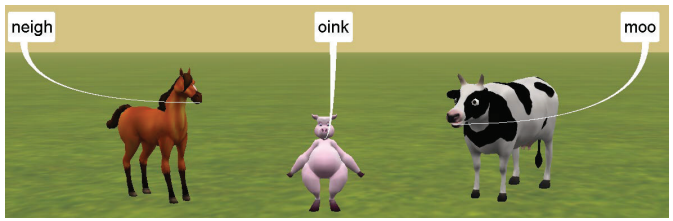
Compsci 101
Functions, Randomness, Selection

Susan Rodger
January 18, 2022

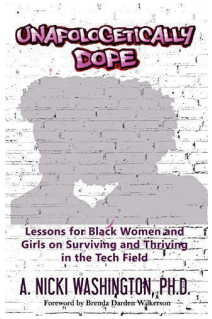


Don't sit in the last four rows

Come closer

`D` is for …

- **Debugging**
  - A key skill in making your programs run
- **Data (Science)**
  - Creating information from 0's and 1's
- **Dictionary**
  - Ultimate Python Data Structure



Prof. Nicki Washington
Duke University

- Research focuses on identity and cultural competence in computing
- Teaches: CompSci 240
- Book: ***Unapologetically Dope: Lessons for Black Women and Girls on Surviving and Thriving in the Tech Field***
- On changing the environment, she says:

"The only way things will change is if those in the majority do the work. This also means that companies should place high expectations of cultural competence on prospective interns and new employees. This, in turn, places more expectations on college and university computing departments to focus on it as well. Only then will we start to see a real paradigm shift."

Announcements

- Assignment 0 due tonight, 11:30pm
- Assignment 1 out today
- APT-1 due Thursday
- Drop/Add over Wednesday
  - You cannot change lab section without a perm no.
- QZ01-QZ04 submitted through Thursday 10:15am
- QZ05 is DUE at 10:15am on Thursday/will turn off!
- Trouble with Pycharm? Get help

- Remember: Ed Discussion back channel during lecture

Plan for the Day

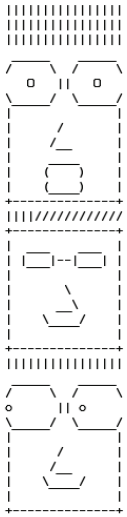
- Assignment 1
- Print vs. Return
- Python Tutor
- Why use functions?
- Selection (if…elif…else)
- Random library

Assignment 1 and Pre-Lab 2

- Assignment 1 Faces due January 27

- Sakai Quiz on Assignment 1
  - Read through assignment 1
  - Take the quiz
  - Can take many times
  - Due January 25!

- Prelab 02 – before lab
  - Read Assignment 1 and take quiz once



Program execution

- Start at first line
- Ignore comments and blank lines
- Function – recognize, don't execute
- Statements – executed one line at a time
  - After one statement, next statement
  - Calling a function transfers control to function
  - Function returns control back to where it was called by one of these:
    - Reach last line in the function, returns with None
    - Execute a return statement, return value

Print vs. Return

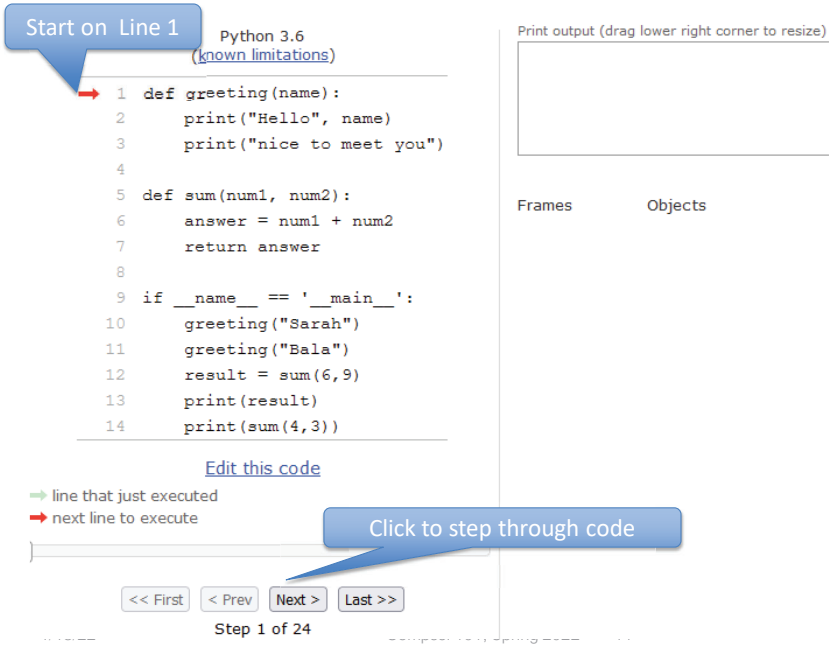
- Function ends one of two ways:
  - Reach end of function
  - Execute return statement
- Printing is not the same as returning
  - Print doesn't leave the function

```
7   def greeting(name):
8       print("Hello", name)
9       print("nice to meet you")
10
11  def sum(num1, num2):
12      answer = num1 + num2
13      return answer
14
15  if __name__ == '__main__':
16      greeting("Sarah")
17      greeting("Bala")
18      result = sum(6,9)
19      print(result)
20      print(sum(4,3))
```


Python Tutor Tool: Understanding Execution

- Using PythonTutor: http://pythontutor.com
  - Tool to trace through code
  - Copy and paste in your code
  - Think about these things as we trace code with Python Tutor
    - How are functions defined?
    - Where does execution begin?
    - What is the global frame?
    - What is a local/function frame?

Trace code with Python Tutor: Start



What PythonTutor Demonstrates

- What happens when program is first "executed"?
  - Execution starts at top of the file
    - Good practice: "Starting" code is in main program block
  - Functions created and referenced in global frame
- What happens when function called?
  - Arguments passed as parameters to function
    - Passed in same order inside parenthesis
    - See green and red arrows when executing
  - Control passes to function which executes
  - Return value replaces function call

WOTO-1 Simple Functions
http://bit.ly/101s22-0118-1

- In your groups:
  - Come to a consensus

Why Use Functions?

- Re-use code/abstractions in multiple contexts
  - Sqrt, wordcount, URL-Webpage examples
- Test code/abstractions separately from their use
  - Develop independently, use with confidence
- Easier to change, re-use in different contexts
  - Relevant to Assignment 1: Faces
- Reduce risk of copy + paste mistakes

Old MacDonald Song!

```
if __name__ == '__main__':
    print("Old MacDonald had a farm, Ee-igh, Ee-igh, oh!")
    print("And on his farm he had a pig, Ee-igh, Ee-igh, oh!")
    print("With a oink oink here")
    print("And a oink oink there")
    print("Here a oink there a oink everywhere a oink oink")
    print("Old MacDonald had a farm, Ee-igh, Ee-igh, oh")

    print()
    print("Old MacDonald had a farm, Ee-igh, Ee-igh, oh!")
    print("And on his farm he had a horse, Ee-igh, Ee-igh, oh!")
    print("With a neigh neigh here")
    print("And a neigh neigh there")
    print("Here a neigh there a neigh everywhere a neigh neigh")
    print("Old MacDonald had a farm, Ee-igh, Ee-igh, oh")
```


How to make code better?

```
if __name__ == '__main__':
    print("Old MacDonald had a farm, Ee-igh, Ee-igh, oh!")
    print("And on his farm he had a pig, Ee-igh, Ee-igh, oh!")
    print("With a oink oink here")
    print("And a oink oink there")
    print("Here a oink there a oink everywhere a oink oink")
    print("Old MacDonald had a farm, Ee-igh, Ee-igh, oh")

    print()
    print("Old MacDonald had a farm, Ee-igh, Ee-igh, oh!")
    print("And on his farm he had a horse, Ee-igh, Ee-igh, oh!")
    print("With a neigh neigh here")
    print("And a neigh neigh there")
    print("Here a neigh there a neigh everywhere a neigh neigh")
    print("Old MacDonald had a farm, Ee-igh, Ee-igh, oh")
```

WOTO-2 Old MacDonald
http://bit.ly/101s22-0118-2

- Discuss what is new in the code

Try out code? Add a Verse?

- I will make the code from lecture available after class as a .zip file
- Steps:
  1. Create new project
     1. Project Interpreter is what created before
  2. Download zip file
  3. Unzip and copy files into new project

Functions Summarized

- Function call and Function definition related
  - Call must provide correct arguments
  - Names don't matter, types are important
    - `print(verse("robot", 42))` ?
- Functions help design, implement, organize
  - Without functions no APIs, no big programs

Making Decisions:

- Execute different code depending on something
  - Ask a question
  - Make decision based on answer
- If condition is true then do something
  - Condition: true or false
  - Something: any Python code

Selection Syntax

```
if BOOLEAN_CONDITION:
    CODE_BLOCK_A
```

```
if BOOLEAN_CONDITION:
    CODE_BLOCK_A
else:
    CODE_BLOCK_B
```

```
if BOOLEAN_CONDITION:
    CODE_BLOCK_A
elif BOOLEAN_CONDITION:
    CODE_BLOCK_B
else:
    CODE_BLOCK_C
```

- What is similar and different?
  - What other variations could work?
  - Could only `elif…else` work?

Example: If

Output:

```
6   def larger(num1, num2):
7       if num1 > num2:
8           return num1
9       return num2
10
11  if __name__ == '__main__':
12      print(larger(9, 17))
13      print(larger(17, 9))
14      print(larger(25, 6))
```


Example2: If-Elif-Else

Output:

```
6   def pluralize(word):
7       if word == "fish":
8           return word + "es"
9       elif word == "brush":
10          return word + "es"
11      else:
12          return word + "s"
13
14  if __name__ == '__main__':
15      print(pluralize("brush"))
16      print(pluralize("card"))
17      print(pluralize("fish"))
18      print(pluralize("frog"))
19      print(pluralize("fox"))
```


Random Module

- https://docs.python.org/3/library/random.html
- random.randint(*a*, *b*)
  - Return a random integer *N* such that a <= N <= b.
- Must import random at top of file to use the library

Example: Random

```
6   import random
7
8   def larger(num1, num2):
9       if num1 > num2:
10          return num1
11      return num2
12
13  if __name__ == '__main__':
14      x = random.randint(1,20)
15      y = random.randint(1,20)
16      print(x, y, larger(x,y))
17      x = random.randint(1,200)
18      y = random.randint(1,200)
19      print(x, y, larger(x,y))
```

Output:

WOTO-3
http://bit.ly/101s22-0118-3