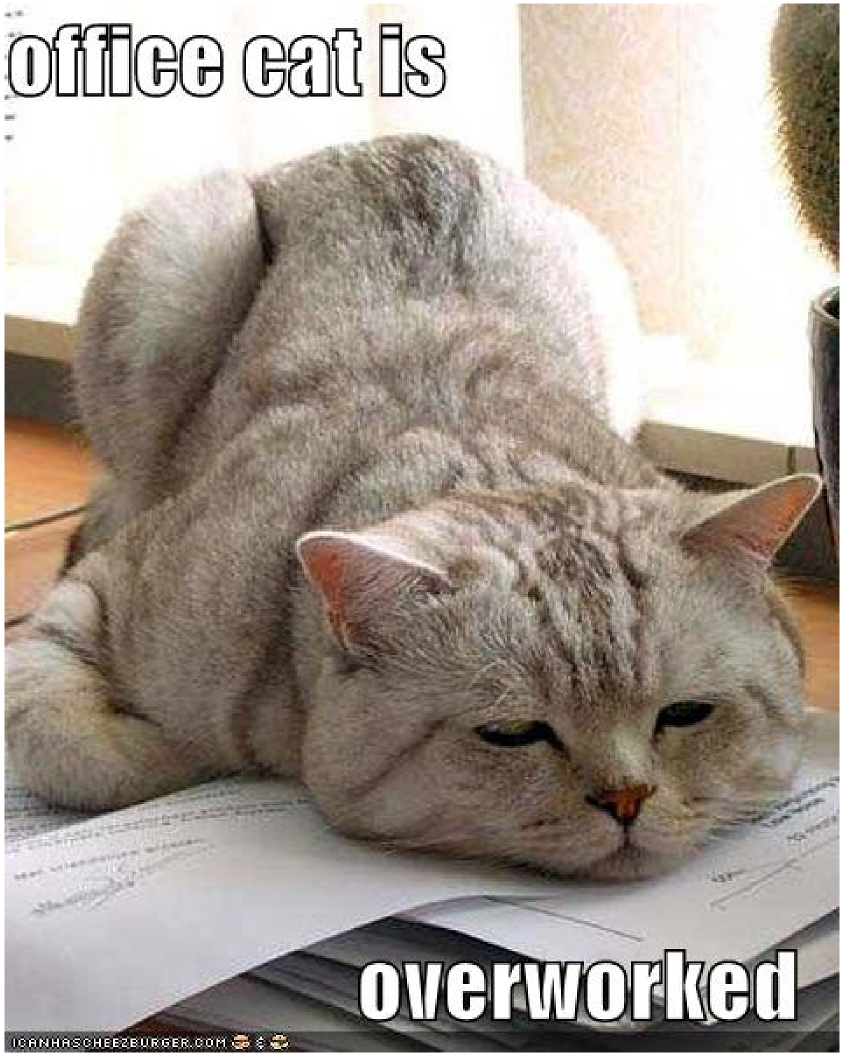
- They will be posted on the website.
- Due 11:59pm on due date, submitted online.
- The first assignment is meant to be small, it will be posted next week.
- The first assignment must be done on your own, remaining assignments can be done in pairs.
- Monogamy and polygamy okay.
- Can use discussion board and labs to meet people.

But I'm busy!

- Fear not! You have 3 grace days.
- Each grace day can be used to get a 24 hour extension on an assignment.
  - You must use grace days in increments of 1 no half days.
  - You can stack grace days, if you wish.
- A team requires two grace days to get an extension.
  - Each partner in a team must contribute one grace day.

But I'm really busy...

- Sorry, that is the only late policy we have.
- Partial solutions that compile will get credit.
- If there's an emergency contact me as soon as possible.



Exams!

- A midterm and a final.
- No, I don't know when or where either are yet.
  - When I find out, I will send out an e-mail and post it on the website.
- They will be closed book written tests.

Labs!

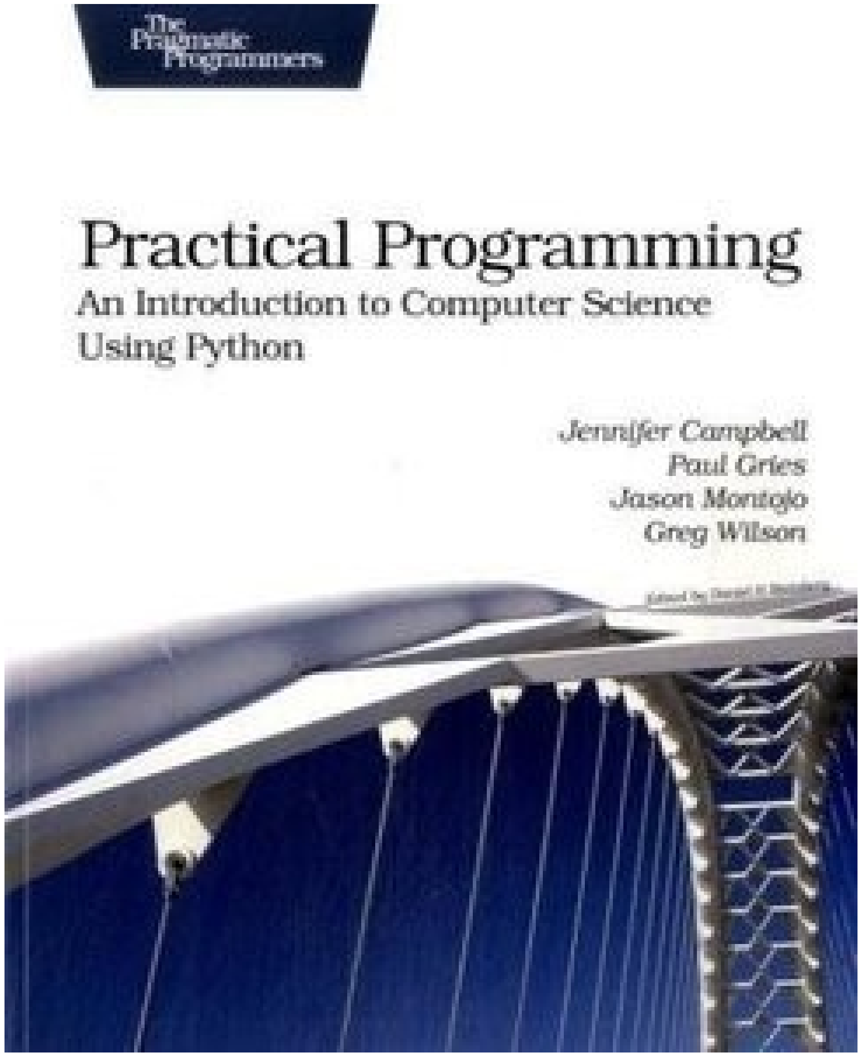
- Labs are done with a partner that is separate from your assignment partner(s).
- They are the tutorials that you sign up for on ROSI.
- They start next week.
- The room assignments will be posted Tuesday.
  - Not everyone has signed up for a lab yet! Please remedy this!

Codelab!

- Weekly online exercises due Tuesdays at 11:59pm.
- They will generally be posted Thursday after lecture.
- You must register online www.turingscraft.com with the registration code TORO-5979-KABQ-9.
- The course website has more information.

The Book.

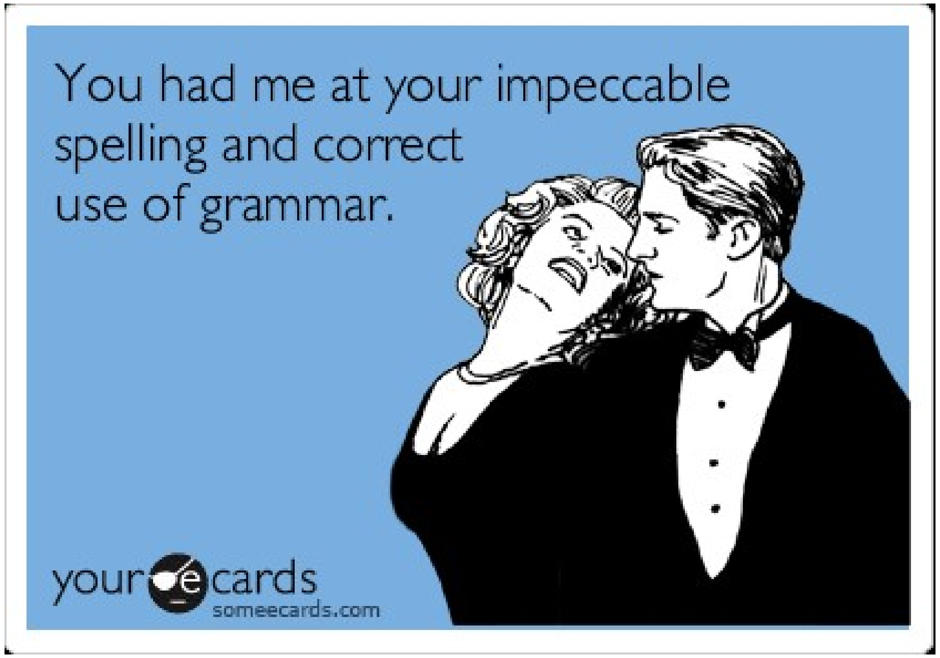
- Practical Programming: An Introduction to Computer Science Using Python.
- Can get it cheaply on Amazon.
- Authors from the department.


The Pragmatic Programmers
Practical Programming
An Introduction to Computer Science Using Python
Jennifer Campbell
Paul Gries
Jason Montojo
Greg Wilson

Getting Help.

- Office Hours.
  - We're deciding on these right now!
- Can ask for help from your TA during labs.
- Course Discussion board.
- Undergraduate Help Centre, BA 2200 4-6, Monday-Thursday.
  - Only 5-6 next week.

But I really need help!


You had me at your impeccable spelling and correct use of grammar.
your ecards
someecards.com

- You can always e-mail me.
  - Please have CSC 108 in the title.
- Please check the discussion board first.

Academic Offences

- You should do all the work that you submit (work by your assignment partner counts).
- Never look at another teams works.
- Never show another team your work.
- Applies to all drafts and partial solutions.
- Discuss how to solve an assignment only with course staff.

Administrivia that you can do!

- Read the course information sheet.
- Make sure you can find the website and discussion board.
- Buy textbook.
- Look up your CDF username.
- Register for Codelab and do the first set of exercises.
- If you're working on your own machine, install the software under Python on the course website.

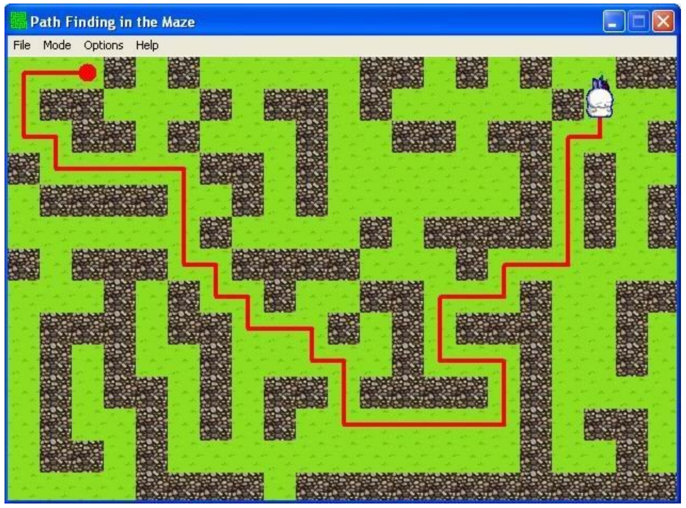
Break, the first.

What is CSC 108H about?

- Learning how to program.
  - We use Python for this, but the concept apply to most languages, and even scripts and macros.
  - Will develop a solid set of programming tools.
- Being able to take human problems, and use programming to solve them.
- Have a better sense of what computer science is about.
    - See how computer science can be applied to climate modelling, bioinformatics, medical science,etc.

Why Programming?

- Powerful and general.
- Can hide a poem in a picture.
- Can remove redeye.
- Allows people to communicate securely.
- Can find optimal paths in huge maps.



What is programming?

- A program is essentially a series of instructions.
  - Like a recipe, or a vague diagram from Ikea.
- So why not use English?
  - Turns out English isn't much better than Ikea diagrams.
  - It's too vague and dependent on context.
    - “Eats shoots and leaves”.
- We need a language that is unambiguous.

Python!

- The answer to our dreams of unambiguous language.
  - Well, in a narrow context.
- Python is unambiguous.
  - Of course, what that means is that you need to be very precise.
  - Think of it as a friend who will never let any small detail go.
- Python is the language, but what reads it?

Wing

- IDE (Integrated Development Enviroment)
- A set of tools used to help us develop code.
- For now we can think of it as the program that runs our python code for us.
- A free version is linked from the website.
- Let's see what it looks like.

Python as a Calculator

- The shell will interpret lines of python that we feed it.
  - Basic mathematical operations are part of python.
  - So we can use python as a calculator.

Python isn't very good at calculating.

- You have multiplication, addition, subtraction, division and powers (*,+,-,/,**) but sometimes the answers are weird.
- If you give python integers, it will assume that you want integers back.
- For fractions, one uses floating point numbers.
  - Python interprets any number with a decimal in it as a float.
- Floats are only approximations of real numbers.

Python comes with a lot of stuff.

- Beyond basic arithmetic there are lots of prebuilt functions in Python.
- Some math ones like max and abs.
- But also other useful ones like dir and help
  - Dir returns a list of functions that are available.
  - Help returns information about a function or module.

Variables.

- A variable is a name that refers to a value.
- Variables let us store and reuse values in several places.
- But to do this we need to define the variable, and then tell it to refer to a value.
- We do this using an assignment statement.

Assignment Statements.

- Form: *variable* = *expression*
  - An expression is a legal sentence in python that can be evaluated.
  - So far we've put in math expressions into the shell and seen them be evaluated to single numbers.
- What it does:
  - 1. Evaluate the expression on the RHS.(This value is a memory address)
  - 2. Store the memory address in the variable on the LHS.

Assignment Statements.

- What it does:
  - 1. Evaluate the expression on the RHS.(This value is a memory address)
  - 2. Store the memory address in the variable on the LHS.
- What this means is that a variable is a name and a memory address. The name points to a memory address where the value is stored.
- This means that variables in python behave fundamentally differently than variables in math.

Break, the second.

Functions

- We already saw that python has a lot of built-in functions.
  - But what if we want to define our own functions?
  - Python allows that.
- First let's think about what it means to define a function in math.
  - Consider f(x)=x^2, and the values of f(3), f(5).
- In python we can do the same with:
- def f(x):

```
    return x**2
```

Functions

- A function definition has the form:

```
def function_name(parameters):
    block
```

  - def is a python keyword; it cannot be used for naming functions or variables.
  - A parameter of a function is a variable. A function can have any number of parameters, including 0.
  - A block is a sequence of legal python statments.
    - A block must be indented.
  - If the block contains the keyword return, it returns a value; otherwise it returns the special value None.

Functions

- Defining a function is different from calling it.
- Think about creating a recipe, vs actually cooking it.
- When we define a function, we essentially say, 'here how we can make a sweet cake'.
- When we call it with some parameters, we actually make the cake with those 'ingredients'.
- But we can repeatedly call functions, so they allow us to have our cake and eat it too.

Naming Conventions.

- Naming rules and conventions apply to functions, variables and any other kind of name that you will see.
- Must start with a letter or underscore.
- Can include letters, numbers, and underscores and nothing else.
- Case matters, so age is not same name as Age.

Naming Conventions.

- Python Convention: pothole_case
  - That is, all lower case, and underscores seperate words.
- CamelCase is sometimes seen, but not for functions and variables.
  - That is, capital letters separate words.
- Single letters are rarely capitalised.
- These conventions are important for legibility which factors into maintaining code.

Types

- Every Python value has a type that describes what sort of value it is and how it behaves.
  - Recall 4 vs 4.0
- There is a built in function `type` that returns the type of an expression.
  - So far we've seen ints and floats.
    - And booleans very briefly, but we'll cover the next week.
- Variables also have types, their type is the type of the expression they refer to.

Home Stretch

- To finish off, we'll see how to create a somewhat useful program quite quickly.
  - Some of the stuff we'll be using is a bit advanced, so don't worry if you don't completely follow everything.
- A lot of people create external modules that extend the capabilities of python.
  - We'll be using the media module, which was created by UofT students.
  - To use a module we import it with import module_name

Media Module

- The basic function of the Media Module is to show pictures.
  - pic = media.load_picture(filename) loads an image into pic.
  - media.show(pic) shows the picture.
- We want to use this to design a program that can take a picture, and make it appear as if it was taken at sunset.

How do we do that?

- Well, we take what we know about image files.
- Basically we know that images files are really many tiny coloured squares called pixels.
- Since we have RGB monitors, this means each colour is a combination of red, green and blue.
- It turns out that the pixel colours are specified by 3 numbers between 0 and 255 that say how much red green and blue each pixel has.
  - So (255,0,0) is red, while (0,255,0) is green and so on.

Leveraging our Knowledge.

- So we know about pixels.
- What do we know about sunset?
    - Colours tend to be redder and less blue or green.
- So if we could change the colour values of each pixel accordingly, we'd probably do pretty well.
    - So let's try decreasing blue and green by 70%,

Pseudo-Code version.

- We want something like:
- For every pixel,

```
    get the (blue/green) component of that pixel.
    Reduce this component by 30%
    set the (blue/green) component of that pixel to the new value.
```

- We're in luck, as there's a way to quickly go over all the pixels.

A General Approach

- While admittedly all planned beforehand, the way we approached the problems was in three stages.
  - Design: We thought about what the right approach was before writing any code.
  - Code: Once we thought we had a good idea, we wrote the code.
  - Verify: we tested our code to make sure we weren't making any dumb mistakes.