Chapter 9.2

So now we've learned a bit about the basic mechanics of dictionaries and

now we're going to solve a problem.

And the problem we're going to solve is,

let's assume that you're going to see a bunch of words.

And you want to know the most common word.

This is, another word for this is the histogram problem. It's like how many of

these things, these things, and these things.

And we're going to use a dictionary for it eventually.

But before we do that I want you to run through a little exercise and

I'm going to show you a number of names.

And I want you to keep track of which one is the most common and

how many times did you see the name.

But I'm only going to show them to you one at a time.

And so the purpose of this exercise is less about you knowing what the number is.

And more about trying to watch your brain and

figure out how your brain struggles with this problem.

And then we'll show you how Python might struggle with this problem and

how we solve this problem in Python.

So are we ready?

So I'm going to show them to you one at a time and grab a piece of paper,

do whatever you want, ready?

从 :1:16 开始播放视频并学习脚本1:16

How are you doing?

从 :1:27 开始播放视频并学习脚本1:27

So that was last name.

从 :1:30 开始播放视频并学习脚本1:30

What was the most common name and how often did it occur?

Maybe you have to scroll back, go back and forth, scroll back and forth.

从 :1:38 开始播放视频并学习脚本1:38

This turns out to be a problem that humans are really not very good at it,

especially if I was going to give you a million words instead of just 16 or

12 or whatever I gave you.

从 :1:50 开始播放视频并学习脚本1:50

The screen I'm going to show you next, you'll see humans are good at this.

Except humans are good when they can see all the data.

Humans are not good when they only see one little bit of the data at a time.

So now your brain goes like, oh,

从 :2:2 开始播放视频并学习脚本2:02

what am I doing? Let's see how many marquard, ah, that's only three.

zhen looks like, oh, zhen's a lot, zhen is one, two, three, four,

five and has anybody got more than zhen? No zhen is like, so zhen. Four. Yeah,

Oh, no actually zhen five. That's another a problem we humans have.

We will miss one, we got it pretty good.

So the way our brain it looked around.

My eyes were going this way and this way and this way and

then I make a hypothesis and test the hypothesis.

从 :2:34 开始播放视频并学习脚本2:34

That's not how computers think. They don't like that.

They're not dynamic like we are.

Even though sometimes we write programs that make them seem dynamic,

that's not how they work.

So if you had to really solve this program, you probably if you were smart,

would have made a little piece of paper.

And you would have a drawn a little picture like this.

And you would have done, each time you would look at the name,

you would check to see if it was a name that you already saw.

And if it was a name you already saw, you'd add one to that name.

And you're like okay, there's one more of those and

there is one of those now I have another one of those and I got one of those,

two of those and whatever and, or here is a new one here and I got one of those and

on and on and on just tick these things off as you move through.

And then when you're all done now you go look at these numbers.

You know 5, 7, 6, 5, 1 and you're like okay that's the one that I want.

And so this is a little set of counters and you can think of this as a histogram.

It's like a little histogram that's growing and

each time you see one you add a little bit more, and you add a little bit more, and

you add a little more and more, add a new one.

Grow that one, grow this one, grow this one,

grow this one, and then you're all done you got sort of the tallest histogram.

You could think of these numbers as growing histograms with the names

here on the horizontal axis.

That would be a way to mentally think about this problem

for a human to think about the problem.

And so, we're going to use dictionaries, and we're in those dictionaries we

are going to make the keys be these things, the actual strings, these names.

And then the values will be the current count, and

then we're going to update those counts.

So that's the data structure that we're going to build

in Python. And it's really common for histograms or

any other word counting, various other things, any kind of frequency.

So what we're going to do is we going to take these names, csev, or cwen, and

we're going to use those as the keys and

we're going to use strings as our keys in our dictionary.

So, we start with our dictionary and the first time we see csev, so

we haven't seen them yet.

And so we put we'll put 1 in the, in the, there, under csev, under the tag csev.

We see cwen for the first time, so we put 1 under for her, and then we print.

It's like this is our current histogram as we got it so far.

csev has 1, cwen has 1.

csev has 1, cwen has 1, but we're not done yet.

Now we see cwen again and we go grab, what was the previous number we had for

cwen, well it was 1.

Add 1 to that so that's kind of like ticking this and then stick it back in so

that's storing it in.

So now we have this dictionary that's kind of growing as time is progressing.

So cwen has 2, chuck has 1, and away we go.

So you get the idea that if we use the names of these people from our input data

as the keys in the dictionaries

and the values are the counts, then we can easily make a histogram that we can expand

every time we see a new name.

Update the old ones, update the new ones, add a new one etc, etc, etc.

That this works in a very nice and dynamic way.

从 :5:32 开始播放视频并学习脚本5:32

Now, it wouldn't be Python if we didn't talk about the kinds of things that you

can do that cause tracebacks.

从 :5:39 开始播放视频并学习脚本5:39

If this is an empty dictionary you can't go grab in a key that doesn't exist.

I wish Python did this differently but it's not how it works.

Python basically does not allow you to look at a key that doesn't exist.

Actually a list works the same thing. If there's four things in a list and

you look at sub 10,

the list blows up too. We saw this with strings. If you look for a character beyond

the end of the string,

the strings are unhappy as well.

So Python is unhappy when you go and look for a key that doesn't exist.

But, like in all situations in Python, there is a workaround. Right?

It's telling us KeyError csev but there's an in operator.

We used it for strings, now we've used it for lists, and

now we're using it in dictionaries.

And it asks not the value, it's saying is this key csev in this dictionary ccc?

And in this case, because we just created this dictionary and we're doing the question,

it's False.

So now we can write an if statement so

that we can do one thing if the key is there and another thing if it's not.

And if you go back to the notion of did we see this person before, yes we did.

Let's add a number. Oh, and then there's a new person, and let's set that person to 1.

So there's two things we're doing.

If they exist, add 1. If they don't exist, make a new one and

set their count to 1.

So, it's not just enough to make a new one, but we have to make a new one and

set their count to 1.

So, what does that code look like?

从 :7:2 开始播放视频并学习脚本7:02

Well, it's conditional execution.

It's an if statement.

So, and eventually we'll read this data from a file, but here are some names.

And we're going to go cruising through there.

We'll start by making a dictionary, it's a dictionary of counts.

Now again, I'm just calling it counts because it's a plural,

because it helps you understand it.

You don't have to name dictionaries with plural variables, although we commonly do it.

We actually very commonly do that.

So I'm going to have this for loop.

The name is just going to go through as the duration variable.

And the if code is the intelligent part.

So we're asking the question,

if the name we're looking at is not in our dictionary, then set counts sub name = 1.

This is a variable, in this case of csev, cwen, csev, zqian , so

that's getting things started. That's adding a new entry and setting it to 1.

If, on the other hand, it's already in there so we find zqian in this case and

zqian has like 2 currently.

If it's in there then we're going to run the else code which

means we're going to take this 2 out.

We're going to add 1 to it and then we're going to put it back in.

Sp that's the idea of adding 1 or incrementing an entry in there.

You pull it out, add 1 to it, and put it back in.

So as this runs it both makes new ones and then updates the existing ones.

从 :8:21 开始播放视频并学习脚本8:21

And like all loops, this is like histogram logic, when it's all said and

done we come out the bottom.

从 :8:28 开始播放视频并学习脚本8:28

And we have a histogram. csev we've seen twice, zgian have seen once, and

cwen we've seen twice.

So we end up with a histogram. So this is like the histogram logic and

again, I just call your attention to this slide.

Later you're going to be doing a lot of stuff and

you want to come back to the slide if you're confused by

this is what it is. Because we're going to show you in a second

a quicker way to do this.

But this notion of if-then-else.

If there's no key in the dictionary, put it in.

If there is a key in the dictionary,

do something to the existing value that's already there.

That's something we're going to do so many times.

And it turns out we do this so many times in Python that they have built

something in that takes care of this for us.

And the basic idea is that there is going to be these four lines. We're going to

do something if it's there. If not, we're going to set it to 0 or something.

So this get method so it counts is a dictionary there's a get method.

You can't use this on lists or strings, it's just part of dictionaries.

So what counts.get is, it says go look up in counts,

use this as the key and this as the default.

从 :9:38 开始播放视频并学习脚本9:38

Meaning this is the value I get back if the key doesn't exist.

So either it looks it up and it finds it.

So csev, it finds a 2 and gives us a 2.

If I looked up Bob here it would give me back 0.

The key thing it does is doesn't traceback, right?

So this works whether the key exists or not and you pick a default. Okay?

从 :10: 开始播放视频并学习脚本10:00

That's the get, that's the get.

It's a method in dictionaries. Okay, so

this is how we make a contraction. This becomes kind of the idiom,

meaning you can look at this and you need to understand it.

But after the tenth time you'll just be putting this line of code in and

you say oh, this is our little histogram trick. So if we look at it in slow motion.

We're going through the names again [SOUND] three, four,

five times, and we're saying, okay.

We're going to set the count of the name for

that particular name, get the current count of the name or 0 and

then add 1 to it. So if there's already a 4 in there,

this becomes 4 plus 1 is 5.

If there's nothing in there we get 0 plus 1 and then we get a 1 and

we store that in.

So if it is a brand new key.

This is for new keys that are not there and

then existing keys that are already there it pulls out uses the get.

And so this combines the if and the else in one line.

So those four lines become one line, and so

this does exactly the same thing as that thing with the if-then-else in

that loop that we wrote in the previous slide with the if-then-else in it.

So this is how we simplify counting.

And again, this is an idiom, we're going to do this over and

over and over again.

Any time we see a set of things and we want to build a count,

we're just going to use this idiom over and over and over again.

So now, up next, we're going to build this into a complete application

where we really read through a file and we split the file.

And this is the code that we looked at when we first started this class.

So we're coming back finally, and we should now understand every single word.