

Objectives

- Encryption
- **str** methods

Feb 19, 2010

Sprenkle - CSCI111

1

Lab 5 Animations

- New Barbie
- Negative reinforcement

Feb 19, 2010

Sprenkle - CSCI111

2

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'`

Feb 19, 2010

Sprenkle - CSCI111

`ascii_table.py`
`ascii.py`

3

Encryption

- Process of encoding information to keep it secure
- One technique: Substitution Cipher
 - Each character in message is replaced by a new character

Feb 19, 2010

Sprenkle - CSCI111

4

Caesar Cipher

- Replace with a character X places away
 - X is your *key*
- Julius Caesar used it to communicate with his generals
- “Wrap around”
- Write program(s) to do this in next lab

Feb 19, 2010

Sprenkle - CSCI111

5

Caesar Cipher

- Using the ASCII handout, what would be the encoded messages?

| Message | Key | Encoded Message |
|-----------------------------|-----|-----------------|
| apple | 5 | |
| zebra | 5 | |
| the eagle flies at midnight | -5 | |

Feb 19, 2010

Sprenkle - CSCI111

6

Caesar Cipher

| Message | Key | Encoded Message |
|-----------------------------|-----|----------------------------|
| apple | 5 | fuuqj |
| zebra | 5 | ejgwf |
| the eagle flies at midnight | -5 | ocz zvbz agdzn vo hdyidbco |

What is your algorithm for the encoding process?
How would you *decode* an encrypted message?

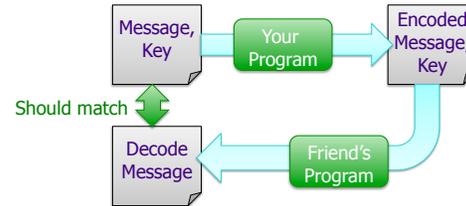
Feb 19, 2010

Sprenkle - CSCI111

7

Next Lab

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



Feb 19, 2010

Sprenkle - CSCI111

8

USING THE STR API

Feb 19, 2010

Sprenkle - CSCI111

9

str Methods

- **str** is a **class** or a **type**
- **Methods**: available operations to perform on **str** objects
 - Provide common functionality
- To see all methods available for **str** class
 - `help(str)`

Feb 19, 2010

Sprenkle - CSCI111

10

str Methods

- Example method: **find(substring)**
 - Finds the index where substring is in string
 - Returns -1 if substring isn't found
- To call a method:
 - `<string>.methodname([arguments])`
 - Example: `filename.find(".py")`

Executed on this string

Feb 19, 2010

Sprenkle - CSCI111

11

Common str Methods

| Method | Operation |
|--|--|
| <code>center(width)</code> | Returns a copy of string centered within the given number of columns |
| <code>count(sub[, start [, end]])</code> | Return # of non-overlapping occurrences of substring <code>sub</code> in the string. |
| <code>endswith(sub), startswith(sub)</code> | Return True iff string ends with/begins with <code>sub</code> |
| <code>find(sub[, start [, end]])</code> | Return first index where substring <code>sub</code> is found |
| <code>isalpha(), isdigit(), isspace()</code> | Returns True iff string contains letters/digits/whitespace only |
| <code>lower(), upper()</code> | Return a copy of string converted to lowercase/uppercase |

Feb 19, 2010

Sprenkle - CSCI111 [string_methods.py12](#)

Common str Methods

| Method | Operation |
|---|---|
| <code>replace(old, new[, count])</code> | Returns a copy of string with all occurrences of substring old replaced by substring new . If count given, only replaces first count instances. |
| <code>split([sep])</code> | Return a list of the words in the string, using sep as the delimiter string. If sep is not specified or is None, any whitespace string is a separator. |
| <code>strip()</code> | Return a copy of the string with the leading and trailing whitespace removed |
| <code>join(<sequence>)</code> | Return a string which is the concatenation of the strings in the sequence with the string this is called on as the separator |
| <code>swapcase()</code> | Return a copy of the string with uppercase characters converted to lowercase and vice versa. |

Feb 19, 2010

Sprengle - CSCI111

13

Functions vs Methods

Functions

- All "input" as arguments/ parameters
- Example: `len` is a built-in function
 - Called as `len(string)`

Methods

- "Input" are argument/ parameters **and** the string the method was called on
 - Example:
 - `string.upper()`

Feb 19, 2010

Sprengle - CSCI111

14

Using str Methods

- Modify our search program to find out if the entered string has the `.py` extension

```
PYTHON_EXT = ".py"

filename = raw_input("Enter a filename: ")

if filename[-(len(PYTHON_EXT)):] == PYTHON_EXT:
    # Appropriate output
if PYTHON_EXT in filename:
    # Appropriate output
```

Feb 19, 2010

Sprengle - CSCI111

15

Are You Smarter Than a 5th Grader?

- Problem in spelling from the show: How many a's are in `abracadabra`?
 - Solve using **str** methods

Feb 19, 2010

Sprengle - CSCI111

16

Get the Username

- Given the directory formatted as
 - `dir = "/home/www/users/username/"`
- Get the username out

Feb 19, 2010

Sprengle - CSCI111

17

Using str Methods

- Modify `binaryToDecimal.py` to verify that the entered string contains only numbers
 - Keep asking them for a number until the string contains only numbers

Feb 19, 2010

Sprengle - CSCI111

18

Using str Methods

- Modify binaryToDecimal.py to verify that the entered string contains only numbers
 - Keep asking them for a number until the string contains only numbers
- 2nd modification: How could we make sure that entered string contains only 0s and 1s?

Feb 19, 2010

Sprenkle - CSCI111

19

Broader Issues in Computer Science

- Testing isn't a broader issue
 - Glad you noticed lots of the issues with testing
 - We'll keep talking about it because I love it!

Kelly Mae
Nick
George
Amy
James

CJ
Shannon
Sirocco
Harrison
Dave

Luke
Logan
Collier
Taylor

Dalena
Jeni
Hank
Andrew

Feb 19, 2010

Sprenkle - CSCI111

20

Broader Issues in Computer Science

- Is the Excel 2007 a "reasonable" bug?
 - Why wasn't it caught?
 - Should it have been caught?
- Are you more Pogue or Spolsky?
- When should a company stop testing?
 - When do **you** stop testing?
- Why doesn't software have guarantees?

Feb 19, 2010

Sprenkle - CSCI111

21

Broader Issues in Computer Science

- Have you ever encountered a bug in a program?
 - What happened?
 - How severe was the problem? Were you able to recover?
 - How did you respond? (Angry? Didn't think about? ...)
- If people can recover from a bug, when does it become important for software developers to fix the problem?
 - Tradeoffs between costs/revenues of implementing new features versus fixing existing code
 - What matters to you (as a consumer) more?

Feb 19, 2010

Sprenkle - CSCI111

22

Notes from a Keynote Speech about Testing Microsoft Vista

- Users are "trained" to not use buggy features
 - After user encounters a certain bug when doing something enough times, the user stops trying to do that buggy activity

$$\text{User's Loss in Confidence} = \text{Disruption Frequency} \times \begin{matrix} \text{Recovery Time} \\ \text{Recover Effort} \\ \text{Lost data} \\ \text{Uncertainty} \end{matrix}$$

- Only ship fixes that affect many users

Feb 19, 2010

Sprenkle - CSCI111

23

Relation to Our Class

- When do **you** stop testing?

Feb 19, 2010

Sprenkle - CSCI111

24

Status from Official Excel Blog

- Post on 9/25/07 Happy Ending
 - We've come up with a fix for this issue and are in the final phases of a broad test pass in order to ensure that the fix works and *doesn't introduce any additional issues* - especially any other calculation issues. This fix then needs to make its way through our official build lab and onto a download site - which we expect to happen very soon.
- Post on 10/9/07
 - As of today, fixes for this issue in Excel 2007 and Excel Services 2007 are available for download ...
 - We are in the process of adding this fix to Microsoft Update so that it will get *automatically pushed* to users running Excel 2007 or Excel Services 2007. Additionally, the fix will also be contained in the first service pack of Office 2007 when it is released (the release date for SP1 of Office 2007 has not been finalized).

Feb 19, 2010

Sprenkle - CSC1111

25

For Friday

- Read about Excel 2007 bug
- Lab 5

Feb 19, 2010

Sprenkle - CSC1111

26