15-110: Principles of Computing, Spring 2018

Lab 3 – Thursday, February 1

Goals

- Practice creating Python functions, experimenting more with modulo
- Experiment with multiple uses of the if statement
- Create a loop to display a pattern of numbers

Part 1 : Print vs. Return [TA Demonstration]

```
def f1(x):
    return x*x

def f2(x):
    print(x*x)

def test():
    y = f1(15)
    print("y is: ", y)
    z = f2(110)
    print("z is: ", z)
```

```
python3 -i printing.py
>>> test()
y is:  225
12100
z is:  None
```

Part 2: If and While Statements [TA Demonstrations]

The `if` and `while` statements depend on Boolean conditions, expressions that evaluate to `True` or `False`.

General structure of the if statement:

```
if condition:
    statement(s)
```

General structure of the while statement:

```
initialize_loop_variable
while condition:
    loop_body
    modify_loop_variable
```

Part 3: Student Activities

3.1 Doomsday Algorithm

In the Gregorian calendar (the one that we use today), an amazing property exists. On any given year, the dates April 4th (4/4), June 6th (6/6), August 8th (8/8), October 10th (10/10), and December 12th (12/12) all occur on the same day of the week. Also, so do May 9th (5/9) and September 5th (9/5) as well as July 11th (7/11) and November 7th (11/7). These days are known as "doomsdays".

To compute the day of the week for the doomsdays given a year, follow the algorithm below (good for the years 2000 – 2099):

i. Let `y` be the last two digits of the year. (How do you compute this using the modulo operator?)

ii. Let `a` be the integer quotient when you divide `y` by 12.

iii. Let `b` be the integer remainder when you divide `y` by 12.

iv. Take the integer remainder from the previous step and divide it by 4, keeping just the integer quotient as `c`.

v. Let `d` be the sum of `a`, `b`, and `c`.

vi. Let `e`, which represents the day of the week for the doomsdays, be the integer remainder after adding 2 to `d` and dividing the result by 7. (Note that `e` should be a value between 0 and 6. Why?)

vii. Return the string shown below given the final value of `e`:

e	String to Return
0	"Sunday"
1	"Monday"
2	"Tuesday"
3	"Wednesday"
4	"Thursday"
5	"Friday"
6	"Saturday"

a. Trace the algorithm above on paper for 2017. What day do you get? Check a calendar online to verify that you got the correct answer.

b. Create a file `doomsday.py` using gedit. In `doomsday.py`, define a Python function `compute_day(year)` that **returns** the day of the week for the doomsdays given the value supplied for the parameter `year` using the algorithm above. Using Python3, test your function with at least 5 different years to see if it is working as you intended. (Is this enough testing in your opinion?)

3.2 Program Logic

Recall that a variable can store different kinds of data. A variable can hold a Boolean (logical) value of True or False.

If we connect two relational expressions (that evaluate to Boolean values) with the operator and, the result is True if both relational expressions are true. Otherwise, the result is False. If we connect two relational expressions with the operator or, the result is True if at least one of the relational expressions is true. Otherwise, the result is False.

a. Create a file legal.py in your lab3 folder and cut and paste the code below from Autolab. This function prints out what is legal to do for a person in the U.S. given the person's age, gender and citizenship based on a simplification of the U.S. law.

```
def legal_actions(age, male, citizen_of_USA):
    print("Legal to:")
    if age >= 21:
        print("Drink alcohol.")
    if age >= 18 and citizen_of_USA == True:
        print("Vote.")
        if male == True:
            print("Get drafted into the armed forces.")
```

Save and then load the function above into python3. When you make the function call below, you are determining what a 25-year-old male citizen of the USA can do legally of the three options:

```
legal_actions(25, True, True)
```

For the function call above, you should get all three actions printed out.

b. Create a file table.txt using gedit. In table.txt, write your answers (age, male, and citizenship that will result in the given output) that correspond to the blank spaces in the table shown below. Use your function in Python3 to test your answers. If a particular parameter does not matter for a specific output, indicate your answer as "anything". If a particular output is impossible, indicate "None" in each of the age, male, and citizenship columns.

Output	Age	Male?	US Citizen?
DRINK, VOTE & DRAFTED	25	True	True
DRINK & VOTE only			
DRINK & DRAFTED only			
VOTE & DRAFTED only			
DRINK only			
VOTE only			
DRAFTED only			
No output			

EXTRA: Think about how your results will change if the operator and in the function above were replaced with or.

3.3 Nested Loops: A Triangular Puzzle

Consider the following output shown below:

```
1
2 2
3 3 3
4 4 4 4
5 5 5 5 5
6 6 6 6 6 6
7 7 7 7 7 7 7
8 8 8 8 8 8 8 8
9 9 9 9 9 9 9 9 9
```

Create a file triangular.py using gedit. In triangular.py, write the following function and fill in the missing pieces so you get the output above when you call the function in the interpreter.

```
def triangle():

    for row in range(______,______):

        for i in range(_______):

            print(______, end=________)

        print()  # move cursor to next line
```

HINTS:

1. The outer loop variable row controls which row you are printing out. Note that the row number corresponds to the numbers being printed out in that row.

2. For each row, the inner loop variable i controls how many numbers are printed in that row.

HARDER: Try to print out a triangle with 15 rows and get the numbers to line up in neat columns.

Submission

When you finish the lab, you should be inside the lab3 folder, which is inside the private/15110 directory. When you type ‘ls’ and press the Enter key, you should see the following files: **doomsday.txt**, **legal.py**, **table.txt**, and **triangular.py**. Once you see all files, please type ‘cd ..’ and press the Enter key. Then, zip your lab3 folder by typing ‘zip –r lab3.zip lab3’. Please submit the zipped file lab3.zip on Autolab.