National University of Singapore

Faculty of Law (2024-2025)

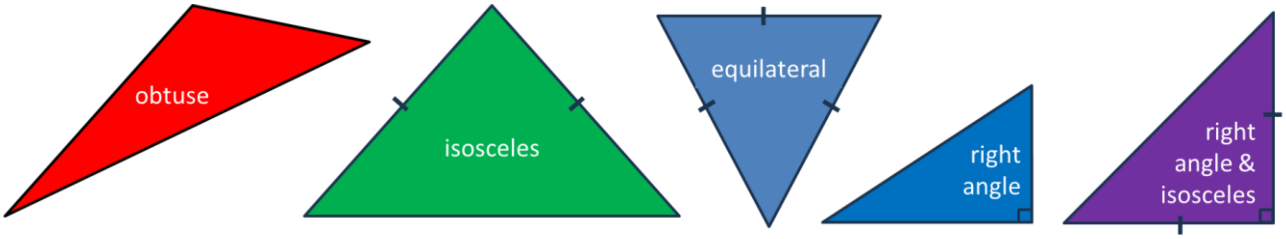
LC2017: Law and Technology

Tutorial (Week 4): Python – Conditionals

This tutorial is an in-class Python coding exercise. Your participation in the tutorial will make up your class performance grades comprising 15% of your overall grades for this course.

Write Python 3.12.x+ code for all of the following, except for questions marked with an *, which are optional. Note that the square brackets [*with italicized phrases*] are placeholders, and that → represents an indentation.

1. You are presented with this chart of triangles, and are asked to identify the triangle, based on inputs of the following two variables: `angle_1` and `angle_2` (two angles of a triangle). Because the angles in a triangle add up to 180 degrees, it is not necessary to define the third angle.



(a) Set the variables `angle_1` and `angle_2` with `input` statements to hold as integers the two angles of a triangle.

(b) Using these two variables – `angle_1` and `angle_2` – formulate a conditional statement with `if` that identifies the triangle as an equilateral triangle and prints "equilateral". Your code should be in the following format that completes the expressions expression using angle_1 and expression using angle_2:

```
if [expression using angle_1] and [expression using angle_2]:
→  print("equilateral")
```

(c) As (a) above, repeat the process by formulating new conditional statements with `if` that each identifies the triangle as an obtuse, isosceles, right angle and right angle & isosceles triangle.

(d) Chain the if conditional statements from (a) and (b) with `if/elif/else` so that your conditional statements can correctly identify one of 5 different types of triangles. (If the triangle cannot be classified, do nothing.)

*(e) Identify *one* triangle classification that is missing from the classification above. (This is an example of a logic issue not related to coding.)

Hints:

- ⓘ The order in which the `if/elif/else` statements are chained together matters. Think through the logic for ordering the `if` statements.
- ⓘ Test your statements with the following combinations (with their corresponding classifications):

angle_1	angle_2	Classification
60	60	equilateral
45	90	right angle & isosceles
90	30	right angle
75	75	isosceles
15	160	obtuse
45	45	†right angle & isosceles
30	60	†right angle
75	30	†isosceles
20	30	†obtuse

- ⓘ Classifications marked with † require you to extend your conditional statements and your logic. Think carefully about how you could do it.

2. Based on your answer to Q3 in Tutorial Week 3 (question on s 376E(1), Penal Code), write a script that does the following:

   (a) Prompts for inputs for the following variables, converts the (corresponding) integer and inputs ("True" for True and "False" for False) to integer and Boolean values (True/False) respectively and saves the inputs into the corresponding variables:

   - `A_age` (integer)
   - `B_age` (integer)
   - `has_met` ("having met" - Boolean)
   - `has_communicated` ("having communicated" - Boolean)
   - `prev_occasion` ("previous occasion" - integer)
   - `A_intentionally_meets_B` (Boolean)
   - `A_travels_with_intent_to_meet_B` (Boolean)
   - `B_travels_to_meet_A` (Boolean)
   - `A_initiated_meeting` (Boolean)
   - `A_agreed_to_meet_B` (Boolean)

   (b) Sets the following variables:

- `A_intends_to_do_anything_to_B` (Boolean - True)
- `involve_A_committing_relevant_offence` (Boolean - True)
- `A_reasonable_belief_B_is_16_or_more` (Boolean - False)

(c) Evaluates the rule in s 376E(1) as a Boolean expression and prints “s 376E(1) offence” or “no s 376E(1) offence” accordingly.

Hints:

ⓘ To convert your input from "True"/"False" to True or False, you may use a Boolean expression that evaluates if the input is equal to "True". You may also use an if conditional statement e.g.:

```
if [input for has_met] == "True":
→  has_met = True
else:
→  has_met = False
```

ⓘ Your code should look like the following:

```
A_age = [input statement + conversion]
B_age = [input statement + conversion]
#[other input statements + conversions]
A_intends_to_do_anything_to_B = True
involve_A_committing_relevant_offence = True
A_reasonable_belief_B_is_16_or_more = False
if [Boolean expression of the rule in s 376E(1)]:
→  print("s 376E(1) offence")
else:
→  print("no s 376E(1) offence")
```

3. Eddie is trying to write a Python program that accepts an integer, *n*, and prints out all numbers between 1 and *n* (including *n*), unless the number is divisible by 3, 5 or 15.

   - If a number is divisible by 3, the program prints “Tick” instead of the number.
   - If a number is divisible by 5, “Tock” is printed.
   - If a number is divisible by 15, “TickTock” is printed.

   However, Eddie's code has bugs.

   You are tasked with the following:

   (a) Modify Eddie's code such that it prints the expected output.

   (b) Explain what was wrong with the original code.

   This is Eddie's code that you are expected to modify:

```
1   n = input()
2   for i in range(1, n):
3       if i % 3 == 0:
4           print("Tick")
5       if i % 5 == 0:
6           print("Tock")
7       if i % 3 == 0 and i % 5 == 0:
8           print("TickTock")
9       else:
10          print(i)
```

This is the output and expected output, compared side-by-side:

Output (where *n* = 15) after correcting one mistake:	Expected output (where *n* = 15):
1 2 Tick 3 4 Tock 5 Tick 6 7 8 Tick 9 Tock 10 11 Tick 12 13 14	1 2 Tick 4 Tock Tick 7 8 Tick Tock 11 Tick 13 14 TickTock

Hints:

ⓘ The first mistake that you should correct is found in line #1 and is a common mistake made by Python programmers who often forget the difference between numbers and strings.

ⓘ The `%` or modulus operator calculates the remainder of a division. Thus 11 % 4 gives 3 (i.e. $4 \times 2 = 8$ and therefore $11 \div 4 = 2$ remainder 3).

ⓘ If `a % b == 0`, this means that a is divisible by b since there is no remainder for $a \div b$.

ⓘ The `for` statement and `range` function creates a repeating sequence (a loop). See Lecture 8. In line #2, `for i in range(1, n):` creates a repeating sequence where `i` is assigned the values 1, 2, … and so on until `i` is `n-1`.

LC2017: Law and Technology

Tutorial (Week 4): Python – Conditionals

You are strongly encouraged to work on these exercises ahead of tutorials. And you may discuss techniques and approaches, including exchange hints, with each other. The temptation to get another student's answer or search for answers online or use AI-powered tools is there, but do remember that you will learn best by thinking and working through the exercises yourself.

Have fun coding!

A/P Daniel Seng

July 28, 2024