One-on-One Sessions

• Starting Friday: 1/2-hour one-on-one sessions
  • Bring computer to work with instructor, TA or consultant
  • Hands on, dedicated help with Lab 3 (or next lecture)
  • To prepare for assignment, not for help on assignment

• Limited availability: we cannot get to everyone
  • Students with experience or confidence should hold back

• Sign up online in CMS: first come, first served
  • Choose assignment One-on-One
  • Pick a time that works for you; will add slots as possible
  • Can sign up starting at 1pm TODAY

9/6/16

String: Text as a Value

• String are quoted characters
  • ‘abc d’ (Python prefers)
  • "abc d" (most languages)

• How to write quotes in quotes?
  • Delineate with “other quote”
  • Example: ‘ ‘ or ' " '

• What if need both " and ’?
  • Solution: escape characters
    • Format: \ + letter
    • Special or invisible chars

9/6/16

Strings

String are Indexed

• s = 'abc d'
  0 1 2 3 4
  A B C D E
  • Access characters with []
    • s[0] is ‘a’
    • s[4] is ‘d’
    • s[5] causes an error
    • s[0:2] is 'ab' (excludes c)
    • s[2:] is 'c d'
  • Called “string slicing”

• s = 'Hello all'
  0 1 2 3 4 5 6 7 8
  A B C D E F G H I
  • What is s[3:6]?
    • A: 'lo a'
    • B: 'lo'
    • C: 'lo '
    • D: 'o '
    • E: I do not know

9/6/16

Other Things We Can Do With Strings

• Operation in: s1 in s2
  • Tests if s1 “a part of” s2
  • Say s1 a substring of s2
  • Evaluates to a bool

• Examples:
  • s = 'abraadabra'
  • 'a' in s == True
  • 'ad' in s == True
  • 'foo' in s == False

9/6/16

Function len: len(s)

• Value is # of chars in s
  • Evaluates to an int

• Examples:
  • s = 'abraadabra'
  • len(s) == 11
  • len(s[1:8]) == 4
  • s[1:len(s)-1] == 'bracadabr'

9/6/16

Defining a String Function

```python
def middle(text):
    # Get length of text
    size = len(text)
    # Start of middle third
    start = size // 3
    # End of middle third
    end = 2 * size // 3
    # Get the text
    result = text[start:end]
    # Return the result
    return result
```

9/6/16

Not All Functions Need a Return

```python
def greet(n):
    # Prints a greeting to the name n
    print('Hello ' + n + '!
    print('How are you?')
```

9/6/16

Strings

06/16

Strings

06/16

Strings

06/16

Strings
Procedures vs. Fruitful Functions

- **Procedures**
  - Functions that do something
  - Call them as a statement
  - Example: `greet('Walker')`

- **Fruitful Functions**
  - Functions that give a value
  - Call them in an expression
  - Example: `x = round(2.56, 1)`

**Historical Aside**
- Historically “function” = “fruitful function”
- But now we use “function” to refer to both

Print vs. Return

- **Print**
  - Displays a value on screen
  - Used primarily for testing
  - Not useful for calculations
  - Example:
    ```python
def print_plus(n):
    print(n+1)
    >>> x = print_plus(2)
    3
    >>>
    ```

- **Return**
  - Defines a function’s value
  - Important for calculations
  - But does not display anything
  - Example:
    ```python
def return_plus(n):
    return(n+1)
    >>> x = return_plus(2)
    >>>
    ```

Advanced String Features: Method Calls

- Methods calls are unique (right now) to strings
- Like a function call with a “string in front”
  - Usage: `string method(x, y...)`
  - The string is an implicit argument
- Example: `upper()`
  - `s = 'Hello World'`
  - `s.upper() == 'HELLO WORLD'`
  - `s[1:8].upper() == 'ELLO'`
  - `'abc'.upper() == 'ABC'`

Examples of String Methods

- `s1.index(s2)`
  - Position of the first instance of `s2` in `s1`
  - Example: `s1.index('a') == 0`
  - `s1.index('rac') == 2`

- `s1.count(s2)`
  - Number of times `s2` appears inside of `s1`
  - Examples: `s1.count('a') == 5`, `s1.count('b') == 2`, `s1.count('x') == 2`

- `s.strip()`
  - A copy of `s` with white-space removed at ends

String Extraction Example

```python
def firstparens(text):
    # Find the open parenthesis
    start = text.index('(')
    # Store part AFTER paren
    tail = text[start+1:]
    # Find the close parenthesis
    end = tail.index(')')
    # Return the result
    return tail[:end]
```

```python
def second(thelist):
    # Find second elt in thelist
    start = thelist.index(')')
    # Return result
    return thelist[start+1]`