Conditionals, Lists, Loops (13 minutes)

Session 2

Purpose of Today's Session

This session is focused on practicing the fundamentals of loops and conditionals using the list data structure.

HCs

Readings

Study Guide

Review the following sections from learnpython.org:

- **Lists**
- **Conditions**
- **Loops**

While reviewing, be sure to do the exercises - you cannot learn to code by just reading!

Special note: We want you to be extra prepared this week in case the technical difficulties of last week happen again. If you have not done so already, please download the Anaconda distribution of Python. One possible link is https://www.anaconda.com/download/. Be prepared to run code using a Jupyter notebook. Furthermore, it is strongly recommended that you work on the exercises given last week if you did not finish them. Please see your tutors for a reminder of the problems.

Pre-Class Work

[✓ **Individual**]
Group

Background Readings for Faculty

Classroom Activities

Conditionals, Lists, Loops 13m

NOTES FOR FACULTY

This is a problem solving session devoted to string and arithmetic exercises. Note that exercise 0 is intended for students who have not completed the beginner problems from last week. It is essential that students have some exposure to defining and calling functions. Should there be difficulties implementing the Cocalc notebooks, please use the document provided in the last step to share the problems with students.

Activity Introduction LAYOUT: 1-UP 2m

Remind students of the format of the structured study sessions: students work on Python exercises in small breakout groups with students of similar proficiency, calling on the peer tutors for help as needed. You should be prepared to move students around in the groups as appropriate. Students may work on problems appropriate to their level, but should continue working through the entire session. Students should also periodically check in with you to report on their progress.

Activity Learning Goal Slide

Slide

Activity Learning Goal

Write programs with loops and conditionals using the list data structure.

Breakouts LAYOUT: 1-UP 10m

Breakout

Breakouts

Groups

8 groups

unlimited

Breakout Notes

BREAKOUT NOTES CONTENT

Exercise 0: Beginner - Do these if you did not get to them last week.

1. **Define a function called print_three() that takes a string as an input and then prints the string three times. Below the function definition, use input() to request a name, then call print_three, using the name as the input to the function.**
2. **Define a function named my_computation() that takes three integers m, n, p, as input, and outputs (returns) the quantity p*(m-n)^5. Write a script that asks for 3 integers as input, passes these integers to my_computation(), then prints the output of my_computation.**

Exercise 1: Beginner

1. **Create a list with the names of each member of your group. (Make sure these are strings!) Print the first name in the list you made by accessing that name from the list then printing.**
2. **Append the name of one of the tutors to your list and then print the entire list.**
3. **Append the name of someone else from the class to your list and then print the entire list.**
4. **Create a for loop over your list that prints**

4. Create a for loop over your list that prints each name on a new line. (Advanced beginners: print each name in the reverse order.)

5. Create a small script that has the user input a name, then checks each entry of the list you have been using to see whether the name is a match. Print “true” or “false” for each entry of the list.

Exercise 2: Beginner

1. Create a list with twenty numbers in it. Write a program that asks users to input two integers, then produces the following outputs: (i) count of the number of entries in the list that are either less in value than the smaller of the two inputs or greater than the value of the larger of the two inputs. (ii) count of the number of entries in the list that are in value strictly between, but not equal to the values of the two inputs.

2. Write a program that asks for an integer as input and outputs the number of times that integer appears in the list from the last part. Do this in two ways:

   (i) Search the Python documentation online for a list method (a pre-made command that works for lists, like `.append()`) that does this and run it here.

   (ii) Define a function *my_counter() *that takes both a list and an integer as input, then returns the number of times the integer occurs in the list, using a loop to compute this. Call *my_counter() *on the above list and a number that you input.

Exercise 3: Advanced Beginner - Project Euler #2

Each new term in the Fibonacci sequence is generated by adding the previous two terms. By starting with 1 and 2, the first 10 terms will be: 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, ... By be: 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, ...By considering the terms in the Fibonacci sequence whose values do not exceed four million, find the sum of the even-valued terms. (Intermediate option: use recursive function calls to solve this problem.)

Exercise 4: Intermediate - Project Euler #4

A palindromic number reads the same both ways. The largest palindrome made from the product of two 2-digit numbers is 9009 = 91 × 99. Find the largest palindrome made from the product of two 3-digit numbers.

[✓ Same notes duplicated for all groups]
Different notes for each group

ALL GROUPS

Back-Up Document LAYOUT: 1-UP 1m

This step contains a Google document with the list of programming exercises. Should there be problems with Cocalc, you can share this version of the worksheet with students.

Resource

Document

Exercises

DOC CONTENT

After teaching all sections of this class...

please feel free to submit your feedback on this LP