# DRAWING A BLANK?

## In General

I have this problem to solve, but I don't know where to start.

1st: Write down exactly what you want to do

2nd: Write down what you know

3rd: Divide & Conquer: Break the problem into smaller pieces

4th: Now focus on a smaller piece from step 3.

Still drawing a blank? Working on the smaller piece, repeat steps 1-4.

This works well for most problems in life, and it is a GREAT approach to solve coding challenges.

## Writing a Function

I have this new function to write, but I don't know where to start.

def gordon( foodOrder, cookBook ):

### 1st: Write down exactly what you want to do.

What will the function do?

# gordon takes food request from customer(waiter) cooks then gives finished food( good or bad ) to waiter to bring out for customer

def gordon( foodOrder, cookBook ):

### 2nd: Write down what you know

What are the arguments? What does an example call look like?

# gordon takes food request from customer(waiter) cooks then gives finished food( good or bad ) to waiter to bring out for customer

# Called like: foodPlate = cooks.gordon( foodOrder, cookBook )

# foodOrder: customer order. "cookedRice"

# cookBook : dict of recipes organized by appliance. {'riceCooker': { 'cookedRice': ["rawRice", "smallWater"] }, .... }

def gordon( foodOrder, cookBook ):

### 3rd: Divide & Conquer: Break the problem into smaller pieces

What steps need to happen in your function? Write them down.

# gordon takes food request from customer(waiter) cooks then gives finished food( good or bad ) to waiter to bring out for customer

# Called like: foodPlate = cooks.gordon( foodOrder, cookBook )

# foodOrder: customer order. "cookedRice"

# cookBook : dict of recipes organized by appliance. {'riceCooker': { 'cookedRice': ["rawRice", "smallWater"] }, .... }

def gordon( foodOrder, cookBook ):

# XXX cook looks up order in cookbook to find recipe

# XXX use the cookbook to get the ingredients for the order, then

# XXX use the cooking implement to make the food and return

# XXX call the right appliance with the recipe. for example : foodPlate = appliances.riceCooker(["beans","meat","smallWater"],cookBook)

# XXX remember, cook needs to return food from appliance as a foodplate for the waiter

### 4th: Now focus on a smaller piece from step 3.

Now start with first XXX, i.e. look up order in cookbook to find recipe.

Drawing a blank? Go to step 1.

## Writing a For Loop

I have a for loop to write, but I don't know where to start.

# gordon takes food request from customer(waiter) cooks then gives finished food( good or bad ) to waiter to bring out for customer

# Called like: foodPlate = cooks.gordon( foodOrder, cookBook )

# foodOrder: customer order. "cookedRice"

# cookBook : dict of recipes organized by appliance. {'riceCooker': { 'cookedRice': ["rawRice", "smallWater"] }, .... }

def gordon( foodOrder, cookBook ):

# XXX cook looks up order in cookbook to find recipe

for ........... ?

### 1st: Write down exactly what you want to do

# Cook looks up order in cookbook to find recipe

# I should look at every recipe in cookBook, then see if the food order matches it.

for ........... ?

### 2nd: Write down what you know

What are the function arguments? Are there other variables defined in the function before my for loop?

# Cook looks up order in cookbook to find recipe

# I should look at every recipe in cookBook

# known variables: foodOrder, cookBook

for ........... ?

### 3rd: Divide & Conquer: Break the problem into smaller pieces

Find a template, then adapt that template to your needs.

First, find a good template. Single for loop? Nested for loop?

In this case, I need to look at every recipe in cookBook. I only know foodOrder and cookBook.

Recipes are not in foodOrder. Recipes are in cookBook.

Recipes are 2 levels deep in cookBook, inside appliance dict (e.g. 'riceCooker'), which is inside cookBook dict. I need a nested for loop.

#### My nested for loop template:

cups = ["big cup", "little cup"]

bowls = ["big bowl", "little bowl"]

cabinet = {'cups': cups, 'bowls': bowls}

for dishType, dishes in cabinet.items():

for dish in dishes:

print( dish )

The template looking at all the items, made up of 'dishType' and dishes in cabinet (outer loop).

Then it looks at every dish in dishes (inner loop).

#### My actual for loop, based on the template:

I want all recipes in cookBook (a dict of recipes organized by appliance): {'riceCooker': { 'cookedRice': ["rawRice", "smallWater"] }, .... }

I will look at all the 'appliances' and recipes in cookbook (outer loop).

Then, I will look at every recipe in recipes (inner loop).

So, cabinet is like cookBook. dishType is like appliance. dishes are like recipes. dish is like recipe.

for appliance, recipes in cookBook.items():

for recipe in recipes:

print( recipe )

\*\*\*\*\*\* This is close, but it has a bug. We stuck too closely to the template. See below. Can you fix it? \*\*\*\*\*

### \*\*\* Use python interpreter to look at each piece so you know what they look like, and to check your work!

>>> cups = ["big cup", "little cup"]

>>> bowls = ["big bowl", "little bowl"]

>>> cabinet = {'cups': cups, 'bowls': bowls}

>>> cabinet

{'cups': ['big cup', 'little cup'], 'bowls': ['big bowl', 'little bowl']}

>>> for dishType, dishes in cabinet.items():

... for dish in dishes:

... print(dish)

...

big cup

little cup

big bowl

little bowl

>>>

>>> import temp

>>> cookBook = temp.getCookBook()

>>> cookBook

{'dessert': {'iceCream': ['icecream'], 'chipIceCream': ['icecream', 'chips']}, 'riceCooker': {'porridge': ['rawRice', 'bigWater'], 'cookedRice': ['rawRice', 'smallWater'], 'chili': ['beans', 'meat', 'smallWater']}, 'microwave': {'yesterdays special': ['teaBag', 'bugs', 'smallWater'], 'tea': ['teaBag', 'smallWater'], 'bugsoup': ['bugs', 'bigWater'], 'bugs': ['bugs'], 'personal Pizza': ['bread', 'cheese', 'tomatoSauce']}}

>>>

>>> for appliance, recipes in cookBook.items():

... for recipe in recipes:

... print( recipe )

...

iceCream

chipIceCream

porridge

cookedRice

chili

yesterdays special

tea

bugsoup

bugs

personal Pizza

## I don't get what I'm seeing

---------------------------

Try printing it more nicely..

>>> import temp

>>> cookBook = temp.getCookBook()

>>> cookBook

{'dessert': {'iceCream': ['icecream'], 'chipIceCream': ['icecream', 'chips']}, 'riceCooker': {'porridge': ['rawRice', 'bigWater'], 'cookedRice': ['rawRice', 'smallWater'], 'chili': ['beans', 'meat', 'smallWater']}, 'microwave': {'yesterdays special': ['teaBag', 'bugs', 'smallWater'], 'tea': ['teaBag', 'smallWater'], 'bugsoup': ['bugs', 'bigWater'], 'bugs': ['bugs'], 'personal Pizza': ['bread', 'cheese', 'tomatoSauce']}}

Huh?

>>> import pprint

>>> pp = pprint.PrettyPrinter(indent=3)

>>> pp.pprint( cookBook )

{ 'dessert': { 'chipIceCream': ['icecream', 'chips'],

'iceCream': ['icecream']},

'microwave': { 'bugs': ['bugs'],

'bugsoup': ['bugs', 'bigWater'],

'personal Pizza': ['bread', 'cheese', 'tomatoSauce'],

'tea': ['teaBag', 'smallWater'],

'yesterdays special': ['teaBag', 'bugs', 'smallWater']},

'riceCooker': { 'chili': ['beans', 'meat', 'smallWater'],

'cookedRice': ['rawRice', 'smallWater'],

'porridge': ['rawRice', 'bigWater']}}

>>>

Ohhh, nice.