CompSci 101
Conditionals (Cont’d), Collections, Strings, Lists
Announcements

• Upcoming due dates
  • All Sakai quizzes due @145pm day of lecture
  • Assignment 1-due 2/11 @ 1130pm

• Piazza channel
  • Direct questions here
E is for ... 

• Escape Sequence  
  • Why \n is newline and \t is a tab  

• Encryption  
  • From Caesar Ciphers to SSL and beyond  

• Enumerate  
  • Adding counters to iterable  

• Emoticon  
  • 😊 😞
Computer Scientists to Know

• Dr. Clarence “Skip” Ellis  
  • 1st Black person to earn a Ph.D. in computer science

• Dr. Timnit Gebru  
  • Co-founder, Black in AI  
  • Ethical AI researcher
PFTD

• Selections/Conditionals continued
• Strings
• Lists

• “The mere imparting of information is not education.”
  • Dr. Carter G. Woodson
Selection/Conditionals: if...elif...else

```python
if BOOLEAN_CONDITION:
    CODE_BLOCK_A
elif BOOLEAN_CONDITION:
    CODE_BLOCK_B
else:
    CODE_BLOCK_C
```

```python
if __name__ == '__main__':
    num1 = 5
    if num1 == 5:
        print("The number is 5!")
    else:
        if num1 < 5:
            print("The number is less than 5!")
        else:
            print("The number is greater than 5!")
```
WOTO 4: Review

• Extending your program from WOTO 3
• Simulate rolling two dice
  • “Roll” two dice
  • Results of the rolls sent to function named \textit{sum}
    • Sum dice values
    • If sum is 7 or 11, then output “You win!”
    • Otherwise, output “Next time!”
Functions Calling Other Functions

def function1(parameter):
    ...
    result=function2(parameter2)
    return result

def function2(parameter2):
    ...
    return result2

if __name__ == '__main__':
    output=function1(argument)
    print(output)

Example code(PyCharm)
Collection Data Type

- Collection of books, toys, shoes
  - Direct access to each item
- Comprised of smaller pieces
  - Strings and lists
- Strings
  - Smaller strings of size one char
  - Empty string- "" or '
- Operations on strings
  - + → concatenation
  - * → repetition
Indexing a String

`string_name[index]`

- Index: 0 to (string_length-1)

**Whitespaces count**
Slicing Strings

• Slicing bread, tomatoes, etc.
• Substring (smaller part) of the larger string

```
string_name[n:m]
```
Comparing Strings

• Compares strings to determine the relationship between them
  • ==, >, <, >=, <=, !=

• **string1 == string2**

**need to output this or store the result**
in and not in operators

• Is string1 a substring of string2?

`string1 in string2`

string can be a variable or a string literal (e.g., “This is literally an example of a string literal.”)
Lists

• Groceries, errands, names, etc.
• Lists are:
  • Ordered
  • Mutable
  • Duplicate elements allowed
  • Elements don’t have to be the same type

```python
list_name = [item1, item2, ...item6]
```

**only top-level items in list**
List access and length

• Similar to strings
  \texttt{list\_name[index]}
• list\_name-your variable name
• index-character element directly accessing
  • leftmost 0 to list\_length-1

• What about \texttt{list\_name[-1]}?
Slicing Lists

• Sublist (smaller part) of the larger list

\[ \text{list} \_\text{name}[n:m] \]

\( n \)-index of the first character in the sublist

\( m \)-index of the character that immediately follows the last character in the sublist
in and not in operators

• Is list1 a member of list2?

```python
list1 in list2
list1 not in list2
```
Assignment 1: Totem Poles
Learning Goals: Totem Pole

• Understand differences and similarities:
  • Function definitions vs function calls
  • Functions with return statements vs those without
  • Functions with parameters vs those without
  • Functions can be arguments

• Be creative and learn lesson(s) about software design and engineering
  • Create a small, working program, make incremental improvements.
  • Read the directions and understand specifications!
<table>
<thead>
<tr>
<th>Function</th>
<th>Parameters</th>
<th>Returns</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>part_DESCRIPTION</td>
<td>No parameters</td>
<td>A string</td>
<td>part_smiling_mouth</td>
</tr>
<tr>
<td>DESCRIPTION_head</td>
<td>No parameters</td>
<td>No return value, only prints</td>
<td>happy_head</td>
</tr>
<tr>
<td>head_with_DESCRIPTION</td>
<td>1 or 2 parameters of type function</td>
<td>No return value, only prints</td>
<td>head_with_mouth</td>
</tr>
<tr>
<td>totem_DESCRIPTION</td>
<td>No parameters</td>
<td>No return value, calls head functions</td>
<td>totem_fixed, totem_selfie, totem_random</td>
</tr>
</tbody>
</table>

selfie_band, head_random – helper functions!
Creating your program

Start small and build incrementally...
With functions grow by...

- Minimal code that does run and can be submitted
- Where go from here?
  - Add head part functions to create happy_head()
  - Create the next head function for totem_fixed and any new head part functions
  - Try a head_with function
  - Go to the next totem
  - etc.
Totem Assignment by Tuesday

• At minimum...
• Read the assignment
• Do the Totem reading quiz
• Create initial design
• Create project and start writing code (do not need to finish)

• Goal: Find your first question about how to do this assignment then ask on Piazza or at consulting/office hours
Remember

• Work smarter, not harder
• Design first
• Try to identify where you are stuck
  • Identify resources to help solve problem
• Leverage your design and PythonTutor to understand program flow of control
  • http://pythontutor.com