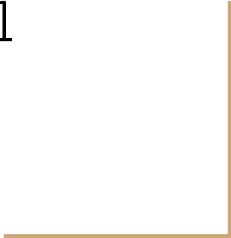


# Programming, Problem Solving, and Algorithms

CPSC203, 2019 W1



# Announcements

Project 2 is released. Due 11:59p, Nov 7.

“Problem of the Day” continues!

## Today:

Dictionaries

MHALL

# Characterizing Mary



	C	D	E	G
C	0	<u>2</u>	<u>0</u>	<u>0</u>
<i>from</i> D	<u>3</u>	<u>3</u>	<u>4</u>	<u>0</u>
E	0	5	5	<u>1</u>
G	0	0	1	1

# Dictionaries



Data structures for storing (key, value) pairs.

*index data*

Suppose we just want to count the notes in a song...

```
song = ['E', 'D', 'C', 'D', 'E', ...]
```

```
d = {}
```

*d = defaultdict(int)*  
*{'E': 11, 'C': 3 ... empty dict, l = []}*  
[For note in song:

```
d[note] += 1
```

```
print(d['E'])
```

Does this work? *orig no - we needed a defaultdict*

# Dictionaries

Is this a dictionary?

Describe the keys:

~~Strings~~

Describe the values:

~~list of floats~~

How do you refer to the entry whose quantity is 440.00?

freq['A'][4]

freq['Eb'][2]

What's freq['F']?

list ...

what structure does freq[4]['A'] imply?

Weird and useful things: freq.keys() returns

freq.values()

a list of keys ['C', 'Db', 'D'...] [[...], [...], [...]]

```
freq = {  
    "C": [16.35, 32.70, 65.41, 130.81, 261.63, 523.25, 1046.50, 2093.00, 4186.01],  
    "Db": [17.32, 34.65, 69.30, 138.59, 277.18, 554.37, 1108.73, 2217.46, 4434.92],  
    "D": [18.35, 36.71, 73.42, 146.83, 293.66, 587.33, 1174.66, 2349.32, 4698.64],  
    "Eb": [19.45, 38.89, 77.78, 155.56, 311.13, 622.25, 1244.51, 2489.02, 4978.03],  
    "E": [20.60, 41.20, 82.41, 164.81, 329.63, 659.26, 1318.51, 2637.02],  
    "F": [21.83, 43.65, 87.31, 174.61, 349.23, 698.46, 1396.91, 2793.83],  
    "Gb": [23.12, 46.25, 92.50, 185.00, 369.99, 739.99, 1479.98, 2959.96],  
    "G": [24.50, 49.00, 98.00, 196.00, 392.00, 783.99, 1567.98, 3135.96],  
    "Ab": [25.96, 51.91, 103.83, 207.65, 415.30, 830.61, 1661.22, 3322.44],  
    "A": [27.50, 55.00, 110.00, 220.00, 440.00, 880.00, 1760.00, 3520.00],  
    "Bb": [29.14, 58.27, 116.54, 233.08, 466.16, 932.33, 1864.66, 3729.31],  
    "B": [30.87, 61.74, 123.47, 246.94, 493.88, 987.77, 1975.53, 3951.07]  
}
```

# Dictionaries

$\{$ 
 $\begin{matrix} & C & D & E & G \\ 'C': & [0, 2, 0, 0] \\ 'D': & [3, 3, 4, 0] \end{matrix}$ 
 $\}$

So how can we use a dictionary to represent this table?

Sketch what we want:

$\{$ 
 $\begin{matrix} 'C': \{ 'D': 2 \}, \\ 'D': \{ 'C': 3, 'E': 4 \} \end{matrix}$ 
 $\}$

Declaration:

$tab[D][E] \rightarrow 4$

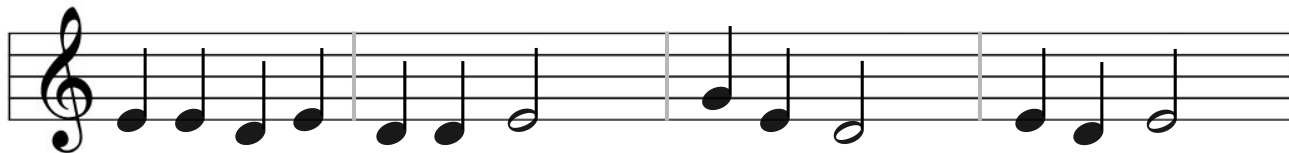
$tab = defaultdict(lambda: defaultdict(int))$

Build the table, given a song.

*list of notes*

	<i>to</i>			
	<u>C</u>	<u>D</u>	<u>E</u>	<u>G</u>
<u>C</u>	0	2	0	0
<u>D</u>	3/10	3/10	4/10	0
<u>E</u>	0	5	5	1
<u>G</u>	0	0	1	1

# Building a Song



1. Randomly choose a start note and put it in a list
2. for 25 notes, in the rhythm of MHaLL
  - a. Generate a new note
  - b. Put the new note in the list
3. play the list of notes

	C	D	E	G
C	0	1.0	0	0
D	0.3	0.3	0.4	0
E	0	0.45	0.45	0.1
G	0	0	0.5	0.5

# The Technical Details

You have just learned about a particular type of random process called a *Markov Chain*.

We modelled it using a *transition table*, or a *finite state machine*, and we used it as the basis for an algorithm to generate music.

Now let's look at some code!

<https://github.students.cs.ubc.ca/cpsc203-2019w-t1/LecMHALL/>



# Other Applications

## PageRank: Google's first search algorithm

Some pages are likely to “follow” (be linked from) others.

Rank of page is based on the probability that you'll be there at any moment

## Natural Language Processing

Some words are more likely to follow others.

“I just ate the whole desert” probably has a misspelling.

“For dinner I \_\_\_ ... ” next word is probably “ate”

## DNA matching

## Chemical reaction simulation

Many others...

But I thought we were talking about graphs...

	C	D	E	G
C	0	1.0	0	0
D	0.3	0.3	0.4	0
E	0	0.45	0.45	0.1
G	0	0	0.5	0.5

<http://setosa.io/markov>

# POTD #26 Tue

<https://github.students.cs.ubc.ca/cpsc203-2019w-t1/potd26>

Describe any snags you run into:

1. Line \_\_\_\_: \_\_\_\_\_
2. Line \_\_\_\_: \_\_\_\_\_
3. Line \_\_\_\_: \_\_\_\_\_
4. Line \_\_\_\_: \_\_\_\_\_
5. Line \_\_\_\_: \_\_\_\_\_

# ToDo for next class...

POTD: Continue every weekday! Submit to repo.

Reading: TLACS Ch 10 & 12 (lists and dictionaries)

References:

<https://brilliant.org/wiki/markov-chains/>

<https://medium.com/@eightlimbed/counting-on-pythons-defaultdict-b652204780bd>