

Lab 7

- Feedback on Lab 6
- Review for Lab 7

1

Lab Musings

- As we learn more computer science, we're moving toward a **much higher ratio of *thinking* to *coding***
 - Give yourself the time and room to think
 - Discuss, reinforce your understanding
- Going beyond simply correctness in solutions
 - Looking for understanding of good coding practices
 - Testing, readability, usability, documentation, organization, efficiency
 - (not necessarily in that order)

2

Lab Musings

- Lab benefit: access to lab assistants and instructor to help
- Lab limitation: may not be the best environment
 - Seems to cause a competitive atmosphere, increased anxiety for some students
 - You have until Friday to complete the lab
 - Work at your pace, **think clearly** and **deeply**

Mar 7, 2023

Sprenkle - CSCI111

3

3

LAB 6 FEEDBACK

Mar 7, 2023

Sprenkle - CSCI111

4

4

Inefficiency in while loops

```
num = 0
while num < 500 or num > 1000:
    num = eval(input("What is your number?"))

    if num <= 1000 and num >= 500:
        print("Eureka!")
    else:
        print("Please try again.")
```

Written as a hybrid between
“when should I stop?” and “when should I keep going?”

Know that the while loop’s condition will *never* be false
→ Doing an extra check every time through loop

Mar 7, 2023

Sprenkle - CSCI111

5

5

As a while True loop

```
num = 0
while True:
    num = eval(input("What is your number?"))

    if num <= 1000 and num >= 500:
        print("Eureka!")
    else:
        print("Please try again.")
```

Mar 7, 2023

Sprenkle - CSCI111

6

6

Using the Sentinel Design Pattern

```
num = eval(input("What is your number?")) Initialize value
while num < 500 or num > 1000: Sentinel check
    print("Please try again.")
    num = eval(input("What is your number?")) Update value
print("Eureka!") After loop completes,
you know you have your number
```

Mar 7, 2023

Sprenkle - CSCI111

7

7

Inefficiency in Craps

```
while True:
    if roll == 7 or roll == 11:
        ...
    elif roll == 2 or ...:
        ...
    else:
        point = roll
    ...
```

These steps only happen once, so they should *not* be in the `while` loop.

We can add code to ensure that they only execute once, but it's easier/less error-prone to not have them in the loop at all.

Mar 7, 2023

Sprenkle - CSCI111

8

8

Reminder: doc strings on all* functions

- Template:
 - What function does
 - Precondition: what parameters are, their types, any restrictions on them
 - Postcondition: what is true after function executes, e.g., what is returned or displayed

Mar 7, 2023

Sprenkle - CSCI111

* Not main or test functions

9

Programmatically Testing Functions

- Trying to help you become more efficient testers
 - Don't worry about user input
 - Just make the test calls
 - Think about input and expected output
- Example: `test.testEqual(stretchString("cs"), "c.s..")`
- Can still print in function during debugging
 - Then remove print statements

Mar 7, 2023

Sprenkle - CSCI111

10

10

Checking if a str contains a substring

Instead of using a method, could use **in** operator because didn't care where in the string it was:

```
if "r" in phrase:
```

Over string

- Why do you **not** need to use `str` in the following code segment?

```
origString = str( input("What is your string? ") )
```

Over string

- Why do you **not** need to use `str` in the following code segment?

```
origString = str( input("What is your string? ") )
```

➤ Because `input` returns a string; no need to cast

- Preferred:

```
origString = input("What is your string? ")
```

Goal: Simplify/reduce code
→ Less code → easier to understand, less error-prone

Mar 7, 2023

Sprenkle - CSCI111

13

13

When to Compute

- Don't do computation until it is needed

```
fileName = input("What is the name of your file? ")
lastPeriod = fileName.rfind(".")
if lastPeriod != -1 and lastPeriod != len(fileName) - 1 :
    extension = fileName[lastPeriod+1:] Find extension here
    print(fileName, "is a(n)", extension, "file.")
else:
    print(fileName, "does not have an extension")
```

Mar 7, 2023

Sprenkle - CSCI111

14

14

Adding to Development Process

- After your program works, consider
 - Is it readable?
 - Can I simplify?
 - Is it efficient?
- Modify, test again

Mar 7, 2023

Sprenkle - CSCI111

15

15

Problem Solving: Smaller Pieces

We implement and test functions separately from the user interaction

User Interaction

- Easy to implement
- Shows results
- But slows down development process

Test Function

- Focus on the “hard” part (typically, the new stuff we’re learning)
- Speeds up development process
 - No need to type in input (and make mistakes)
- But, can’t see progress as easily

Easy fix: print out result of the called function with hard-coded parameter(s)

Mar 7, 2023

Sprenkle - CSCI111

16

16

Problem Solving: Smaller Pieces

- Example development process for myFunction

1. Initial development

```
print(myFunction("example"))
```

Problem Solving: Smaller Pieces

- Example development process for myFunction

1. Initial development

```
print(myFunction("example"))
```

2. Add tests in test function

```
print(myFunction("example"))  
testMyFunction()
```

Problem Solving: Smaller Pieces

- Example development process for myFunction

1. Initial development

```
print(myFunction("example"))
```

2. Add tests in test function

```
print(myFunction("example"))  
testMyFunction()
```

3. Add in user input in main

```
main()  
#testMyFunction()
```

Mar 7, 2023

Sprenkle - CSCI111

19

19

str Review

Review on your own

- How can we combine strings?
 - How can you repeat a string multiple times?
- How can we find out how long a string is?
- How can you tell if one string is contained in another string?
- How can we find out the character at a certain position?
- How can we iterate through a string?
- How do you call a method on a string?
- Can we change a string after it's been initialized?

Mar 7, 2023

Sprenkle - CSCI111

20

20

Review

- What is the special name for characters that start with “\”?
 - Give examples of those special characters
- How do you format strings?
 - What does a format specifier look like?
 - What questions should you ask when formatting strings/creating the format specifier?
- How can we find the ASCII value for a character?
- How can we find the character associated with an ASCII value?
- How can we check if a character belongs to a character *class*? (reviewing from textbook)
- How do you encode a letter in the Caesar Cipher?

Review: String Formatting

- Use the format method
 - `"templatestring".format(replacementvalues)`
- Format specifiers syntax:
`{[flags][width][.precision][code]}`
- When determining format specifiers, consider
 - Data type of the replacement value
 - If float, how many decimal places desired
 - Desired width
 - Justification, other flags

Review: Translating to/from ASCII

- Translate a character into its ASCII numeric code using **built-in function ord**
 - `ord('a')` ==> 97
- Translate an ASCII numeric code into its character using **built-in function chr**
 - `chr(97)` ==> 'a'

Mar 7, 2023

Sprenkle - CSC1111

[ascii_table.py](#)
[ascii.py](#)

23

23

Review: Character Classification

- From string module
- Classifications:
 - `string.ascii_lowercase`
 - `string.ascii_uppercase`
 - `string.digits`
 - `string.punctuation`

Mar 7, 2023

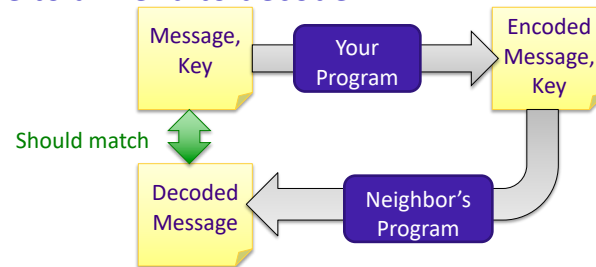
Sprenkle - CSC1111

24

24

Caesar Cipher

- Write an encoding/decoding program
 - Encode a message
 - Give to a friend to decode



Mar 7, 2023

Sprenkle - CSCI111

25

25

What is the algorithm for encoding a letter?

- Assuming a lowercase letter
- Test Cases:
 - `test.assertEqual(encryptLetter('a', 1), 'b')`
 - `test.assertEqual(encryptLetter('y', 1), 'z')`
 - `test.assertEqual(encryptLetter('z', 5), 'e')`
 - `test.assertEqual(encryptLetter('b', -4), 'x')`

Mar 7, 2023

Sprenkle - CSCI111

26

26

What is the algorithm for encoding a letter?

(Assuming a lowercase letter)

1. Convert the character to its ASCII value
2. Add the key to that value
3. Make sure that the new value is a “valid” ASCII value, i.e., that that new value is in the range of lowercase letter ASCII values
 - a. If not, “wrap around” to adjust that value so that it’s in the valid range
4. Convert the ASCII value into a character

Mar 7, 2023

Sprenkle - CSCI111

27

27

What is the algorithm for encoding a message?

- Assuming message made up of only lowercase letters, spaces, and punctuation

- Examples:

```
test.testEqual(encryptMessage('cat', 1), 'dbu' )  
test.testEqual(encryptMessage('w and l!', 5), 'b fsi q!' )
```

Mar 7, 2023

Sprenkle - CSCI111

28

28

Encrypt a Message

- Accumulate a new encrypted message
- For each character in the message
 - Check if the character is a space or punctuation; if it is, it doesn't change
 - Add character to the encrypted message
 - Otherwise
 - Encrypt letter
 - Add encoded letter to the encrypted message

We need to accumulate the encrypted message in a new string rather than change the message because strings are *immutable*

Mar 7, 2023

Sprenkle - CSCI111

29

29

Honor System Review

- Person who needs help should *never* look at the code of the person who is helping
- No sharing code
 - No emailing, printing, ...
- Cite the help you're receiving outside of lab
- Pledge your assignments
- Report suspicious behavior

Mar 7, 2023

Sprenkle - CSCI111

30

30

Lab 7

- Caesar Cipher
- Strings
 - Escape sequences
 - Formatting