Welcome to Python

Table of Contents

This lesson contains problems (and solutions!) which test material from lectures 5 through 7 -- you've just learned about non-Karel Python, and we hope these will help you build your understanding! We recommend you use this as a resource rather than running it through front to back -- pick which concepts you need the most practice with, find a problem (or multiple!) that covers those concepts and looks fun, and have a good time! Whether you do these problems yourself or go straight to reading the solutions, we hope these will be helpful.

Recall that to run a program in Python, you can open up your Terminal, type `python name_of_program.py`, (replace `name_of_program.py` with the name of the .py file you want to run) and press enter.

> Ed tip: we've enabled the Mark button on these examples. Unlike the assignment problems, there won't be any tests that run to check your code when you press it, but you can still use the Mark button to keep track of which problems you feel done with and which ones you haven't done yet!

Happy coding!

- Pythagorean theorem
- Weighted coin

Control flow (loops, expressions) (lecture 7)

- Tall enough to ride
- Wholesome machine
- Fibonacci
- International voting age
- Leap year
- First person Karel

Variables & user input

The following problems test your understanding of storing information into variables and prompting a user for input.

How old are they?

Write a program to solve this age-related riddle!

Anton, Beth, Chen, Drew, and Ethan are all friends. Their ages are as follows:

- Anton is 21 years old.
- Beth is 6 years older than Anton.
- Chen is 20 years older than Beth.
- Drew is as old as Chen's age plus Anton's age.
- Ethan is the same age as Chen.

Your code should store each person's age to a variable and print their names and ages at the end.

Agreement bot

Write a program which asks the user what their favorite animal is, and then always responds with "My favorite animal is also ___!" (the blank should be filled in with the user-inputted animal, of course).

Here's a sample run of the program (user input is in bold italics):

```
$ python agreement_bot.py
What's your favorite animal? cow
My favorite animal is also cow!
```

Arithmetic operators, casting

The following problems test your understanding of doing arithmetic operations (+, -, *, **, /, //, %) and casting to different types in Python.

Square number

Ask the user for a number and print its square (the product of the number times itself).

Here's a sample run of the program (user input is in bold italics):

```
$ python square.py
Type a number to see its square: 4
4.0 squared is 16.0
```

Perimeter of a triangle

Prompt the user to enter the lengths of each side of a triangle and then calculate and print the perimeter of the triangle (the sum of all of the side lengths).

Here's a sample run of the program (user input is in bold italics):

```
$ python perimeter.py
What is the length of side 1? 3
What is the length of side 2? 4
What is the length of side 3? 5.5
The perimeter of the triangle is 12.5
```

Fahrenheit to Celsius

Write a program which prompts the user for a temperature in Fahrenheit (this can be a number with decimal places!) and outputs the temperature converted to Celsius.

The Celsius scale is widely used to measure temperature, but places like the US still use Fahrenheit. Fahrenheit is another unit for temperature, but the scale is different from Celsius -- for example, 0 degrees Celsius is 32 degrees Fahrenheit!

The equation you should use for converting from Fahrenheit to Celsius is the following:

degrees_celsius = (degrees_fahrenheit - 32) * 5/9

Here's a sample run of the program (user input is in bold italics):

```
$ python f_to_c.py
Enter temperature in Fahrenheit: 76
Temperature: 76.0F = 24.444444444444443C
```

Remainder division

Ask the user for two numbers, one at a time, and then print the result of dividing the first number by the second and also the remainder of the division. Here's a sample run of the program (user input is in bold italics):

```
$ python remainder.py
Please enter an integer to be divided: 5
Please enter an integer to divide by: 3
The result of this division is 1 with a remainder of 2
```

Constants

The following problems test your understanding of constant variables.

Seconds in a year

Use Python to calculate the number of seconds in a year, and tell the user what the result is in a nice print statement! You should use constants for this exercise -- there are 365 days in a year, 24 hours in a day, 60 minutes in an hour, and 60 seconds per minute.

Tiny mad libs

Write a program which prompts the user for an adjective, then a noun, then a verb, and then prints a fun sentence with those words!

Mad Libs is a word game where players are prompted for one word at a time, and the words are eventually filled into the blanks of a word template to make an entertaining story! We've provided you with the beginning of a sentence (the `SENTENCE_START` constant) which will end in a user-inputted adjective, noun, and then verb.

Here's a sample run (user input is in bold italics):

```
$ python tiny_mad_libs.py
Please type an adjective and press enter. tiny
Please type a noun and press enter. plant
Please type a verb and press enter. fly
Code in Place is fun. I learned to program and used Python to make my tiny plant fly!
```

Dog years

Write a program which asks a user to input an age in human years, and converts it to the equivalent age in dog years.

Dogs are man's best friend, but they have different lifespans than humans. If you divide the average human lifespan by the average lifespan of a dog, you can calculate the multiplier for converting an age in human years to an age in dog years. The average lifespan of a human is 79 years and the average lifespan of a dog is 11 years. So, 1 human year is equal to 79/11 = **7.18 dog years**. To convert, say, 3 human years to dog years, you'd multiply 3 * 7.18 = 21.54 dog years. That means, if your dog is 3 years old in human years, they're past their teenage years in dog years!

Here's a sample run of the program (user input is in bold italics):

```
$ python dog_years.py
Enter an age in human years: 10
That's 71.8 in dog years!
```

E = mc^2

Write a program that continually reads in mass from the user and then outputs the equivalent energy using Einstein's mass-energy equivalence formula (E stands for energy, m stands for mass, and C is the speed of light:

$$E \; = \; m \cdot C^2$$

Almost 100 years ago, Albert Einstein famously discovered that mass and energy are interchangeable and are related by the above equation. You should ask the user for mass (m) in kilograms and use a constant value for the speed of light -- **C = 299792458** m/s.

Here's a sample run of the program (user input is in bold italics):

```
$ python emc2.py
Enter kilos of mass: 100
e = m * C^2...
m = 100.0 kg
C = 299792458 m/s
8.987551787368176e+18 joules of energy!
```

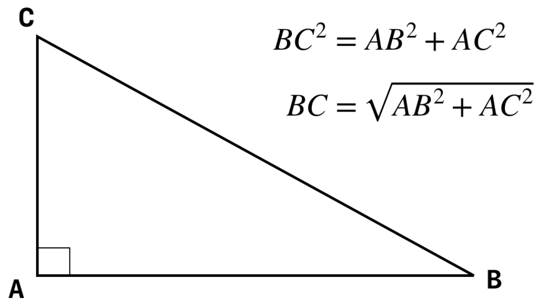
Math and random

The following problems test your understanding of the `math` and `random` modules.

Pythagorean theorem

Write a program that asks the user for the lengths of the two perpendicular sides of a right triangle and outputs the length of the third side (the hypotenuse) using the Pythagorean theorem!

The Pythagorean theorem, named after the ancient Greek thinker, Pythagoras, is a fundamental relation in geometry. It states that in a right triangle, the square of the hypotenuse is equal to the sum of the square of the other two sides.



For instance, let's consider a right triangle ABC, with the right angle located at C. According to the Pythagorean theorem:

$$\vec{BC}^2 = \vec{AB}^2 + \vec{AC}^2$$

which implies that:

$$\vec{BC} = \sqrt{\vec{AB}^2 + \vec{AC}^2}$$

Your code should read in the lengths of the sides AB and AC, and that outputs the length of the hypotenuse (BC). You will probably find `math.sqrt()` to be useful.

Here's a sample run of the program (user input is in bold italics):

```
$ python pythagorean.py
Enter the length of AB: 3
Enter the length of AC: 4
The length of BC (the hypotenuse) is: 5.0
```

Weighted coin

Write a program which will flip a weighted coin.

Coin flips are usually fair, but can be cheated -- a weighted coin is a coin where the probability of heads isn't 50%. Your code should use the `random` module to "flip" coin where the probability of heads is 70% and print the outcome (heads or tails).

Control flow (loops, expressions)

The following problems test your understanding of control flow in Python (`if`, `elif`, `else`, `while` and `for` loops, `==`, `!=`, etc.).

Tall enough to ride

Write a program which asks the user how tall they are and prints whether or not they're taller than a pre-specified minimum height.

In amusement parks (ah, the good old pre-pandemic days...), rollercoasters frequently have minimum height requirements for safety reasons. Assume for now that the *minimum height is 50* of whatever height unit you'd like :)

Here's a sample run (user input is in bold italics):

```
$ python tall_enough.py
How tall are you? 100
You're tall enough to ride!
```

(For an extra challenge, write code which will repeatedly ask a user how tall they are and tell them whether or not they're tall enough to ride, until the user doesn't enter an age at all, in which case the program stops. Curious about how to do this? See the function `tall_enough_extension()` in the solution code!)

Wholesome machine

Write a program which prompts the user to type an affirmation of your choice (we'll use "I am capable of doing anything I put my mind to.") until they type it correctly. Sometimes, especially in the midst of such uncertain times, we just need to be reminded that we are resilient, capable, and strong; this little Python program may be able to help!

Here's a sample run of the program (user input is in bold italics):

```
$ python wholesome.py
Please type the following affirmation: I am capable of doing anything I put my mind to
.
Hmmm
That was not the affirmation.
Please type the following affirmation: I am capable of doing anything I put my mind to
.
I am capable of doing anything I put my mind to.
That's right! :)
```

Note that you can call `input()` with no prompt and it will still wait for a user to type something!

Fibonacci

Write a program to print terms in the Fibonacci sequence up to a maximum value.

In the 13th century, the Italian mathematician Leonardo Fibonacci, as a way to explain the geometric growth of a population of rabbits, devised a mathematical sequence that now bears his name. The first two terms in this sequence, Fib(0) and Fib(1), are 0 and 1, and every subsequent term is the sum of the preceding two. Thus, the first several terms in the Fibonacci sequence look like this:

Fib(0) = 0
Fib(1) = 1
Fib(2) = 1 (0 + 1)
Fib(3) = 2 (1 + 1)
Fib(4) = 3 (1 + 2)
Fib(5) = 5 (2 + 3)

Write a program that displays the terms in the Fibonacci sequence, starting with Fib(0) and continuing as long as the terms are less than 10,000 (you should store this value as a constant!). Thus, your program should produce the following sample run:

```
$ python fibonacci.py
0
1
1
2
3
5
8
13
21
34
55
89
144
233
377
610
987
1597
2584
4181
6765
```

International voting age

Write a program which asks a user for their age and lets them know if they can or can't vote in the following three fictional countries.

Around the world, different countries have different voting ages. In the fictional countries of Peturksbouipo, Stanlau, and Mayengua, the voting ages are very different:

- the voting age in Peturksbouipo is 16 (in real life, this is the voting age in, for example, Scotland, Ethiopia, and Austria)
- the voting age in Stanlau is 25 (in real life this is the voting age in the United Arab Emirates)
- the voting age in Mayengua is 48 (in real life, as far as we can tell, this isn't the voting age anywhere)

Your code should prompt the for their age and print whether or not they can vote in Peturksbouipo, Stanlau, or Mayengua.

Here's a sample run of the program (user input in bold italics):

```
$ python voting.py
How old are you? 20
You can vote in Peturksbouipo where the voting age is 16.
You cannot vote in Stanlau where the voting age is 25.
You cannot vote in Mayengua where the voting age is 48.
```

Leap year

Write a program that reads a year from the user and tells whether a given year is a leap year or not.

A leap year (also known as an intercalary year or bissextile year) is a calendar year that contains an additional day (or, in the case of a lunisolar calendar, a month) added to keep the calendar year synchronized with the astronomical year or seasonal year. In the Gregorian calendar, each leap year has 366 days instead of 365, by extending February to 29 days rather than the common 28.

In the Gregorian calendar, three criteria must be checked to identify leap years:
1. The given year must be evenly divisible by 4;
2. If the year can also be evenly divided by 100, it is NOT a leap year; unless:
3. The year is also evenly divisible by 400. Then it is a leap year.

Your code should use the above criteria to check for a leap year and then print either "That's a leap year!" or "That's not a leap year."

First person Karel

Write a program that lets you play Karel in a rectangular world (moving and turning left and not walking off of the world, all that good stuff)!

Have you ever wanted to be Karel, wandering about a nice rectangular world, not a worry in the world, just moving and turning left whenever you'd like? You can emulate this feeling by writing a console-based first person Karel program! For simplicity, assume that beepers and painting are out of the question, and the world has no walls. *All you should implement is moving and turning left.* Store the *number of columns and rows in your "world" as constants*, and assume you *start in row 1 column 1 facing East*. Prompt the user for an action ("move" or "turn left" until they press enter without typing anything, in which case the result of `input()` will be an empty string, and you can stop looping).

Recall that you can only move if you're not about to move off of the world! This means you can only move in the following cases:

- facing East and current column is less than the number of columns (move right)
- facing West and current column is greater than 1 (move left)
- facing North and current row is less than the number of rows (move up)
- facing South and current row is greater than 1 (move down)

Here's what turning left does to the direction you (Karel) are facing:

- if you're facing East, turning left will make you face North
- if you're facing North, turning left will make you face West
- if you're facing West, turning left will make you face South
- if you're facing South, turning left will make you face East

Here's a sample run of the program (user input is in bold italics):

```
$ python karel.py
Welcome to first person Karel. You're at row 1, column 1, facing East (facing column 3
)
What would you like to do? You can move and turn left. Press enter to finish. move
You moved one step forward and now you're at row 1 column 2
What would you like to do? You can move and turn left. Press enter to finish. turn lef
t
You turned left and are now facing North.
What would you like to do? You can move and turn left. Press enter to finish. turn lef
t
You turned left and are now facing West.
What would you like to do? You can move and turn left. Press enter to finish. move
You moved one step forward and now you're at row 1 column 1.
```

```
What would you like to do? You can move and turn left. Press enter to finish. move
You can't move forward!
What would you like to do? You can move and turn left. Press enter to finish.
You've ended up at row 1 and column 1 facing West.
```

This may be a confusing problem (it's certainly a pretty big problem)! If you'd like a hint about how to get started, scroll down and keep reading (though we recommend you give it your best shot first!). If you're feeling good, stop reading right here and go right ahead!

As a hint, this is what the beginning of our solution code looks like:

```
N_COLS = 3 # notice these constants!
N_ROWS = 3

def main():
    print("Welcome to first person Karel. You're at row 1, column 1, facing East (facir
    facing_direction = 'East' # this variable will keep track of the way Karel is facir
    curr_col = 1 # this variable ...
    curr_row = 1 # ... and this one keep track of Karel's position in the world! they m
    action = input("What would you like to do? You can move and turn left. Press enter
    # ... more code! there's a while loop that starts on this line, but our hint ends h
```