# Traversing a List

An supplemental lesson after Mission 9

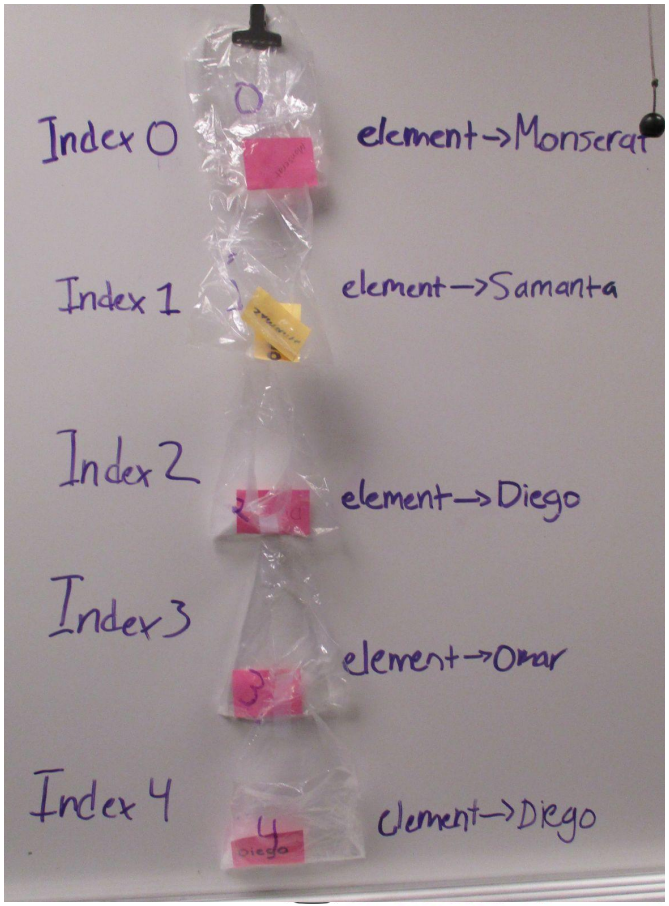
# Warm-up

Lists

# Lists

You have used lists for several assignments now.

What do you remember about lists?



# Activity

Work through these activities at the white boards.

# Traversing a list

Do you remember Mission 7 and Mission 8?

- Mission 7 - Personal Billboard
- Mission 8 - AnswerBot

Both missions used a list.

- In Mission 7, you went through the list one item at a time using the buttons
- In Mission 8, you selected a random item from the list

FIRIA LABS

# Traversing a list

What if you wanted to go through all the items of the list in order, but want it go automatically, without needing to press a button?

This is called “**traversing**”.

- It means that you are traveling, or traversing, through a list one item (or element) at a time.
- Start with the first item (index is 0) and go to the last item (index is len(list) - 1)
- This is **sequential**

# Traversing a list

Traversing a list can be very useful in many ways.

Some reasons are:

- Displaying all items one at a time, in order, sequentially
- Looking to see if a specific value is in the list
- Creating a sub-list from the complete list, like all numbers less than 10

# Traversing a list

Traversing a list is accomplished using a loop (**iteration**).

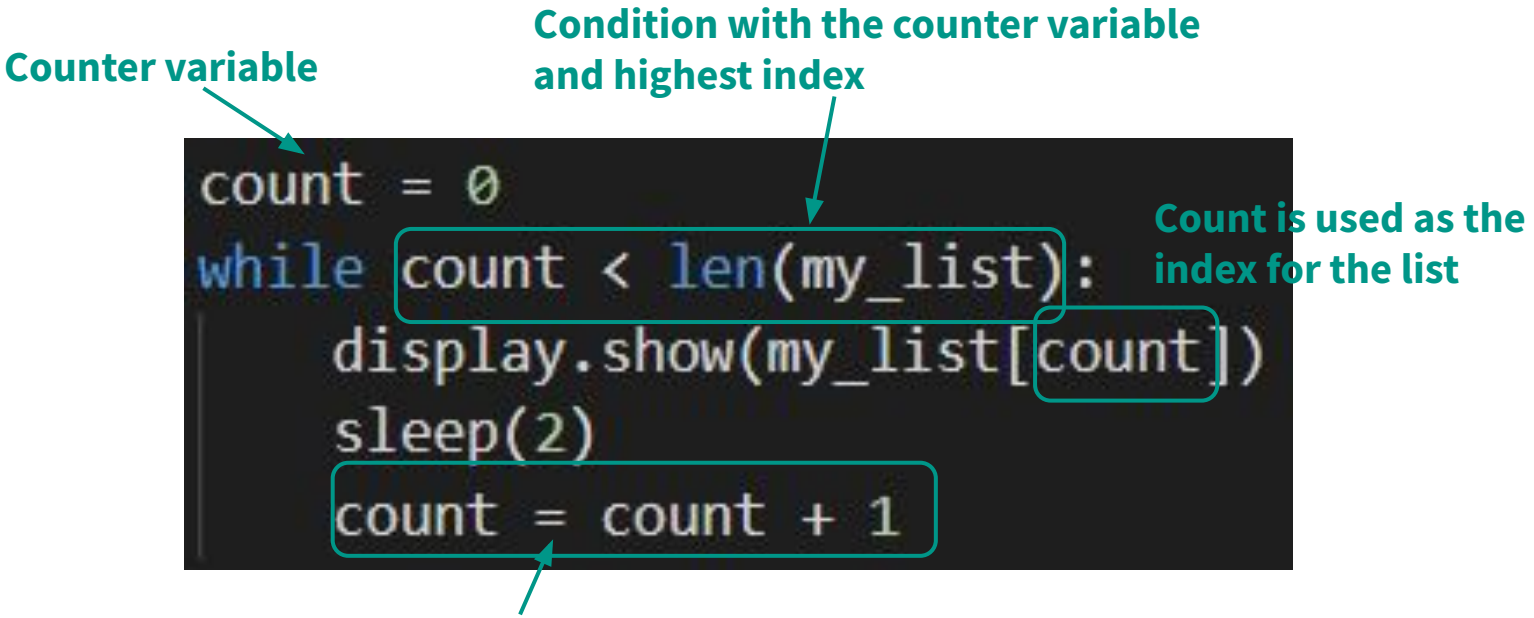
Here is what it looks like with a **while loop**:

```
count = 0
while count < len(my_list):
    display.show(my_list[count])
    sleep(2)
    count = count + 1
```

FIRIA LABS

# Parts of the loop:

When traversing a list, you must include these parts:



# For loop:

With this kind of problem, you always know exactly how many times the loop will execute – the length of the list.

So … there is a shorter way – The for loop!

A **for loop** embeds the counter and incrementing the counter in its design.

# For loop:

These two loops do EXACTLY the same thing. Compare the loops and where the different parts of the loop can be found.

```
count = 0
while count < len(my_list):
    display.show(my_list[count])
    sleep(2)
    count = count + 1
```

```
for count in range(len(my_list)):
    display.show(my_list[count])
    sleep(2)
```

In the for loop, the count variable will always start at 0 and go until it equals the upper limit. The increment is built in.

FIRIA LABS

# While loop and For loop:

The counter doesn't have to be called "count". It can be anything.

Explain what each of these loops is doing, line by line:

```
index = 0
while index < len(my_list):
    the_image = my_list[index]
    if type(the_image) == tuple:
        display.fill(the_image)
    else:
        display.show(the_image)
    sleep(2)
    index = index + 1
```

```
for index in range(len(my_list)):
    the_image = my_list[index]
    if type(the_image) == tuple:
        display.fill(the_image)
    else:
        display.show(the_image)
    sleep(2)
```

FIRIA LABS

# Traversing a list:

Write a for loop that will traverse this list and turn pixel 0 the designated color for 1 second:

```
colors = [WHITE, YELLOW, ORANGE, PINK, CYAN, MAGENTA]
pixels.set(0, colors[index])
sleep(1.0)
```

# Traversing a list:

Write a for loop that will traverse this list and print each message with a 1 second delay:

```
instructions = ["These are instructions",
                "to tell a person",
                "what to do when",
                "running some code.",
                "Then press BUTTON A"]
```

FIRIA LABS

# Traversing a list:

Loops can be used for accessing items in several lists at the same time. Let's say you create a function to move a robot. You can call the function every time you want the robot to move, like the code on the left. Or you can create three lists with all the data and traverse the lists. **Write the while loop on the right as a for loop.**

```
def move(left, right, tm):
    motors.run(LEFT, left)
    motors.run(RIGHT, right)
    sleep(tm)

move(60, 60, 3.0)
move(60, -60, 1.0)
move(40, 40, 3.0)
move(20, 50, 1.0)
move(60, 60, 2.5)
```

```
def move(left, right, tm):
    motors.run(LEFT, left)
    motors.run(RIGHT, right)
    sleep(tm)

left_moves = [60, 60, 40, 20, 60]
right_moves = [60, -60, 40, 50, 60]
times = [3.0, 1.0, 3.0, 1.0, 2.5]
index = 0
while index < len(times):
    move(left_moves[index], right_moves[index], times[index])
    index = index + 1
```

FIRIA LABS

# Traversing a list:

This code uses the index in two ways in the loop. Change the while loop to a for loop:

```
colors = [GREEN, BLUE, RED, YELLOW]
index = 0
while index < len(colors):
    pixels.set(index, colors[index])
    index = index + 1
```

FIRIA LABS

# Traversing a list:

Write a for loop that will traverse the two lists:

```
right_side = [RED, BLUE, GREEN, PURPLE, GRAY, BROWN]
left_side = [WHITE, YELLOW, ORANGE, PINK, CYAN, MAGENTA]
pixels.set(0, left_side[index])
pixels.set(1, left_side[index])
pixels.set(2, right_side[index])
pixels.set(3, right_side[index])
sleep(2.0)
```

FIRIA LABS

# For loops and lists:

Traversing a list always follows the same structure. If you are traversing only one list, we can simplify even more. Compare these two for loops. They do EXACTLY the same thing:

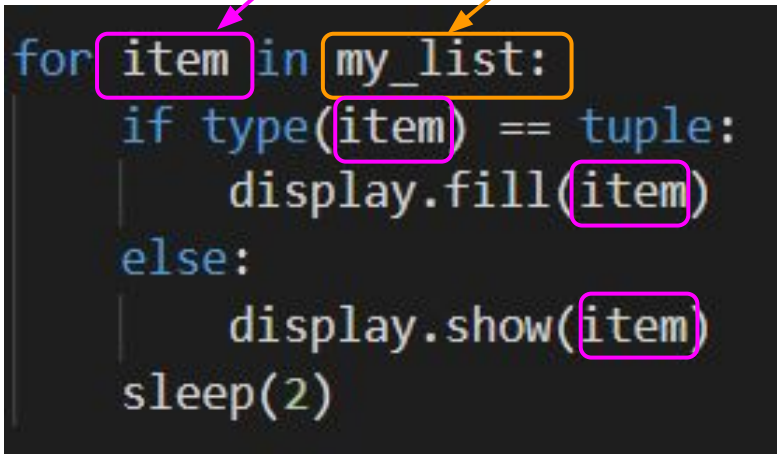
```
for index in range(len(my_list)):
    the_image = my_list[index]
    if type(the_image) == tuple:
        display.fill(the_image)
    else:
        display.show(the_image)
    sleep(2)
```

```
for item in my_list:
    if type(item) == tuple:
        display.fill(item)
    else:
        display.show(item)
    sleep(2)
```

FIRIA LABS

# For loops and lists:

In this **specialized for loop**, instead of a counter or index variable, you will simply reference each item in the list. The only drawback is that you do not know the index of each item.



# Traversing a list:

Change this for loop to a specialized for loop:

```
for count in range(len(my_list)):
    display.show(my_list[count])
    sleep(2)
```

FIRIA LABS

# Traversing a list:

Change this for loop to a specialized for loop:

```
for index in range(len(my_list)):
    the_image = my_list[index]
    if type(the_image) == tuple:
        display.fill(the_image)
    else:
        display.show(the_image)
    sleep(2)
```

FIRIA LABS

# Traversing a list:

Change this for loop you wrote for this code to a specialized for loop:

```
colors = [WHITE, YELLOW, ORANGE, PINK, CYAN, MAGENTA]
pixels.set(0, colors)
sleep(1.0)
```

FIRIA LABS

# Traversing a list:

Change this for loop you wrote for this code to a specialized for loop:

```
instructions = ["These are instructions",
                "to tell a person",
                "what to do when",
                "running some code.",
                "Then press BUTTON A"]
```

FIRIA LABS

# Wrap-up

Abstraction and Functions

# Traversing a list

- Now you have had some experience with traversing a list three ways:
  - While loop
  - For loop
  - Specialized for loop
- Open your assignment document and answer the questions in the reflection.

FIRIA LABS