## Repeating With Loops

## Learning Objectives

* Explain what a for loop does.
* Correctly write for loops that repeat simple commands.
* Trace changes to a loop variable as the loops runs.
* Use a for loop to process multiple files

**Part – Intro to Loops**

Why we need loops

What we would do without loops:

Clearing the workspace

Introduce placePokemon and movePokemon functions. Show where the images are and explain input arguments for each fucntion

% clear workspace

clear all

clc

% Place a single pokemon on map

pokemonAxis = placePokemon(1, 0);

Using variables as placeholders (replace number with variable)

Pokenum =39; % variable containing pokemon number

% Place a single pokemon on map

pokemonAxis = placePokemon(Pokenum, 0);

disp(['Hooray! Successfully placed pokemon # ' num2str(pokenum) ' on the map! ']);

Turning it into a loop (just write syntax)

% display pokemons in a loop

for pokenum=1:5 % variable containing pokemon number

figure;

pokemonAxis = placePokemon(pokenum, 0);

disp(['Hooray! Successfully placed pokemon # ' num2str(pokenum) '

on the map! ']);

end

Explain syntax

Set breakpoint and discuss

Step through (F10)

Do every second pokemon (explain syntax 1:2:10)

Make an unordered list of pokemon numbers of arbitray size and use size function

randPokeList = randi(151,5,1);

for pokenum=1:size(randPokeList,1)

figure;

pokemonAxis = placePokemon(randPokeList(pokenum), 0);

disp(['Hooray! Successfully placed pokemon # '

num2str(randPokeList(pokenum)) 'on the map! ']);

end

CHALLENGE:

% Using the worked example as a guide, write a script places each Pokemon on the map for 1 sec and then places the next pokemon on the same figure. Hint: check documentation for ‘pause’ function

% Change your loop so that it only displays only pokemons 4,7,13,16,28

% and 33 for a second and then closes the figure.

EXTENSION:

%% Extension

% Modify your loop to instead place multiple pokemons on the same figure. Hint: Open placePokemon function and study the for loop implemented in it.

% Save this figure as a png file to disk and tweet in! Don’t forget to use @ResPlat :D

**Part 2 – Using A Loop To Analyse Data**

Let’s write a loop that tries to print the names of each one of our files:

Write script and run

% loop to print names of data files

for idx = 1:4

file\_name = sprintf('inflammation-%d.csv', idx);

disp(file\_name);

end

Explain sprint:

“The sprintf function is useful when we want to generate MATLAB strings based on a template. In our case, that template is the string inflammation-%d.csv. sprintf generates a new string from our template by replacing the %d with the data referred to by our second argument, idx.”

Trace execution of loop

Filename that’s printed is wrong

Modify template:

“We’ve replaced %d in our earlier template with %02d. With this, we’re specifying that we want our data to be displayed with a minimum width of 2 characters, and that we want to pad with 0 for data that isn’t at least 2 digits long.”

for idx = 1:4

file\_name = sprintf('inflammation-%02d.csv', idx);

disp(file\_name);

end

Use this to loop through data files in analyse.m

Make file\_name and img\_name variables, change image saving function

file\_name = sprintf('inflammation-%02d.csv', idx);

img\_name = sprintf('patient\_data-%02d', idx);

CHALLENGE:

% Insert for loop so that the script loads in each data file sequentially,

% and saves out a heatmap for each data file (using imagesc).

% HINT: loop using the variable idx.

**Part 3 – Debugging For Loops**

Make an error in the loop, debug it

CHALLENGE:

* Pie.