Introduction to Programming Python

Peter Newhook

August 10, 2013

http://sdrv.ms/12uSLo9

ladies learning code

Agenda

- Programs
- Basic math with Python
- Types
- Functions
- Variables
- Printing
- Conditions
- Loops
- Data structures
- Building a game

ladies learning code

Introduction

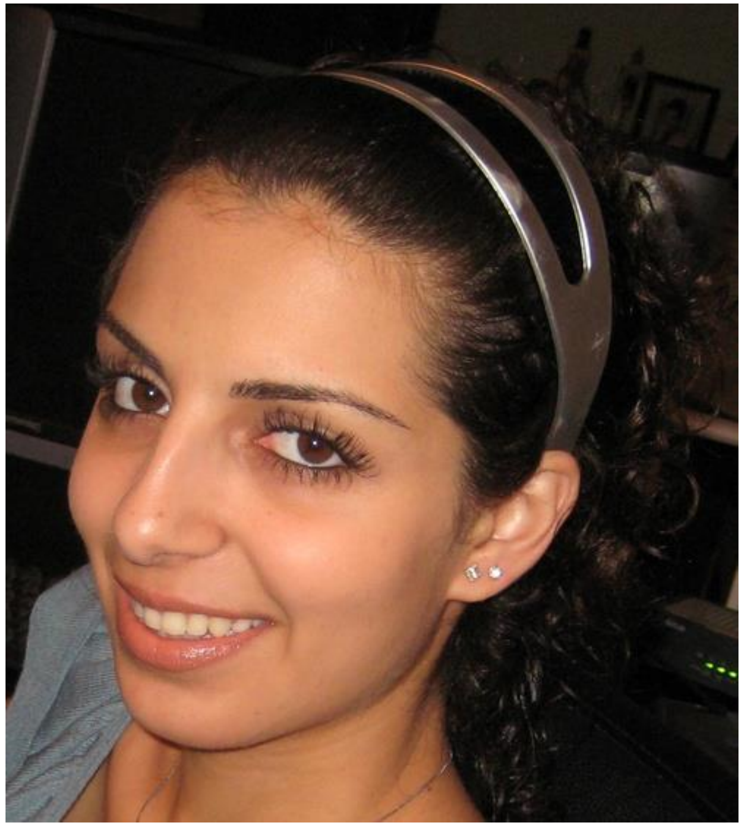
A bit of house keeping

- Ask questions at any point
- Bathroom keys are on the fridge
- Files located at http://sdrv.ms/12uSLo9

ladies learning code

Acknowledgements

- Juan Musleh - @juanmusleh



Quick Prerequisites Check

- Python installed
  - Open the command line/terminal and type ‘python’
  - We’re going to refer to this as the ‘python shell’
- Text editor
  - Notepad++ on Windows
  - TextWrangler on Mac
- A burning desire to learn programming

ladies learning code

You’re already a python programmer

```
x = 5
if x > 2:
    print "I’m greater than two!"
```

ladies
learning
code

Running Python

- We’re going to being in the Python interpreter
- Windows
  - Start > Programs > IDLE
- Mac
  - Applications > Terminal > Utilities > python

Your First Program

```
print "Hello, world!"
```

What Is Python

- Python is a programming language
- What is a programming language?
- Why do we need one?

Your computer speaks another language

```
'  ' 000b 0000             NOP
'PH' 000c 5048             AND $0048
'XR' 000d 5952             DSZ $0152
'EC' 000e 4543             JSM Lbl_0052
'  ' 000f 2020             ADA $0020
'RE' 0010 5245             AND $fe45
'V ' 0011 5620             AND $fe31
'B.' 0012 422e             JSM $fe2e
'  ' 0013 8000 Lbl_0001:   LDA A,I
'  ' 0014 ffff             SES *-1,S [Lbl_00
' $' 0015 1624 Lbl_0002:   CPA $fe39
'  ' 0016 10f9             CPA $00f9
```

ladies learning code

Why python is great

- Easy to use: It is closer to human language than to machine language.
- Powerful: It's used in academic research, Google and NASA.
- Runs Everywhere: It's platform independent. You can write your program on a Mac and run it on Windows.
- Free and Open Source.
- Great Community.

Learning to Code is as Simple as ABC

Always

Be

Coding

Python is good at math

```
>>> 1 + 1
2
>>> 2 - 3
-1
>>> (345 + 3) * 9
3132
```

ladies learning code

Math Operators

- + Addition
- – Subtraction
- * Multiplication
- / Division

Exercise: Try some math

- Does Python follow order of operation rules?
- Did division return the whole result?

We can use more than just numbers

```
>>> "Buenos" + " dias"
'Buenos dias'
>>> 'or I can use single quotes'
'or I can use single quotes'
>>> 'but I cannot use both"
SyntaxError: EOL while scanning string literal
```

ladies learning code

But what if we combine text and numbers?

```
>>> "I'll see you at " + 2
Traceback (most recent call last):
 File "<stdin>", line 1, in <module>
TypeError: cannot concatenate 'str'
and 'int' objects
```

ladies
learning
code

Types

- Python is extremely picky and needs to classify everything before it knows what to do with it
- “I’ll see you at “ is a **String**
- 2 is an **Integer**
- Use the `type()` function to tell what type an object is

Types

```
>>> type("Ladies Learning Code")
<type 'str'>
>>> type(6)
<type 'int'>
>>> type(6.1)
<type 'float'>
>>> type(6.0)
<type 'float'>
```

Why do we need types?

- Python has rules for what to do with different types of objects.
- When you ask Python what **1 + 2** is:
  - It knows that 1 is an INTEGER
  - It knows that 2 is an INTEGER
  - It finds the instruction sheet for INTEGER
  - Scans the instruction sheet for **+**
  - Understands that when you ask for an INTEGER + INTEGER, you want it to add the two together
- The rules for combining “hello” and “world” are different

ladies learning code

Sometimes we can change types

```
>>> "I'll see you at " + str(2)
"I'll see you at 2"
>>> 1 + int("2")
3
```

ladies learning code

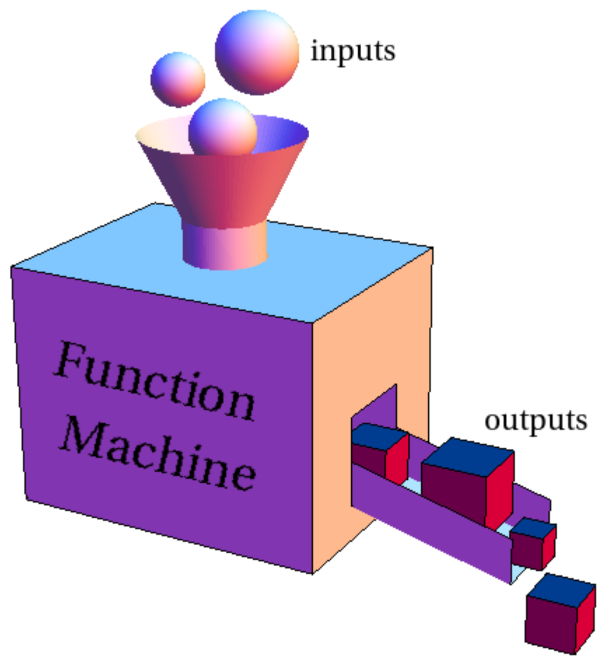
But casting won’t always work

```
>>> 1 + int("two")
Traceback (most recent call last):
   File "<stdin>", line 1, in <module>
ValueError: invalid literal for int() with base 10: 
'two'
```

ladies
learning
code

Beyond math symbols

- What if you want to do something more complex than addition, subtraction, etc?
- Python gives us *functions* to do more complex operations
- Functions act on specific types



ladies learning code

Number Functions

- abs
- pow

Hint: to figure out what a function does use the help function

```
>>> help(abs)
```

String Methods

```
>>> "toronto".capitalize()
'Toronto'
>>> "Ladies Learning Code".replace("Ladies","Girls")
'Girls Learning Code'
>>> "MOAR PYTHON!!!".lower()
'moar python!!!'
```

Try the Following String Methods

- title
- strip
- index
- split

Hint: if to find out what a method does use the help function

```
>>> help("f".strip)
```

But functions only work on certain types

```
>>> pow("elephant", "brain")
Traceback (most recent call last):
 File "<stdin>", line 1, in <module>
TypeError: unsupported operand type(
s) for ** or pow(): 'str' and 'str'
```

ladies learning code

Quick Recap

- Programming languages are intermediates between computer and human language
- Programs are sets of instructions
- Languages have ‘types’
  - string
  - integer
  - float
  - and lots more
- Functions add behaviour
- Most functions work on only one type

Keeping your code DRY

ladies
learning
code

DRY

- **D**on’t **R**epeat **Y**ourself
- You should only have to type anything once

Variables

```
>>> password = "SuperSecure"
>>> password
'SuperSecure'
>>> num1 = 4
>>> num1
4
>>> num2 = 1234.5
>>> num2
1234.5
```

Why variables are useful

- Variable means “liable to change”
- Variables change throughout the lifetime of our program
- Without variables our programs wouldn’t be able to change
- Setting a fixed value in a program is called ‘hard coding’

Let’s throw a party

```
>>> party_budget = 1000
>>> angel_round = 100000
>>> party_budget + angel_round
101000
```

What is your party budget now?

New Budget

```
>>> party_budget
1000
```

We have to tell the party_budget it has a new value

```
>>> party_budget = party_budget + angel_round
>>> party_budget
101000
```

The guestlist

```
>>> guestlist = "Peter June Shreek"
>>> guestlist_copy = guestlist
>>> guestlist_copy
'Peter June Shreek'
>>> guestlist = guestlist + " Roberto"
>>> guestlist
'Peter June Shreek Roberto'
>>> guestlist_copy
'Peter June Shreek'
```

ladies learning code

Anything you can do to a ‘literal’ you can do to a variable

```
>>> startup = "my elephant brain"
>>> startup.title()
'My Elephant Brain'
>>> revenue = -429376
>>> abs(revenue)
429376
>>> revenue + 1000000
570624
```

But did the variable change?

```
>>> startup
'my elephant brain'
```

The function returned a new value, it did not change the original value.

Exercise

- Try using the math operators we saw earlier on variables

Some more strings

```
>>> combined = "combined"
>>> "Strings can be "+ combined + " with variables."
'Strings can be combined with variables.'
>>> fun = "not fun"
>>> "But it is " + fun
'But it is not fun'
```

String Format method

```
>>> utensil = "spoon"
>>> "My {} is too big".format(utensil)
'My spoon is too big'
>>> "The {0} came before the {1}".format('chicken', 'egg')
'The chicken came before the egg'
>>> "The {1} came before the {0}".format('chicken', 'egg')
'The egg came before the chicken'
>>> template = "The Jaguar F-Type has {}hp"
>>> template.format(488)
'The Jaguar F-Type has 488hp'
```

ladies learning code

Exercises (for later)

- Use multiple placeholders but only one argument
- Round Pi to 3 decimals
- Print centered text
  - '   centered   ‘
- Print centered text and pad with plus signs
  - `'+++++++++++centered+++++++++++'`
- Look at http://docs.python.org/2/library/string.html#formatstrings for help

ladies learning code

Recap

- Variables save us from repeating ourselves
- Anything you can do to a literal you can do to a variable
- Variables don’t change until you tell them to
- Formatting strings with the plus sign (+) is error prone
- Use str.format()

Honest to Goodness Programs

Files

- So far we’ve been typing out our entire program
- But when you run real programs (like Chrome or Outlook) you don’t write all the lines
- We need to put repeatable code in a file

Programs in a File

- Create a folder named exercises somewhere on your computer
- Open your text editor
- Create a new file called party.py
- Add the following code

```
party_budget = 1500
party_budget
```

A quick lesson about the command line

- To run Python on the command line you'll have to move around
- To change the directory(folder) we type cd <directory name>
- To move one directory up we type cd ..
- That's all the command line wizardry you need to know today

Running your program

- On the command line, go to the exercises folder
- Type `python party.py`
- Wait, where’s my party budget?

In programs, you have to tell Python to print

```
party_budget = 1500
print party_budget
```

Shouldn’t print be a function?

- Oh boy, now you’ve stepped in it.
- It should
- And in Python 3 it is
- But there’s still a lot of code that uses Python 2.x
- So for now, just use print like a statement

Getting fancy with special characters

```
print "215 Spadina"
print "Toronto"
print "Ontario"
```

```
print  "215 Spadina \nToronto \nOntario"
```

ladies
learning
code

Other Special Characters

- \n
- \'
- \"
- \\
- \t

Comments

- Python ignores anything after the # character

```
# This is a comment
print 'Hello World' # so is this
# print("this won't be printed")
```

ladies learning code

Asking for Input

```
#raw input prompts the user
raw_input("Please enter your name:\n")
#store the result in a variable
my_name = raw_input("Please enter your name:\n")
#store the result in a variable
my_age = int(raw_input("How old are you?\n"))
```

ladies learning code

A Ticket App

- We’re going to write a ticket app for Peter’s Python Prognostications
- Here’s how it’s work
    1. We start with 8 tickets
    2. We ask the user how many they want
    3. We print out that number
    4. We show the user how many tickets are left
- Check tickets1_begin.py if you’re stuck

Issues you may have had

- raw_input always returns a string
  - You’ll need to cast it to an int
- Did you forget to close a quotation mark?
  - It happens to the best of us
- num_tickets wasn’t updated
  - You have to explicitly update the number
  - This is called tracking ‘state’

Is it working

- Good
- Now break it

It depends

Introduction to conditions

ladies
learning
code

Control Flow

- Programs don’t run line by line from start to end
- They jump around according to rules
- One way to specify those rules is to use conditions

Conditional Logic

- We frequently want to test if two values are equal
    - For equality we use the double equal sign
    - 1==1
- We can test if two things aren't equal with !=
    - 1!=2
- There are also many of the math operators we know
    - Greater than: >
    - Greater than or equal to: >=
    - Less than: <
    - Less than or equal to: <=

ladies learning code

Exercise

- In the Python shell use all the operators on the previous slide

Boolean types

- Either true or false
- Useful when making a decision

If statement

```
if 1==1:
     print "I am the walrus!"
```

Beautiful is better than ugly (indentation)

```
# so beautiful
if True:
    print "I am the egg man!"
# my eyes!
if True:
print "coo coo ca choo!"
```

else Statement

```
if 0:
        print "0 is always false"
else:
        print "So the else clause is executed"
```

ladies
learning
code

elif statement

```
today = "Saturday"

if today == "Friday":
     print "Everybody’s working for the weekend!"
elif today ==  "Saturday":
     print "S-A-T-U-R-D-A-Y Night!"
elif today == "Sunday":
     print  "Pancakes."
else:
     print "Early to bed and early to rise."
```

ladies learning code

Exercise

- Stop the user from buying tickets if they ask for more than are available

What we’ve covered

- Programs
- Basic math with Python
- Types
- Functions
- Variables
- Printing
- Conditional logic

Morning Recap

- Write a program that asks you what your favourite colour is and tells you something else that is the same colour
  - What is your favourite colour?
    Green
    I like turtles

- Write a program that prints a string if the input is less than 10 **or** greater than 20
  - Do the same thing for greater than 10 **and** less than 20

ladies learning code

Colour

```
fav = raw_input('What is your favourite colour?\n')
fav_lower = fav.lower()
if fav_lower == 'green':
    print 'I like turtles'
elif fav_lower == 'blue':
    print 'Like the sky'
elif fav_lower == 'purple':
    print 'My favourite ice cream when I was young was called Purple People Eater'
else:
     print "{} is my favourite too.".format(fav)
```

ladies learning code

and/or

```
input = raw_input("Give me a number: ")
input = int(input)
if input < 10 or input > 20:
    print 'That\'s a good number'
if input > 10 and input < 20:
    print "You've found the sweet spot"
```

ladies learning code

Getting loopy

Loops

- So far our code has run from top to bottom
- Programs usually need to do something over and over agin
- Computers don’t mind doing repetitive tasks

While Loop

```
#"while" something is true
#perform an action
while True:
    print 'Are we there yet?'

#use Ctrl + c to cancel the program
```

ladies
learning
code

Apple Campus
One Infinite Loop
CAUTION

We need a way to stop the loop

```
#code within the loop my invalidate the condition
loop = True
while loop:
    print 'Are we there yet?'
    loop = False
print 'Done the loop'
```

Or we could ask the user for input

```
loop = True
while loop:
     answer = raw_input('Are we there yet?')
     loop = answer == 'no'
print 'Done the loop'
```

break dancing

```
#or we can use the break keyword
loop = True
while loop:
    print 'Are we there yet?‘
    break
print 'Done the loop'
```

To run a certain number of times we test a variable and increment within the loop

```
x = 1
while x < 10:
    print x
    x += 1
```

Exercise: Print from 10 to 1

```
x = 10
while x > 0:
    print x
    x -= 1
```

ladies learning code

Guided Exercise

- We’re going to write a game for the computer
- The computer will pick a number and we have to guess it
  - `from random`
  - `import randint x = randint(1,10)`
- The user should know if they are too high or too low
- We know how to ask for input
- We know how to compare guesses
- We want to give the user multiple chances

ladies learning code

Data structures

ladies
learning
code

Lists

- Lists are lists
- That was useless
- But seriously, they’re just a set of objects
- Lists can hold strings, integers, floats, booleans, and anything else you can create

Creating Lists

```
mylist = [1,2,3,4]
print mylist #[1, 2, 3, 4]
companies = ['My Elephant Brain', 're:3D', 'Hacker You']
print companies
#['My Elephant Brain', 're:3D', 'Hacker You']
kitchen_sink = [1 , 'a', True]
print kitchen_sink
#[1, 'a', True]
```

ladies learning code

Accessing Items in a List

```
>>> companies[1]
're:3D'
>>> companies[-1]
'Hacker You'
>>> companies[10]
Traceback (most recent call last):   File "<stdin>",
 line 1, in <module>
IndexError: list index out of range
```

※ladies learning code

Exercise

- Using a list and a while loop, print the days of the week
- ‘break’ the loop on Wednesday

Modifying a List

```
#add and item
l.append(5)
print l
#remove and item
l.remove(3)
print l
# remove throws an error if the item doesn't exist
l.remove(9)
```

ladies
learning
code

A Quick jump Back to Loops

```
# for loops make it easy to loop over a list
for company in companies:
    print company
# they're also useful for looping
# a konwn number of times
for i in range(1,10):
    print i
```

ladies learning code

Exercise

- Use a for loop to print the days of the week
- Use a for loop to count from 10 to 1
  - Hint: by default range() increases by 1 each time, but you can give it a third argument to tell it the number to increment by each time
  - E.g range(0, 10, 2)

The Grand Finale

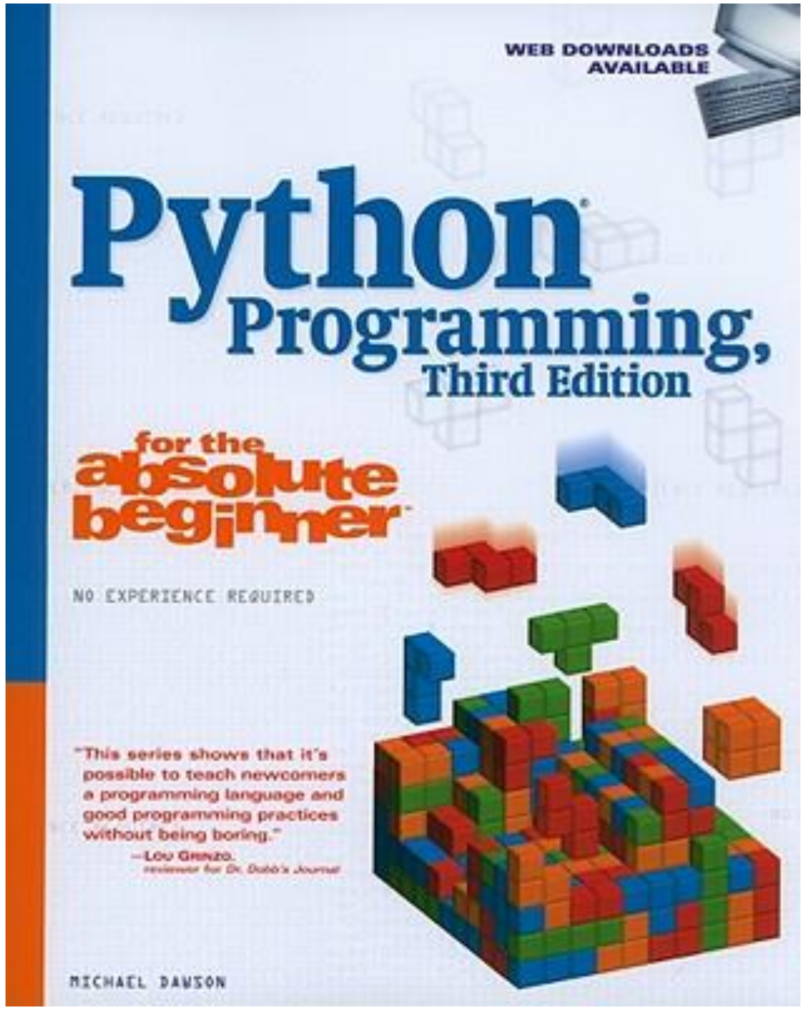
ladies learning code

Hangman

- Let's make a hangman game using what we learnt today.
- But before you start coding away, it helps to think about what you're trying to do.
- Remember, computers need specific instructions, so you need to break it down.
- You can start by playing hangman on paper with each other.
- Start with hangman_draft.py

ladies learning code

Hangman credit to Python Programming for the Absolute Beginner


WEB DOWNLOADS AVAILABLE
Python
Programming,
Third Edition
for the absolute beginner
NO EXPERIENCE REQUIRED
"This series shows that it's possible to teach newcomers a programming language and good programming practices without being boring."
—Lou Grinzo,
reviewer for Dr. Dobb's Journal
MICHAEL DAWSON

ladies learning code

You’re programmers now!

- Sure there’s more to know
- There’s always more to know
- The trick is using what you know
- http://www.pyschools.com/
- http://www.djangobook.com/en/2.0/index.html
- http://learnpythonthehardway.org/

Congratulations!

- Thank-you Ladies Learning Code
- Thank-you mentors
- And thank-you participants

ladies
learning
code

Creative Commons By Attribution

http://creativecommons.org/licenses/by/3.0/deed.en_US

Putting the ‘fun’ in functions!

ladies learning code

Functions

```
# we put reusable code inside 'functions'
def add_two(a, b):
    return a + b
ans = add_two(1, 2)
print ans
```

ladies
learning
code

Exercise

- Write functions to do the following
  - Multiply two numbers
  - Combine an int and a string
  - Print the sequence of a given number, down to zero
    - 5, 4, 3, 2, 1, 0

Modules

ladies
learning
code

Python code is reusable by other code

```
# start python in the same folder as functions.py
from functions import add_two
print add_two(1,2)

# from math import factorial
```