CSC 111:
Intro to Computer Science through Programming

Spring 2017
Prof. Sara Mathieson

TODO

- Sit somewhere new! Sit by someone you haven't met yet
- Today we'll be doing a few more structured programming exercises, so if you can use your laptop or sit next to someone who has theirs today, that would be great

- Make sure you can access Piazza (email me if not)
- Make sure you can access the website (first lab will be there)
- Make sure you can access Moodle (to submit lab/homework)

Reminders: Office Hours

- Office hours tomorrow (Thursday) 11am-1pm in Ford 015
- TA hours every night Sun-Thurs 7:30-9:30pm in Ford 241
- Labs start today/tomorrow (due Friday)
- Homework 1 due Tuesday Feb 7

Outline: 2/1

- Recap last time
- Strings, range, round
- Function practice (conversions)
- Debugging
- Creating a transcript

Recap

Code from last time

```
# File: chaos.py (Zelle p.15)
# A simple program illustrating chaotic behavior.

def main():
    print( 'This program illustrates a chaotic function' )
    x = eval( input( 'Enter a number between 0 and 1: ' ) )
    for i in range(10):
        x = 3.9 * x * (1 - x)
        print( x )

main()
```

Informal quiz (discuss with a partner)

```
# File: chaos.py (Zelle p.15)
# A simple program illustrating chaotic behavior.

def main():
    print( 'This program illustrates a chaotic function' )
    x = eval( input( 'Enter a number between 0 and 1: ' ) )
    for i in range(10):
        x = 3.9 * x * (1 - x)
        print( x )

main()
```

1) What do the hashtags at the beginning denote?
2) List the built-in functions in this code
3) What is the module name?
4) What is the name of the function in this module?
5) What is the variable i being used for?
6) (open ended) Is an algorithm like a recipe? Why or why not?

Informal quiz (discuss with a partner)

```
# File: chaos.py (Zelle p.15)
# A simple program illustrating chaotic behavior.

def main():
    print( 'This program illustrates a chaotic function' )
    x = eval( input( 'Enter a number between 0 and 1: ' ) )
    for i in range(10):
        x = 3.9 * x * (1 - x)
        print( x )

main()
```

1) What do the hashtags at the beginning denote? **comments**
2) List the built-in functions in this code
3) What is the module name?
4) What is the name of the function in this module?
5) What is the variable i being used for?
6) (open ended) Is an algorithm like a recipe? Why or why not?

Informal quiz (discuss with a partner)

```
# File: chaos.py (Zelle p.15)
# A simple program illustrating chaotic behavior.

def main():
    print( 'This program illustrates a chaotic function' )
    x = eval( input( 'Enter a number between 0 and 1: ' ) )
    for i in range(10):
        x = 3.9 * x * (1 - x)
        print( x )

main()
```

1) What do the hashtags at the beginning denote? **comments**

2) List the built-in functions in this code **print, eval, input, range**

3) What is the module name?

4) What is the name of the function in this module?

5) What is the variable i being used for?

6) (open ended) Is an algorithm like a recipe? Why or why not?

Informal quiz (discuss with a partner)

```
# File: chaos.py (Zelle p.15)
# A simple program illustrating chaotic behavior.

def main():
    print( 'This program illustrates a chaotic function' )
    x = eval( input( 'Enter a number between 0 and 1: ' ) )
    for i in range(10):
        x = 3.9 * x * (1 - x)
        print( x )

main()
```

1) What do the hashtags at the beginning denote? **comments**

2) List the built-in functions in this code **print, eval, input, range**

3) What is the module name? **chaos**

4) What is the name of the function in this module?

5) What is the variable i being used for?

6) (open ended) Is an algorithm like a recipe? Why or why not?

Informal quiz (discuss with a partner)

```
# File: chaos.py (Zelle p.15)
# A simple program illustrating chaotic behavior.

def main():
    print( 'This program illustrates a chaotic function' )
    x = eval( input( 'Enter a number between 0 and 1: ' ) )
    for i in range(10):
        x = 3.9 * x * (1 - x)
        print( x )

main()
```

1) What do the hashtags at the beginning denote? **comments**

2) List the built-in functions in this code **print, eval, input, range**

3) What is the module name? **chaos**

4) What is the name of the function in this module? **main**

5) What is the variable i being used for?

6) (open ended) Is an algorithm like a recipe? Why or why not?

Informal quiz (discuss with a partner)

```
# File: chaos.py (Zelle p.15)
# A simple program illustrating chaotic behavior.

def main():
    print( 'This program illustrates a chaotic function' )
    x = eval( input( 'Enter a number between 0 and 1: ' ) )
    for i in range(10):
        x = 3.9 * x * (1 - x)
        print( x )

main()
```

1) What do the hashtags at the beginning denote? **comments**

2) List the built-in functions in this code **print, eval, input, range**

3) What is the module name? **chaos**

4) What is the name of the function in this module? **main**

5) What is the variable i being used for? **keeps track of the iteration of the for loop**

6) (open ended) Is an algorithm like a recipe? Why or why not?

Is an algorithm like a recipe?

Is an algorithm like a recipe?

- Textbook says yes

Is an algorithm like a recipe?

- Textbook says yes
- Does this incorporate the idea of parameters? (changeable parts)

Is an algorithm like a recipe?

- Textbook says yes
- Does this incorporate the idea of parameters? (changeable parts)
- Parameters are ingredients? Flexible substitutions?

Is an algorithm like a recipe?

- Textbook says yes
- Does this incorporate the idea of parameters? (changeable parts)
- Parameters are ingredients? Flexible substitutions?
- Process of putting parameters together is like the set of recipe instructions?

Is an algorithm like a recipe?

- Textbook says yes
- Does this incorporate the idea of parameters? (changeable parts)
- Parameters are ingredients? Flexible substitutions?
- Process of putting parameters together is like the set of recipe instructions?
- Programming: turning an algorithm into software/code

Strings, range, round

Strings

Image: Rainbow * Works

- Idea: like a string of beads
- For us: beads are like different characters
- Examples: “hello”, “there”, “Smith College”, “0.5”
- Important string information: it’s length
- To print many strings together, two different ways
  - + operator (concatenates strings together)
  - , (comma) between different strings

Strings

Image: Rainbow * Works

Try out the following commands in the IDLE shell:

Commands:

```
>>> word1 = "hello"
>>> word2 = "there"
>>> print(word1 + word2)
>>> print(word1, word2)
>>> print(word1, "!")
>>> print(word1 + "!")
```

Strings

Image: Rainbow * Works

Try out the following commands in the IDLE shell:

Commands:

```
>>> word1 = "hello"
>>> word2 = "there"
>>> print(word1 + word2)
>>> print(word1, word2)
>>> print(word1, "!")
>>> print(word1 + "!")
```

Output:

```
hellothere
hello there
hello !
hello!
```

Strings

Image: Rainbow * Works

Try out the following commands in the IDLE shell:

Commands:

```
>>> word1 = "hello"
>>> word2 = "there"
>>> print(word1 + word2)
>>> print(word1, word2)
>>> print(word1, "!")
>>> print(word1 + "!")
```

Output:

```
hellothere
hello there
hello !
hello!
```

Commands:

```
>>> len(word1)
>>> len("!")
>>> len("")
>>> len(word1 + word2)
```

Strings

Image: Rainbow * Works

Try out the following commands in the IDLE shell:

Commands:

```
>>> word1 = "hello"
>>> word2 = "there"
>>> print(word1 + word2)
>>> print(word1, word2)
>>> print(word1, "!")
>>> print(word1 + "!")
```

Output:

```
hellothere
hello there
hello !
hello!
```

Commands:

```
>>> len(word1)
>>> len("!")
>>> len("")
>>> len(word1 + word2)
```

Output:

```
5
1
0
10
```

Range (flexible number of parameters)

- range(**stop**)
  - Start = 0
  - Excludes **stop**
  - Step = 1

Range (flexible number of parameters)

- range(**stop**)
  - Start = 0
  - Excludes **stop**
  - Step = 1
- range(**start**, **stop**)
  - Includes **start**
  - Excludes **stop**
  - Step = 1

Range (flexible number of parameters)

- range(**stop**)
  - Start = 0
  - Excludes **stop**
  - Step = 1
- range(**start**, **stop**)
  - Includes **start**
  - Excludes **stop**
  - Step = 1
- range(**start**, **stop**, **step**)
  - Includes **start**
  - Excludes **stop**
  - Flexible **step** size

Range (flexible number of parameters)

- range(**stop**)
  - Start = 0
  - Excludes **stop**
  - Step = 1
- range(**start**, **stop**)
  - Includes **start**
  - Excludes **stop**
  - Step = 1
- range(**start**, **stop**, **step**)
  - Includes **start**
  - Excludes **stop**
  - Flexible **step** size

Try out:

```
>>> for i in range(7):
        print(i)
```

What calls to range would produce the following lists of numbers?

```
0
1
2
```

```
87
88
89
90
91
```

```
30
32
34
36
38
40
```

Range (flexible number of parameters)

- range(**stop**)
  - Start = 0
  - Excludes **stop**
  - Step = 1
- range(**start**, **stop**)
  - Includes **start**
  - Excludes **stop**
  - Step = 1
- range(**start**, **stop**, **step**)
  - Includes **start**
  - Excludes **stop**
  - Flexible **step** size

Try out:

```
>>> for i in range(7):
        print(i)
```

What calls to range would produce the following lists of numbers?

```
>>> for i in range(3):
        print(i)
```

```
0
1
2
```

```
87
88
89
90
91
```

```
30
32
34
36
38
40
```

Range (flexible number of parameters)

- range(**stop**)
  - Start = 0
  - Excludes **stop**
  - Step = 1
- range(**start**, **stop**)
  - Includes **start**
  - Excludes **stop**
  - Step = 1
- range(**start**, **stop**, **step**)
  - Includes **start**
  - Excludes **stop**
  - Flexible **step** size

Try out:

```
>>> for i in range(7):
        print(i)
```

What calls to range would produce the following lists of numbers?

```
>>> for i in range(3):
        print(i)
```

```
0
1
2
```

```
>>> for i in range(87,92):
        print(i)
```

```
87
88
89
90
91
```

```
30
32
34
36
38
40
```

Range (flexible number of parameters)

- range(**stop**)
  - Start = 0
  - Excludes **stop**
  - Step = 1
- range(**start**, **stop**)
  - Includes **start**
  - Excludes **stop**
  - Step = 1
- range(**start**, **stop**, **step**)
  - Includes **start**
  - Excludes **stop**
  - Flexible **step** size

Try out:

```
>>> for i in range(7):
        print(i)
```

What calls to range would produce the following lists of numbers?

```
>>> for i in range(3):
        print(i)
```

```
0
1
2
```

```
>>> for i in range(87,92):
        print(i)
```

```
87
88
89
90
91
```

```
>>> for i in range(30,41,2):
        print(i)
```

```
30
32
34
36
38
40
```

Round

- New built-in function to round a number to the nearest integer
- Try out the commands:

```
>>> round(0.6)
>>> round(0.5)
>>> round(0.51)
>>> round(3.4)
>>> round(3)
>>> x = 0.3
>>> round(x)
>>> y = round(x)
>>> y
>>> x
```

Round

- New built-in function to round a number to the nearest integer
- Try out the commands:

```
>>> round(0.6)  → 1
>>> round(0.5)  → 0
>>> round(0.51) → 1
>>> round(3.4)  → 3
>>> round(3)    → 3
>>> x = 0.3
>>> round(x)    → 3
>>> y = round(x)
>>> y           → 0
>>> x           → 0.3
```

Function practice (conversion)

Celsius to Fahrenheit conversion

- With a partner, complete the following function started below (don't use the textbook!)
- What is the error?
- Multiply Celsius temperature by 9, then divide by 5, then add 32

```
def main():
    celsius = eval(input("Enter temperature in celsius: "))
    print(celcius)

main()
```

Celsius to Fahrenheit conversion

```
# CSC 111, Day 3
# Author: Sara Mathieson and CSC 111 class
# Program to convert from Celsius to Fahrenheit

def main():
    celsius = eval(input("Enter temperature in celsius: "))
    fahrenheit = round(9/5 * celsius + 32)
    print("The temperature is", fahrenheit, "degrees Fahrenheit.")

main()
```

Demo:
debugging and transcript