Compsci 101
7-steps, Functions, Order of Execution

Susan Rodger
January 13, 2022

C is for …

- Computer Science and Computing
  - It’s what we do
- Collaboration
  - Review the policy
- Cookies
  - Good for the web and for …
- CSV
  - Comma Separated Values: Data

Ayanna Howard

- Educator, Researcher and Innovator
- Professor & Chair of the School of Interactive Computing, Georgia Tech
- Now Dean of Engineering at The Ohio State University!
- Robotics – Robots and Bias, Robots changing lives of children with disabilities, Robots beyond part of the family
- Top 50 U.S. Women in Tech, Forbes, 2018

Announcements

- Lab 01 Friday,
  - Complete Prelab before going to lab
- APT-1 out today, due Thursday, January 20
- Assignment 0 due Tuesday, January 18
- Sakai quizzes on readings due 10:15am on date due
  - Get three tries, score highest score
  - First two weeks we allow you to submit late
  - First 5 quizzes turn off, 10:15am Jan 20

I believe that every engineer has a responsibility to make the world a better place. We are gifted with an amazing power to take people’s wishes and make them a reality.
Join Duke Mailing lists
compsci@duke.edu

- Mailing list about
  - Jobs, internships, research positions
  - Events related to computer science
- How to join:
  - Go to: lists.duke.edu
  - Be sure to authenticate
  - Add compsci@duke.edu

- BE IN THE KNOW ABOUT COMPSCI!

What is a Function?

- Function has:
  - Name
  - Maybe inputs
  - Processes or calculates something
  - Has a result

Functions in the Real World: URL in webpage

- Function has:
  - Name: “Search”
  - Input: www.duke.edu
  - Calculates:
  - Returns back:
Functions in the Real World: URL in webpage

- Function has:
  - Name: "Search"
  - Input: www.duke.edu
  - Calculates: Figures out where web page is
  - Returns back: the actual web page

Functions in the Real World: calculator

- Function has:
  - Name: calculator
  - Input: number(s), operator
  - Example: 25, squareroot
  - Calculates: value of expression
  - Returns back: 

\[ \sqrt{25} \]
Functions in the Real World: Counting words in Microsoft Word

- Function has:
  - Name: Word Count
  - Input: contents of the document (e.g. a story)
  - Calculates: counts number of words
  - Returns back: number of words (e.g. 352)

Built-in Python Function – len()
already exists, you use it

- len() function
- Function has:
  - Name: len
  - Input: a string
  - Calculates: number of characters in string
  - Returns back: number

Examples:

x = len("duke")
# value of x: 4

y = len("computer")
# value of y: 8
Built-in Python Function – `str()` already exists, you use it

• `str()` function
• Function has:
  • Name: `str`
  • Input: an expression
  • Calculates: string version of expression’s value
  • Returns back: string

Examples:

```
x = str(623)
# value of x: “623”
y = len(str(2**8))
= len(str(256))
= len(“256”)
# value of y: 3
z = str(6 + 8.3)
# value of z: “14.3”
```

Other Python built-in functions

• `type(something)`
  • Returns type of variable `something`

• `int(7.8)`
  • Returns integer value of decimal number, e.g. 7

• `float(4)`
  • Returns float value of integer, e.g. 4.0

print() function

• General function has:
  • Name
  • Maybe inputs
  • Processes or calculates something
  • Has a result

• `print(“hi cat”)`
  • Name: `print`
  • Input: “hi cat”
  • “processes”, generates output
  • No return value, returns None
Example with lines numbered:

```python
1  x = float(6)
2  print("x is", x)
3  y = print("x is", x)
4  print("y is", y)
```

Output:

```
x is 6.0
```

Example with lines numbered:

```python
1  x = float(6)
2  print("x is", x)
3  y = print("x is", x)
4  print("y is", y)
```

Output:

```
x is 6.0
```

---

Example with lines numbered:

```python
1  x = float(6)
2  print("x is", x)
3  y = print("x is", x)
4  print("y is", y)
```

Output:

```
x is 6.0
```

---

Example with lines numbered:

```python
1  x = float(6)
2  print("x is", x)
3  y = print("x is", x)
4  print("y is", y)
```

Output:

```
x is 6.0
```

---

Example with lines numbered:

```python
1  x = float(6)
2  print("x is", x)
3  y = print("x is", x)
4  print("y is", y)
```

Output:

```
x is 6.0
```

---

Example with lines numbered:

```python
1  x = float(6)
2  print("x is", x)
3  y = print("x is", x)
4  print("y is", y)
```

Output:

```
x is 6.0
```

---

Example with lines numbered:

```python
1  x = float(6)
2  print("x is", x)
3  y = print("x is", x)
4  print("y is", y)
```

Output:

```
x is 6.0
```

---

Example with lines numbered:

```python
1  x = float(6)
2  print("x is", x)
3  y = print("x is", x)
4  print("y is", y)
```

Output:

```
x is 6.0
```

---

Example with lines numbered:

```python
1  x = float(6)
2  print("x is", x)
3  y = print("x is", x)
4  print("y is", y)
```

Output:

```
x is 6.0
```

---

Example with lines numbered:

```python
1  x = float(6)
2  print("x is", x)
3  y = print("x is", x)
4  print("y is", y)
```

Output:

```
x is 6.0
```

---

Example with lines numbered:

```python
1  x = float(6)
2  print("x is", x)
3  y = print("x is", x)
4  print("y is", y)
```

Output:

```
x is 6.0
```

---

Example with lines numbered:

```python
1  x = float(6)
2  print("x is", x)
3  y = print("x is", x)
4  print("y is", y)
```

Output:

```
x is 6.0
```

---

Example with lines numbered:

```python
1  x = float(6)
2  print("x is", x)
3  y = print("x is", x)
4  print("y is", y)
```

Output:

```
x is 6.0
```

---

Example with lines numbered:

```python
1  x = float(6)
2  print("x is", x)
3  y = print("x is", x)
4  print("y is", y)
```

Output:

```
x is 6.0
```

---

Example with lines numbered:

```python
1  x = float(6)
2  print("x is", x)
3  y = print("x is", x)
4  print("y is", y)
```

Output:

```
x is 6.0
```
Example with lines numbered:

```
1 x = float(6)
2 print("x is", x)
3 y = print("x is", x)
4 print("y is", y)
```

Output:
```
x is 6.0
x is 6.0
y is None
```

Writing your own Python function

• Format:
  ```python
def nameOfFunction(parameters):
    <body, or lines of code>
    return value  # optional, but likely
  ```

• Example define function:
  ```python
def inchesToCentimeters(inches):
    centi = inches * 2.54
    return centi
  ```

• Use or call function:
  ```python
  answer = inchesToCentimeters(10.0)
  print(answer)
  ```
  Output:
  ```python
  25.4
  ```
Writing your own Python function

- **Parameter**
  - Variable, placeholder for a value
  - In parenthesis in first line of definition of function
- **Argument**
  - Expression or value
  - In parenthesis when calling or using a function
- **Example:**
  ```python
def inchesToCentimeters(inches):
    centi = inches * 2.54
    return centi
```
- **Use or call function:**
  ```python
answer = inchesToCentimeters(10.0)
print(answer)
```

What happens when executes?

Output:
```
def inchesToCentimeters(inches):
    centi = inches * 2.54
    return centi
```
```
if __name__ == '__main__':
    answer = inchesToCentimeters(10.0)
    print(answer)
    answer = inchesToCentimeters(3.0)
    print(answer)
```

Note function inchesToCentimeters is on line 8

What happens when executes?

Output:
```
def inchesToCentimeters(inches):
    centi = inches * 2.54
    return centi
```
```
if __name__ == '__main__':
    answer = inchesToCentimeters(10.0)
    print(answer)
    answer = inchesToCentimeters(3.0)
    print(answer)
```

Note lines below line 13 are indented 4 spaces each

Ignore lines indented, so next line is line 13. If __name__ == '__main__' is special and means: Start executing program on next line
What happens when executes?

def inchesToCentimeters(inches):
    centi = inches * 2.54
    return centi

if __name__ == '__main__':
    answer = inchesToCentimeters(10.0)
    print(answer)
    answer = inchesToCentimeters(3.0)
    print(answer)

Evaluate the right hand side of the “=”
Call the function inchesToCentimeters
Pass the argument 10.0 for the parameter inches

What happens when executes?

The RHS inches * 2.54 is calculated as 25.4.
Then centi is assigned the value 25.4.

What happens when executes?

Execution moves to line 8 where the definition of function inchesToCentimeters is.
inches has the value 10.0

What happens when executes?

The value of the variable centi (25.4) is returned to the RHS of line 14 where the function was called.
What happens when executes?

```python
def inchesToCentimeters(inches):
    centi = inches * 2.54
    return centi
```

If `__name__ == '__main__'`:
```python
answer = inchesToCentimeters(10.0)
print(answer)
answer = inchesToCentimeters(3.0)
print(answer)
```

**Output:**
```
25.4
```

The value of variable `answer` is printed.

Evaluate the right hand side of the `==`.
Pass the argument 3.0 for the parameter `inches`.

Execution moves to line 8 where the definition of function `inchesToCentimeters` is.
`inches` has the value 3.0.
What happens when executes?

```python
def inchesToCentimeters(inches):
    centi = inches * 2.54
    return centi
```

Output:
25.4

The RHS `inches * 2.54` is calculated as 7.62. Then `centi` is assigned the value 7.62.

```
if __name__ == '__main__':
    answer = inchesToCentimeters(10.0)
    print(answer)
    answer = inchesToCentimeters(3.0)
    print(answer)
```

Output:
```
25.4
7.62
```

The value of the variable `centi` (7.62) is returned to the RHS of line 16 where the function was called.

answer is assigned the return value 7.62 and line 16 has completed executing.

```
if __name__ == '__main__':
    answer = inchesToCentimeters(10.0)
    print(answer)
    answer = inchesToCentimeters(3.0)
    print(answer)
```

Output:
```
25.4
7.62
```

The value of variable `answer` is printed.
Let's go see this in Pycharm and add a function

```python
def pluralize(word):
    word = word + "es"
    return word
```

```python
newWord = pluralize("fish")
print(newWord)
word1 = "dress"
word2 = pluralize(word1)
print(word2)
word1 = "book"
print(pluralize(word1))
```

WOTO – Working Together (breakout groups)

- Given a bitly link
  - Type it in OR click on it on the calendar page

- What you should do:
  - Introduce yourselves
  - Each person fills out google form
  - Put in your name, email and netid
  - Discuss each question and fill out
  - Be mindful of time

WOTO: Calling Functions

Details: `print(addTen(addTen(x)))`

```python
print(addTen(addTen(x)))
print(addTen(addTen(5)))
print(addTen(15))
print(25)
```

Output:
25
APTs in 101 and 201

- Algorithm Problem-solving and Testing
  - Algorithm that’s Automatically Tested
  - In use at Duke since 2003, million+ APTs solved

- Given a problem statement
  - Read, think, plan on paper …
  - Write a function to solve the problem
  - Submit the code for testing, debug if necessary
- Where do you start with problem solving?

---

Programming Process: High-level

The Seven Steps

- First part: devise the algorithm
  - The meta-problem solving piece
  - Big/complex enough to be 4 steps (more shortly)

- After devising the algorithm, translate to code
  - Plan first, then code
  - Bridge analogy: blue prints, then construction
  - Essay analogy: outline, then prose

- Next test our program
  - Testing important, often under-taught skill
The Seven Steps
Programming Process: High-level

- Ideally would be correct first time; may need to debug
  - Identify problem (with science!)
  - Return to appropriate prior step to fix the problem

Steps 1—4: Devise Algorithm

- Steps 1—4: devise the algorithm
  - Learn to do this well, be an excellent programmer
  - Language: does not matter

Steps 1—4: Example:
Calculate the average of two numbers

- Step 1: 2 + 5 = 7, 7/2 = 3.5
- Step 2:
  - Add 2 + 5 and get 7
  - Divide 7 by 2 and the result is 3.5
Steps 1—4: Example: Calculate the average of two numbers

• Step 3:
  • Two variables num1 and num2
  • Add the two numbers together: result is num1 + num2
  • Divide the result by 2 and you have the answer answer is result / 2

Step 4: Try a different example
  • Use 8 and 6, num1 is 8, num2 is 6
  • Add the two numbers together: result is num1 + num2, is 14
  • Divide the result by 2 and you have the answer – Answer is result/2, which is 7

• IT WORKS!

Step 5: let’s convert it to code!

• Go to Pycharm
• We will also:
  • Test it – Step 6
  • Debug it – Step 7
Solving Laundry APT

- Navigate to APTs in class website and ...

CompSci 101, Spring 2022

APTs

APTs

See below for hints on what to do if your APT doesn’t run.

For each problem in an APT set, complete these steps by the due date:

- first click on the APT set below to go to the APT page.
- write the code, upload the file and click the Submit link.
- check your grade on the grade code page by clicking on check submissions.

In solving APTs, your program should work for all cases, not just the test cases we provide. We may test your program on additional data.

<table>
<thead>
<tr>
<th>APT</th>
<th>Due Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>APT-1</td>
<td>January 20</td>
</tr>
</tbody>
</table>

Solving Laundry APT

- Navigate to APTs in class website and ...

Problem Statement

Consider the problem of trying to do a number of loads of laundry, given only one washer and one dryer. Washing a load takes 25 minutes, drying a load takes 25 minutes, and folding the clothes in a load takes 10 minutes, for a total of 1 hour per load (assuming that the time to transfer a load is built into the timings given). 10 loads of laundry can be done in 12 hours, 600 minutes, using the method of completing one load before starting the next one. Though it can be done faster, see examples.

Write the method, minutesNeeded, that returns the shortest time needed to do n loads of laundry. In other words, given an integer value representing the number of loads to complete, n, determine the smallest number of minutes needed to complete all loads of laundry.

Not ready for coding yet!!!!!
Solving Laundry APT – Step 1


• What is important info?

Problem Statement

Consider the problem of trying to do a number of loads of laundry, given only one washer and one dryer. Washing a load takes 25 minutes, drying a load takes 25 minutes, and folding the clothes in a load takes 10 minutes, for a total of 1 hour per load (assuming that the time to transfer a load is built into the timings given). 10 loads of laundry can be done in 10 hours, 600 minutes, using the method of completing one load before starting the next one. Though it can be done faster, see examples.

Write the method, minutesNeeded, that returns the shortest time needed to do m loads of laundry, representing the number of loads to complete, m. Determine the smallest number of minutes needed to do the loads of laundry.

Specification

```python
filename: Laundry.py
def minutesNeeded(m):
    """
    Return integer number of minutes
    """
    # you write code here
```

• m = 1

```
Wash  | Dry  | Fold  
Wash  | Dry  | Fold  
```

• Return: 25 + 25 + 10 = 60 minutes

Write down what we just did for m=2

• m = 2

```
Wash  | Dry  | Fold  
Wash  | Dry  | Fold  
Wash  | Dry  | Fold  
```

• Return: 25 + 25 + 25 + 10 = 85 minutes

• = 85 minutes
Step 1: Work an instance yourself
Step 2: Write down exactly what you just did
Step 3: Generalize your steps
Step 4: Test your steps (with new input)

Reading an APT

1. m = 1
   returns: 60
   You must was minutes.
2. m = 2
   returns: 85

Solving Laundry APT – Steps 3 and 4


• What is important info?

Problem Statement

Consider the problem of trying to do a number of loads of laundry, given only one washer and one dryer. Washing a load takes 25 minutes, drying a load takes 25 minutes, and folding the clothes in a load takes 10 minutes, for a total of 1 hour per load (assuming that the time to transfer a load is built into the timings given). 10 loads of laundry can be done in 10 hours, 600 minutes, using the method of completing one load before starting the next one. Though it can be done faster, see examples.

Write the method, minutesNeeded, that returns the shortest time needed to do m loads.

Solving an APT

• Create new project
  • File > New Project
  • Existing interpreter (first project you made from installation)

• Create new Python File
  • Right click on project > New > Python File

• Create function within module
  • Name it properly!
APT Correct → The Green Dance (Fall 2020)

APT Testing and Submission

• You wrote the code, how is it tested?
  • Submit .py file with function to server
  • Server imports it
  • Server tests and checks by calling your function

• The APT testing framework calls your code!
  • Don't call us, we'll call you: Hollywood principle

• Test + Submit + Check Grade

Laundry dissected

def minutesNeeded(m):
  • Wrote formula using code to define a function

def minutesNeeded(m):
  return (m-1) * 25 + 60

• Wrote formula using code to define a function
• How to use and re-use? By “calling” it
  • Functions allow code to be re-used
  • Len(), float(), minutesNeeded()

  time = minutesNeeded(2)
Laundry dissected

```python
def minutesNeeded(m):
    return (m-1) * 25 + 60
```

- Wrote formula using code to define a function
- How to use and re-use? By “calling” it
  - Functions allow code to be re-used
  - `float()`, `minutesNeeded()`

```python
time = minutesNeeded(2)
print(time)
```

Output is 85

Testing Laundry – three ways

1) Run on the apt page
   - Need internet connection, may take time

2) Locally in Python Program Laundry
   - Get it working before you use apt page

```python
if __name__ == '__main__':
    num = 1
    print("m is", num, minutesNeeded(num))
    num = 2
    print("m is", num, minutesNeeded(num))
    num = 3
    print("m is", num, minutesNeeded(num))
    num = 10
    print("m is", num, minutesNeeded(num))
```

Testing Laundry – three ways

3) Run in Python Console
   - Must import `Laundry`
   - Must specify which file `minutesNeeded` is in

   ```python
   >>> import Laundry
   >>> minutesNeeded(2)
   Traceback (most recent call last):
   File "C:\Users\Susan\AppData\Local\Programs\Python\python.exe", line 1, in <module>
   NameError: name 'minutesNeeded' is not defined
   >>> import Laundry
   >>> minutesNeeded(2)
   Traceback (most recent call last):
   File "C:\Users\Susan\AppData\Local\Programs\Python\python.exe", line 1, in <module>
   NameError: name 'minutesNeeded' is not defined
   >>> import math
   >>> math.sin(0.0)
   0.0
   ```

Where to put/use what in Python file

- Top: docstring with date and username
- Function definitions right after docstring
- Test code inside `if __name__ == '__main__':`

- Variables inside vs outside a function
  - **Only** use the variables inside that function
  - Therefore, **do not** use the variables outside the function (like in the main)
    - Your code will not work on the server