Homework Day 4 Instructions

Problem 1

Write 3 functions, each converting one unit to another:

A function `lbs2kg` which takes in a variable `lbs` (float) and returns the converted value in kg (1 lb = 0.453592 kg)

A function `feet2meters` which takes in a variable `feet` (float) and returns the converted value in meters (hint: 1 ft = 0.3048 m)

A function `inches2meters` which takes in a variable `inches` (float), and returns the converted value in meters (hint: 1 lb = 0.0254 m)

Problem 2

At Haley's Diner, employees generally work only up to 40 hours each week at a certain rate, but when they have to work overtime, their rate increases by 50%.

For example, employee A gets a rate of $20/hr and worked 30 hours this week, the total pay is $600.

- 30 * 20 = 600

Employee B also gets a rate of $20/hr but worked 50 hours this week, the total pay is $1100.

- (40 * 20) + (10 * 30)
- where 30 = 1.5 * 20, and 10 is the number of hours they worked overtime

Write a function called `weekly_pay` which takes in two variables `hours` and `rate` and returns how much the employee earned as an output `pay` .

Problem 3

Preamble: In this problem, you will implement a popular game in South Korea called the 3-6-9 (sam-yuk-gu) game. In this game, you play with a number of people (the more, the better!) sitting in a circle. Then, people in the group take turns counting to a number, starting from 1, 2 and so on. However, when the number has a '3', or '6', or '9', you do not say the number and clap instead. The number of times you clap will equal the number of times ' 3-6-9 ' occurs in that number. For example, you clap one time for '6' but clap twice for '33' and '39'. You will lose the game if you forget to clap correctly.

Instructions

Write a function lets_play_369(N) which receives the 'last number to be played'/total number of rounds as input N

- your function should count the number of claps happening in the game and return it as an output count_clap
- N is an integer less than 100
- you are not required to print out the game play record. However, it is highly recommended to visualize the counts while playing

```
count = lets_play_369(13)
# output:
# 1,
# 2,
# clap! (3),
# 4,
# 5,
# clap! (6),
# 7,
# 8,
# clap! (9),
# 10,
# 11,
# 12,
# clap! (13),
print(count)
# output: 4
```

Problem 4

Write a function `fib` which takes in a number `L` (which must be at least 1) and prints out the fibonacci sequence of that many numbers

you will need to track how many numbers you have printed, as well as the values of the numbers you are working with in the sequence!

optional challenge: can you make the code print the sequence in one line?

```
fib(3)
# will print 1, 1, 2
```

Problem 5 -- Optional

This problem explores a concept called *recursion* which we do not have time to cover in class. Recursion is when a function calls itself, so that multiple copies of it run until some value / option is reached. An example is included below, which you can follow with environment diagrams. Try to see what this function does (it is a computation you know!).

```
def mystery(a, b):
```

```python
if b == 1:
  c = a
else:
  c = a + mystery(a, b - 1)
return c
```