CS21: INTRODUCTION TO COMPUTER SCIENCE

Prof. Mathieson
Fall 2017
Swarthmore College
String accumulator: warmup to start now (with a *new* partner!)

1) update21

2) cd cs21/practice/

3) atom stretch.py

4) Complete the program (*hint: when we were accumulating a number we first set it equal to 0. When we are accumulating a string, what should be the initial value?*)

Finish early or did last time?
- practice/star_string.py
- inclass/week03/username.py
- inclass/week03/telephone.py
stretch.py example solution

For loop and accumulator pattern practice.
Write a program that that "stretches" a string by doubling every character.
Let the user input any string. For example:

```
python3 stretch.py
Enter a string: weekend
Stretch string: wweekkeenndd
```

Author: Sara Mathieson
Date: 9/18/17

```python
def main():
    string = input("Enter a string: ")
    doubled = ""
    for ch in string:
        doubled = doubled + ch + ch # or ch*2
    print("Strech string:", doubled)

main()
```
Outline Sept 18:

• Accumulator pattern with strings
• Random library
• Boolean types
• Comparison operators
• First conditionals

Notes

• Lab 2 due Saturday night
• Quiz 1: next Friday (9/22), let me know about conflicts
• Practice problems in the practice directory (try on paper first)
Recap average program with randomness
Random library (similar to math)

```python
import random

def main():
    # initialize a sequence (list, range, or string)
    coin = ["H", "T"]

    # choose randomly from the sequence
    flip = random.choice(coin)

    # display the output
    print("You flipped", flip)

main()
```
Rolling a die many times, compute the average

- cd cs21/inclass/week03/
- atom random_average.py

Incremental design:
- Start by printing one random number
- Use a for loop to print many random numbers
- Use the accumulator pattern to compute the sum of all the random numbers
- After the loop, compute the average
Booleans and comparison operators
New type: booleans

- Can only be True or False
- The result of a logical expression
- Comparison or relational operators
  - Less than: `<`
  - Greater than: `>`
  - Less than or equal to: `<=`
  - Greater than or equal to: `>=`
  - Is equal to: `==`
  - Is not equal to: `!=`

```
>>> 3 < 5
True
>>> 7 > 10
False
>>> 3.0 == 3
True
>>> x = "hello"
>>> x != "hello"
True
>>> x == "hello"
True
```
New control statements: if/elif/else

- Idea: if something is true, we want one thing to happen
- If something is false, we want another thing to happen
- Example:

  If it is raining:
  I will wear rain boots
  If it is not raining:
  I will wear sandals
New control statements: if/elif/else

• Control statements:
  
  • 1) Functions (keyword: `def`, then indent afterwards)
  • 2) For-loops (keyword: `for`, then indent afterwards)
  • 3) If-statements (keyword: `if`, then indent afterwards)

• If-statement syntax:
  
  • `if <condition>:`
    `<statements>  # Executed if <condition> is True`
New control statements: if/elif/else

Example: based on class year, has a student graduated or not?

```python
if <condition1>:
    <statements_1>  # Executed if <condition1> is True

elif <condition2>:
    <statements_2>  # Executed if <condition1> is False and <condition2> is True

else:
    <statements_3>  # Executed if <condition1> and <condition2> are both False
```

conditionals1.py
First programs with conditionals. Write a program that asks the user for their favorite integer. Then print out whether the integer is positive, negative, or zero. For example:

```python
python3 number_conditionals.py
Enter your favorite integer: -4
Your number is negative!
```

Author: Sara Mathieson
Date: 9/17/17

```python
def main():
    number = int(input("Enter your favorite integer: "))

    if number > 0:
        print("Your number is positive!")
    elif number < 0:
        print("Your number is negative!")
    else:
        print("Your number is zero!")

main()
```