ITI 1120

Lab # 2

numbers, expressions, assignment statements, functions … bit of strings

Objectives

Getting familiar with Python's expressions, function calls, assignment statements, function design via:

- two exercise sets (one on paper, on one the web)
- four programming exercises

Starting Lab 2

- Open a browser and log into Blackboard Learn
- On the left hand side under Labs tab, find lab2 material contained in lab2-students.zip file
- Download that file to the Desktop and unzip it.

Task 1

- Open the pdf file called in lab2-exercises.pdf
- Read the instructions and complete all the exercises

Note: If you have not printed this document beforehand or do not have a tablet with a pen, just take a piece of blank paper and write your answers on that paper.

Task 2

- Open a browser and go to https://class.coursera.org/programming1-002
- If you have not opened an coursera account open one, under your own name, and remember your account for the rest of this semester.
- Log in and click on "go to course" if you are not already in the course area
- On the left hand side click on Exercises
- A page entitled "Homeworks" will open.
- Click on Week 1
- Click on Attempt Homework
- Answer all 15 questions and click Submit Answer to see how well you did. The coursera will automatically grade you. It will give you some understanding of how well you understand the material. You can pick a random answer for Question 2 since we did not cover that material yet.

Strings

In addition to integer, float (i.e. number) and boolean objects. Python has string objects. (For now think of objects as just values)

- A string is a sequence of characters between single quotes, double quotes or triple quotes.

  'This is a string'

  Note that these are also strings:

  " " this is a string that is comprised of one blank space

  '257' this is a string unlike 257 which is an integer

- Strings can be assigned to variables. Examples:

  s1='Harry'

  s2="Potter"

- There are many operations that can be applied on strings. For example when the + operator is applied to two strings, the result is a string that is the concatination of the two. For example, s1+s2, would result in a string 'HarryPotter'

  Note that "The year is "+ 2525 would cause a syntax error since the + operator can be applied to two numbers or two strings but not the mix of the two. This however is a valid expression 'The is year "+ "2025"

  Python also has * operator for strings. It can be applied to a string and an integer. Eg: 4 * "la" gives 'lalalala'

Programming Exercises

Pretend that the following programming questions are your Assignment 1. And solve the following questions as if they were assignment 1. In other words, write all the following questions in one file called lab2_prog_solutions.py

IMPORTANTE NOTE: for this LAB and the ASSIGNMENT(s):

If a question specifies the function name and the names of its parameters, then that same function name and function parameter names must be used when defining the functions. That is the case in every question in your assignment 1. For example in the question on the next page, your function definition MUST start with:

```
def repeater(s1, s2, n):
```

as that is specified as a part of the question

Question 1:

Write a function called repeater(s1, s2, n) that given two strings s1 and s2 and an integer n **returns** a string that starts with an underscore, then s1 and s2 alternate n times, then ends with an underscore. (For those who know loops: you may not use loops to solve this questions.)

Testing your code:

Here is what the output of your function should look like when you make the following function calls:

```
>>>
>>> repeater("AAA", "x", 3)
'_AAAxAAAxAAAx_'
>>> repeater("+", "--", 30)
'_+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--_'
>>> repeater("T", "|_|", 100)
'_T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|
T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|
_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|
T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|T|_|_'
>>>
```

Question 2:

Read the first paragraph of this page on quadratic equation and finding its roots (it. solutions)

https://en.wikipedia.org/wiki/Quadratic_equation

Write a function called roots(a, b, c) that given three coefficients a and b and c **prints** a nicely formatted message displaying the equation and its two roots (the two roots maybe be the same number). You may assume that a is a non zero number, and that a and b and c are such that b^2-4ac is a positive number. (Do you know why we are making this assumption?)

```
>>>
>>> roots(-1, 4, 1.5)
The quadratic equation with coefficients a = -1 b = 4 c = 1.5
has the following solutions (i.e. roots):
-0.345207879911714 87  and 4.345207879911715
>>> roots(1, 2, 1)
The quadratic equation with coefficients a = 1 b = 2 c = 1
has the following solutions (i.e. roots):
-1.0  and -1.0
```

Question 3:

Think back on the previous question …

Write a function called real_roots(a, b, c) that **returns** True if the quadratic equation with the given three coefficients a and b and c has real roots. Otherwise it returns False.

Recall that roots of a quadratic equation are real if and only if b^2-4ac is a non-negative number.

Testing your code:

```
>>>
>>> real_roots(-1, 4, 1.5)
True
>>> real_roots(1, 2, 1)
True
>>> real_roots(1, 1, 1)
False
>>>
```

Question 4:

Write a function called reverse(x) that given a two digit positive integer x **returns** the number with reversed digits. (You may assume that x is a two digit positive integer)

Hints: Think of mod and div operators and how they can help. What number should you div x with to get the 1st digit.

Testing your code:

```
>>>
>>> reverse(27)
72
>>> reverse(44)
44
>>> reverse(19)
91
>>>
```