

CompSci 101

range, Accumulators, Traversing strings/lists

1

Announcements

- No lab this week
- Upcoming due dates
 - APT-2 (Tuesday)
- Office hours and consulting hours
- PythonTutor
- Piazza channel

2

H is for ...

- **HTTP**
 - A Protocol we use every day, and HTTPS
- **Hello World**
 - The quintessential first program: 40 years ago!
- **Hack**
 - Hacker, Hacktivism, Hack Duke
- **Hashing**
 - How Dictionaries work



3

Computer Scientists to Know

- **Dr. Marc Hannah**
 - Co-Founder, Silicon Graphics, Inc. (SGI)
- **Tiffani Ashley Bell**
 - Founder and Executive Director, The Human Utility



4

PFTD

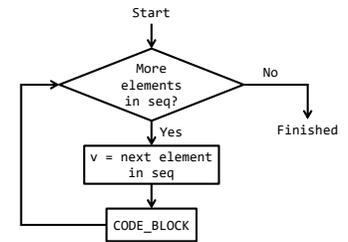
- True
- Udcjh#xqfwrq
- Dffxp xdlwrq

5

Anatomy of a for loop

```
for v in seq:
    CODE_BLOCK
```

```
if __name__ == '__main__':
    for number in [0, 1, 2, 3]:
        print(number)
```



6

range() function

```
if __name__ == '__main__':
    for number in [0, 1, 2, 3]:
        print(number)
```

- What about larger numbers?

- **range(stop)**
 - 0 up to (not including) stop
- **range(start, stop)**
 - Specify start value (increment by 1)
- **range(start, stop, step)**
 - Specify step value

7

WOTO 1: <http://bit.ly/101s21-0218-1>

8

Why use loops?

- Repetition
 - Keeping a running total (counter)
 - Summing (other repetitive calculations)
- Accumulators
 - "Accumulate"-*acquire an increasing number of quantity of.*
- Rules for accumulators
 - Initialize the "running total"
 - Don't initialize inside the loop
 - Increase the total with each iteration

9

Another way to use accumulators

```
def square(x):
    "raise x to the second power"
    runningtotal = 0
    for counter in range(x):
        runningtotal = runningtotal + x
    return runningtotal
```

```
def square(x):
    "raise x to the second power"
    runningtotal = 0
    for counter in range(x):
        runningtotal += x
    return runningtotal
```

10

WOTO 2: <http://bit.ly/101s21-0218-2>

11

Traversing strings

```
if __name__ == '__main__':
    name = "Tiana"
    for i in range(5):
        print(name[i])
```

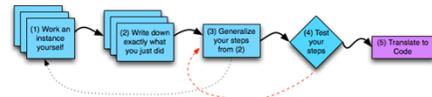
12

Accumulators with Strings

- How is "+" used with strings?
 - Concatenation
 - result = "string1" + "string2"
- Still require initialization
 - Empty string ("") instead of 0
- Still "acquiring/increasing quantity."

13

Designing Solution



1. Work an instance: "Duke" -> "DK"
2. What did we do?
 - a. Paper and pencil, write it down!
3. Generalize
4. Test: "Computer" -> "Cmpttr"?

14

WOTO 3: <http://bit.ly/101s21-0218-3>

- Write a function `isVowel(ch)`
 - Returns true if the input is a vowel and false otherwise
 - Input: string of a single character (length 1)
 - Output: bool

15

Which is better to traverse list?

```

fruits = ["apple", "orange", "banana", "cherry"]   fruits = ["apple", "orange", "banana", "cherry"]

for position in range(len(fruits)): # by index   for afruit in fruits: # by item
    print(fruits[position])                print(afruit)
  
```

16

WOTO 4: <http://bit.ly/101s21-0218-4>

- Given lst (which contains a list of ages), write code that traverses the list to count the number of ages that can purchase alcohol
 - Must be 21+

Example:

```
lst=[24, 18, 32, 12, 19, 55, 10, 20]
```

17

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>

18