Purpose of Today’s Lecture

• Return to the string (str) type
  * Saw it the first day of class
  * Learn all of the things we can do with it
• See more examples of functions
  * Particularly functions with strings
• Learn the difference between
  * Procedures and fruitful functions
  * print and return statements

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

Char | Meaning
--- | ---
'| single quote
"| double quote
| new line
\| tab
\\| backslash

String are Indexed

• s = ’abc d’
  0 1 2 3 4
  | a | b | c | d |
• 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”

Other Things We Can Do With Strings

• Operation in: s_1 in s_2
  * Tests if s_1 “a part of” s_2
  * Say s_1 a substring of s_2
• Function len: len(s)
  * Value is # of chars in s
• Examples:
  * s = ‘abraadabra’
  * ‘a’ in s == True
  * ‘cad’ in s == True
  * ‘foo’ in s == False

Defining a String Function

• Start w/ string variable
  * Holds string to work on
  * Make it the parameter
• Body is all assignments
  * Make variables as needed
  * But last line is a return
• Try to work in reverse
  * Start with the return
  * Figure ops you need
  * Make a variable if unsure
  * Assign on previous line

Not All Functions Need a Return

```python
def middle(text):
    """Returns: middle 3\text{rd} of text"
    Param text: a string"
    # 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
```

```python
def greet(n):
    """Prints a greeting to the name n"
    Parameter n: name to greet
    Precondition: n is a string"
    print('Hello '+n+'!')
    print('How are you?')
```
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)
  >>> x
  3
  >>>
  >>> x = return_plus(2)
  >>> x
  Nothing here!
  ```

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[1:5].upper() == 'ELLO'`
  - `'abc'.upper() == 'ABC'`

Will see why we do it this way later in course

Examples of String Methods

- `s1.index(s2)`
  - Position of the first instance of `s2` in `s1`
- `s1.count(s2)`
  - Number of times `s2` appears inside of `s1`
- `s.strip()`
  - A copy of `s` with white-space removed at ends

See Python Docs for more

String Extraction Example

```python
def firstparens(text):
    """Returns: substring in ()
    Uses the first set of parens
    Param text: a string with ()""
    # Find the open parenthesis
    start = s.index('(')
    # Store part AFTER paren
    tail = s[start+1:]
    # Find the close parenthesis
    end = tail.index(')')
    # Return the result
    return tail[:end]
```

```python
>>> s = 'Prof (Walker) White'
>>> firstparens(s)
'Walker'
```

```python
def second(thelist):
    """Returns: second elt in thelist
    The list is a sequence of words separated by commas, spaces.
    Ex: second(A, B, C) => B
    Param thelist: a list of words""
    start = thelist.index("
    tail = thelist[start+1:]
    end = tail.index("
    result = tail[end]
    return result
```

String Extraction Puzzle

```python
>>> second('cat, dog, mouse, lion')
'dog'
```

```python
>>> second('apple, pear, banana')
'pear'
```

```python
1
2
3
4
5
```