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



Type: str

String are Indexed

- s = 'abc d'
- 0 1 2 3 4 a b c d
- Access characters with []
 What is s[3:6]?
 - 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 H e 1 1 o a 1 1
- - A: 'lo a'
 - B: 'lo'
 - C: 'lo '
 - D: 'o '
 - E: I do not know

Other Things We Can Do With Strings

- Operation in: s_1 in s_2
 - Tests if s₁ "a part of" s₂
 - Say s₁ a substring of s₂
 - Evaluates to a bool
- **Examples:**
 - s = 'abracadabra'
 - 'a' in s == True
 - 'cad' in s == True
 - 'foo' in s == False

- Function len: len(s)
 - Value is # of chars in s
 - Evaluates to an int
- Examples:
- s = 'abracadabra'
 - len(s) == 11
 - len(s[1:5]) == 4
 - s[1:len(s)-1] == 'bracadabr'

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

- def middle(text):
 - """Returns: middle 3^{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
- The call frame is **EMPTY**

Not All Functions Need a Return

def greet(n):

"""Prints a greeting to the name n

Parameter n: name to greet Precondition: n is a string"""

print('Hello '+n+'!')

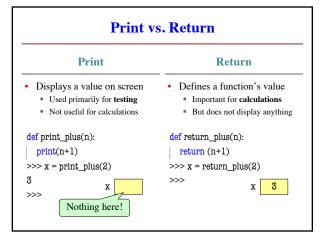
print('How are you?')

No assignments or return

Displays these

strings on the screen

Procedures vs. Fruitful Functions Procedures Fruitful Functions Fruitful Functions Functions that do something Call them as a statement Example: greet('Walker') 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



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'

