CS-UY 1114: Lab 3

# `if` Statements

You must get checked out by your lab CA **prior to leaving early.** If you leave without being checked out, you will receive 0 credits for the lab.

### Restrictions

The Python structures that you use in this lab should be restricted to those you have learned in lecture so far. Please check with your teaching assistants in case you are unsure whether something is or is not allowed!

***Create a new python file for each of the following problems.***

**Your files should be named** *lab[num]_q[num].py* **similar to homework naming conventions.**

## Problem 1: *What Does It Print?*

*Solve this problem by hand.*

(a) What would be printed if x = 7?

```
x = 7
if 10 > x >= 5:
  x -= 4
if 10 > x >= 5:
  print(x)
else:
  print(x + 10)
```

(b) What would be printed if x=7?

```
x = 7
if 10 > x >= 5:
    x -= 4
    print(x)
elif 10 > x >= 5:
    print(x)
else:
    print(x + 10)
```

(c) What would be printed if y = 15?

```
y = 15
if y > 7:
```

```
    y += 1
elif y > 10:
    y += 1
elif y > 15:
    y += 1
print(y)
```

(d) What would be printed if y=15?

```
y = 15
if y > 7:
    y += 1
if y > 10:
    y += 1
if y > 15:
    y += 1
print(y)
```

(e) What would be printed if y=15?

```
x = 0
y = 15
if y % 2 == 0:
    if y > 5:
         x = y ** 2
    else:
        x += 1
else:
    if y > 5:
        y = y ** 2
    else:
        y += 1
print(x)
```

## Problem 2: *Parking Costs*

Create a program that will calculate the total price of parking based off of the rates below:


*"Up to" in this diagram will mean inclusive and "out by" will mean exclusive*

Your program must:

- Prompt the user which garage they parked in (`"North"` or `"South"`)
- Prompt the user what hour they entered the parking garage (in military time)
- Prompt the user what minute they entered the parking garage
- Prompt the user what hour they left the parking garage (in military time)
- Prompt the user what minute they left the parking garage
- Output to the user their total cost of parking

You may assume the parking garage follows a no overnight parking policy where cars must leave on the same day they entered. You may also assume cars will not exceed parking for 12 hours.

Below is a sample execution of the program:

```
Which garage did you park in? North
Please enter the hour, followed by minutes, you entered the garage (military
time):
9
45
Please enter the hour, followed by minutes, you left the garage (military time):
10
15
Your total price for parking will be $14.00
```

```
Which garage did you park in? South
Please enter the hour, followed by minutes, you entered the garage (military
time):
22
50
Please enter the hour, followed by minutes, you left the garage (military time):
23
0
Your total price for parking will be $0.00
```

*Hints*

- Arithmetic with time can be tricky. You may find it useful to first convert everything to minutes. So, the time (hour and minutes) the car entered into total minutes, then the time (hour and minutes) the car left into total minutes. You can then calculate the total minutes a car spent in the garage by subtracting these two values

## Problem 3: *What Is My Grade?*

In this problem you will be writing some code to calculate a final grade for this course.

1. Get the highest exam grade, second highest exam grade, third highest exam grade, homework average, and lab average values from the user.
2. Calculate the weighted average of the grades using following weights.

```
Highest exam (25%)
Second highest exam (20%)
Third highest exam (15%)
Homework (20%)
Lab (20%)
```

3. Determine the letter grade that correlates to this weighted average based on the following.

| Average | Letter Grade |
|---|---|
| > 92 | A |
| 90 - 92 | A- |
| 87 - 89 | B+ |
| 83 - 86 | B |
| 80 - 82 | B- |
| 77 - 79 | C+ |
| 73 - 76 | C |
| 70 - 72 | C- |
| 67 - 69 | D+ |
| 60 - 66 | D |
| < 60 | F |

Your code should produce the following:

```
Enter your highest exam grade: 92
Enter your second highest exam grade: 86
Enter your third highest exam grade: 73
Enter your average homework grade: 89
Enter your average lab grade: 95
Your final grade is B+
```

Feel free to use this program to calculate your grade at the end of the semester!

## Problem 4: *Single Use Calculator*

This problem will focus on using conditionals to create a single use four operation calculator.

You should take in user input for the first number, second number, and the operation. You should handle addition, subtraction, multiplication, division, and division by 0.

For all successful operations provide the following message:

```
first_number operation second_number = answer
```

For all invalid operations provide the following message:

```
first_number operation second_number is an invalid operation.
```

Here are a couple of examples of the expected output.

```
Enter your first number: 2.3
Enter the operation (+, -, *, /): *
Enter your second number: 4.1
2.3 * 4.1 = 9.429999999999998
```

```
Enter your first number: 4
Enter the operation (+, -, *, /): /
Enter your second number: 0
4.0 / 0.0 is an invalid operation.
```

## Problem 5: *It Pays to be Lucky*

Try your luck in pulling one single Pokémon card from the new trading card pack! Pokémon trading cards each have a picture of a Pokémon along with its respective stats and attacks. Each card can come in different design variations in different rarities that can increase its resell value. Let's say the prices increase according to 3 categories, each with subcategories:

- **Rarity**: *Uncommon* cards receive a $5 price increase, and *rare* cards receive a $15 price increase.
- **Holographicness**: *Reverse holos* receive a $10 price increase, *holos* receive a $15 price increase, and *full holos* receive a $20 price increase.
- **Special Pokémon**: *Starters* receive a $5 price increase and *legendaries* receive a $30 price increase.

All Pokémon cards start at a $5 value.

Create a program that asks the user yes or no questions about their card. Your program should:

- Prompt **at least** 3 inputs about the card's categories: if the card is of a special rarity, if the card is holographic, and if the card is a special Pokémon.
- For every category the user answers yes to, prompt the user with yes or no questions about the subcategories for that category
- Output the card's final resale price

The following are examples of possible outputs:

```
Welcome to the Pokémon card price calculator! Is your card of a special rarity?
(Y/N) N
Is your card holographic? (Y/N) N
Is your card of a special Pokémon? (Y/N) N
Your card has a final resale price of: $5
```

```
Welcome to the Pokémon card price calculator! Is your card of a special rarity?
(Y/N) Y
Is your card uncommon? (Y/N) Y
Is your card holographic? (Y/N) Y
Is your card a reverse holo? (Y/N) N
Is your card a holo? (Y/N) Y
Is your card of a special Pokémon? (Y/N) Y
Is your card of a starter? (Y/N) N
Is your card of a legendary? (Y/N) Y
Your card has a final resale price of: $55
```

Use this final output as a guideline to all the possible questions your program *could* ask the user:

```
Welcome to the Pokémon card price calculator! Is your card of a special rarity?
(Y/N) Y
Is your card uncommon? (Y/N) N
Is your card rare? (Y/N) N
Is your card holographic? (Y/N) Y
Is your card a reverse holo? (Y/N) N
Is your card a holo? (Y/N) N
Is your card a full holo? (Y/N) N
Is your card of a special Pokémon? (Y/N) Y
Is your card of a starter? (Y/N) N
```

```
Is your card of a legendary? (Y/N) N
Your card has a final resale price of: $5
```

## Problem 6: *Python and Friends*

If you are reading this, you are probably really familiar with python now. A lot of people hate them, they can be difficult to work with, they can get very long, and they are very dangerous, sometimes even able to kill humans. They are some of the world's largest snakes after all. Wait, what were you thinking about when I mentioned python?

Anyways, there are many species of snakes in the world, some are dangerous and others are relatively safe. Therefore, it is important to classify the types of snakes. This is where you come in. Write a program that asks for information regarding a snake species, classifies what kind of snake it is, and prints out the snake along with fun facts about it.

The information you would need to ask from the user are whether the snake is venomous, whether the snake is small, and whether the snake is aggressive. You can assume the user will only answer `"yes"` or `"no"` for each question. A sample execution for collecting input may look like this:

```
Is the snake venomous? no
Is the snake small? yes
Is the snake aggressive? no
```

Next, identify what the type of snake is from the inputs. You are **NOT** allowed to use `and` and `or` in your solutions. Instead, you should be using nested `if` statements (`if` statements inside another `if` statement). The possible types of snakes are:

Python Python

- Fun Facts: One of the largest and most famous snakes. However, they are pretty **slow**, and are commonly used to **introduce** people to learning about snakes.
- Characteristics:
    - Non-Venomous
    - Large
    - Non-Aggressive

Assembly Anaconda

- Fun Facts: Many people hate learning about these snakes, as they look very intimidating. In the Totally Official CS1114 Snake **Register**, they are said to love being in **low level** altitudes.
- Characteristics:
    - Non-Venomous
    - Large
    - Aggressive

Java Kingsnake

- Fun Facts: Very befitting of their name, they are **object**ively the most sophisticated snake species. One may even say they are very **classy**.
- Characteristics:
    - Non-Venomous
    - Small
    - Non-Aggressive

Javascript Treesnake

- Fun Facts: Despite its name, they are completely different from the Java Kingsnake. They like to lay on the **nodes** of a tree to **browse** through nearby animals for their next meal.
- Characteristics:
    - Non-Venomous
    - Small
    - Aggressive

C Serpent

- Fun Facts: Can be found in the sea. Has the ability to control their memory, being able to **allocate** parts of their brain for certain tasks and permanently delete information from their memories.
- Characteristics:
    - Venomous
    - Large
    - Non-Aggressive

C++ Cobra

- Fun Facts: Evolved from the C Serpants many years ago. Reports show they have the weird habit of **pointing** at objects with their tails.
- Characteristics:
    - Venomous
    - Large
    - Aggressive

Verilog Viper

- Fun Facts: Many people first see these snakes around **architectures**. Reports claim that they like to chew on **circuit** wires.
- Characteristics:
    - Venomous
    - Small
    - Non-Aggressive

Matlab Mamba

- Fun Facts: Commonly used to introduce mechies to working with snakes. They often hatch **plots** to catch their prey and enjoys **graphical** images.
- Characteritics:
    - Venomous
    - Small

- Aggressive

Finally, print out the identified snake along with its fun fact. A full sample execution would look like this:

```
Is the snake venomous? no
Is the snake small? no
Is the snake aggressive? no
That is a Python Python. One of the largest and most famous snakes. However, they
are pretty slow, and are commonly used to introduce people to learning about
snakes.
```