

Lab 8

- Reading, writing files
- Modules
- Exception Handling
- Using lists to solve problems

Lab 8: Pair Programming

Alice	Andrew	Hayden	Callie
Andrew	Alice	Jake	Kassi
August	Ellis	James	Mike
Bobby	Karel	Jenna	Natalie
Callie	Hayden	Karel	Bobby
Cat	Giovanni	Kassi	Jake
Charlotte	Nate	Laurie	Dan
Dan	Laurie	Matt	Melissa
Danielle	Danny	Melissa	Matt
Danny	Danielle	Mike	James
Ellis	August	Natalie	Jenna
Giovanni	Cat	Nate	Charlotte

Pair Programming

- There are two of you, so
 - Double check problem specifications
 - Push each other: better tests, comments, var names, ...
 - Iterate
- Discuss
 - What is the role of the navigator?
 - What is the role of the driver?
 - What has worked well in your previous pairs?
 - What would you like to do more of?
 - If you haven't felt that the balance between teammates is 50/50, what can you do to change that?

Compare Solutions

```
words = sentence.split()

shorthandList = []
for word in words:
    shorthandList.append(word[0])

shorthand = "".join(shorthandList)
shorthand = shorthand.lower()
print("Shorthand is:", shorthand)
```

```
words = sentence.split()

shorthand = ""
for word in words:
    shorthand += word[0]

shorthand = shorthand.lower()
print("Shorthand is:", shorthand)
```

Compare Solutions

```
words = sentence.split()

shorthandList = []
for word in words:
    shorthandList.append(word[0])

shorthand = "".join(shorthandList)
shorthand = shorthand.lower()
print("Shorthand is:", shorthand)
```

In general, strive for less complex solutions.

Both are valid solutions.
I'm not sure which is more efficient in practice.

However, the solution at left has more conceptual complexity (appending to a list and then converting to a string, as opposed to just creating the string).

```
words = sentence.split()

shorthand=""
for word in words:
    shorthand += word[0]

shorthand = shorthand.lower()

print("Shorthand is:", shorthand)
```

Comment Example

```
def encodeLetter(letter, key):
    """Encodes a single letter.
    PRE: Input parameters are a single, lowercase
    character string (char) and an integer key
    (between -25 and 25, inclusive)
    POST: returns the encoded character as a str"""
```

- Does not say *who* called function, where parameters came from, or where returned to
 - Any code can call the function and pass in input from anywhere (e.g., hardcoded, from user input, test function, ...)
- Does not say variable name returned

Review Caesar Cipher

- Consider the following solutions

```
for char in message:
    asciiVal = ord(char)
    if asciiVal == 32:
        ...
    else:
        ...
```

```
for char in message:
    if char == " ":
        ...
    else:
        ...
```


Review Caesar Cipher

- Consider the following solutions

```
for char in message:
    asciiVal = ord(char)
    if asciiVal == 32:
        ...
    else:
        ...
```

I know what " " means.
I don't immediately know
what 32 means.
**Lesson: prefer words
over numbers.**

```
for char in message:
    if char == " ":
        ...
    else:
        ...
```



Review: Cleaning Up Data

Focus: Newline characters, Formatting

- **Given:** a CSV file containing students' names and their class according to the Registrar
- **Problem:** This file has TMI
 - Just want the last name and the class year
 - Instead of Ugr:Sophomore, say "Sophomore"
- **Solution:**
 - Read through file "data/years.csv", clean up data
 - Write the cleaned up data to a new file called "data/roster.txt"
 - 1st iteration: lastname class
 - 2nd iteration: nice tables of data

`cleanRoster.py`

March 12, 2019

Sprenkle - CSCI111

9

Review

- What are things we can do to lists?
- How do we work with files?
- What is the structure we use to do exception handling?
- How do we create a module?

March 12, 2019

Sprenkle - CSCI111

10

Lab 8 Overview

- Modules
- Reading Files
- Writing Files
- Exception handling
- Functions/Lists