Traversing a List

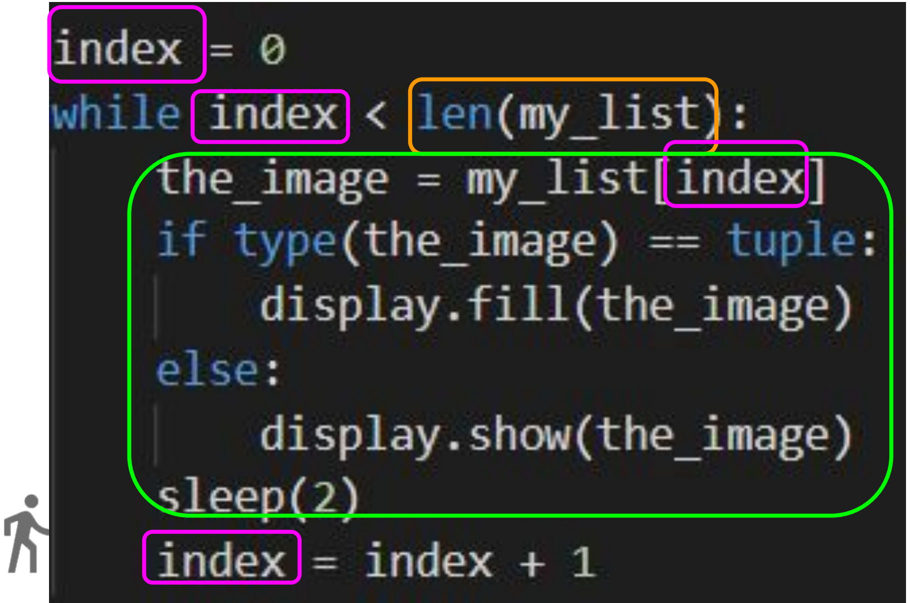
An supplemental lesson after Mission 9

FIRIA LABS

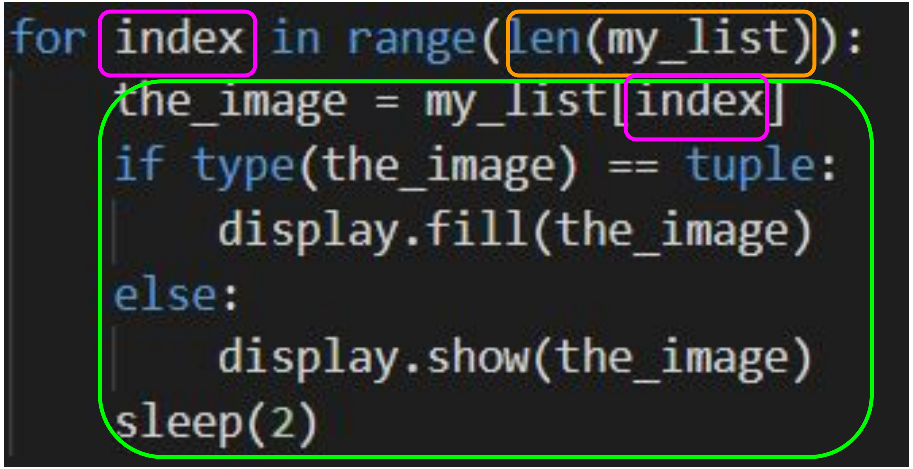
While loop and For loop: (Question)

The counter doesn’t have to be called “count”. It can be anything.

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

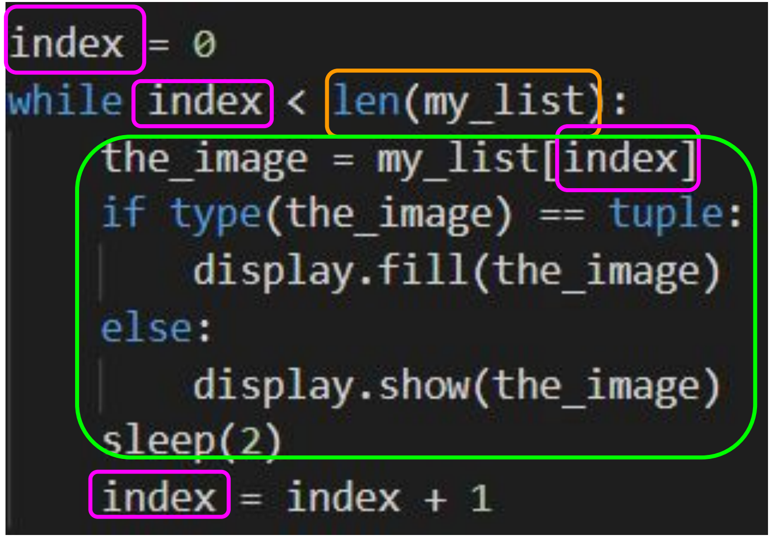


```
index = 0
while index < len(my_list):
    the_image = my_list[index]
    if type(the_image) == tuple:
        display.fill(the_image)
    else:
        display.show(the_image)
    sleep(2)
    index = index + 1
```



```
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)
```

While loop and For loop: (Answer)



```
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
```

First line: index is initialized to 0

Second line: while loop is started; the condition is index < len(my_list), so it will start at 0 and go through each element

Inside the loop:

A variable is assigned an element from the my_list (in order)

An if statement checks the type of the element.

It will either fill the display with a color or show an image for 2 seconds.

The index variable will increment

While loop and For loop: (Answer)

First line: the for loop is started using index as the counter variable. It will start at 0 and go through each element in my_list.

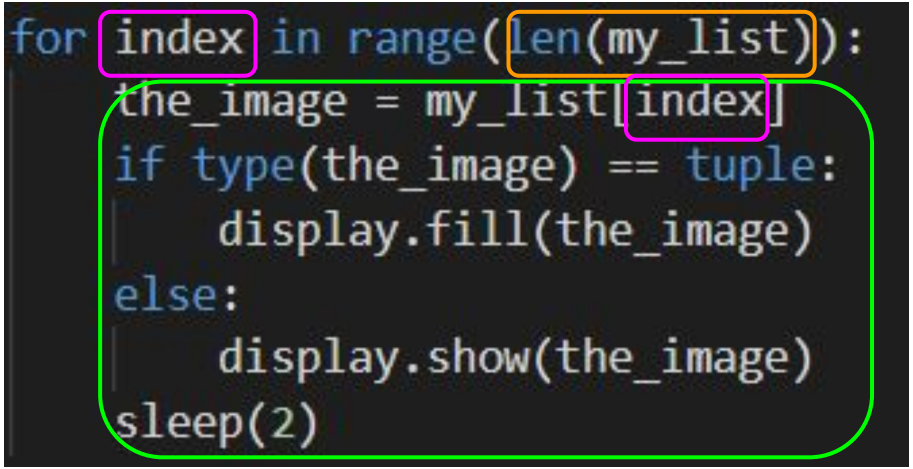
Inside the loop:

A variable is assigned an element from the my_list (in order)

An if statement checks the type of the element.

It will either fill the display with a color or show an image for 2 seconds.

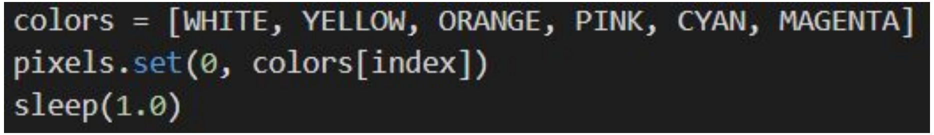
When the loop continues, the index will be incremented.



```
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)
```

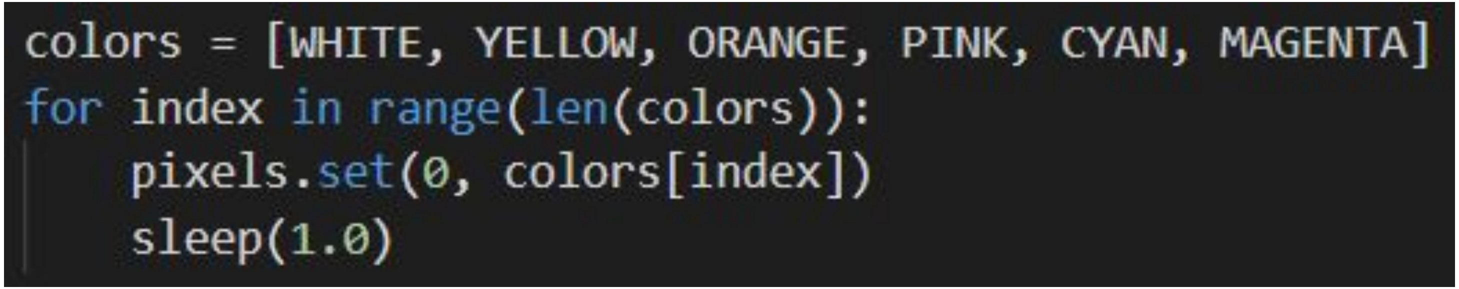
Traversing a list:

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)
```

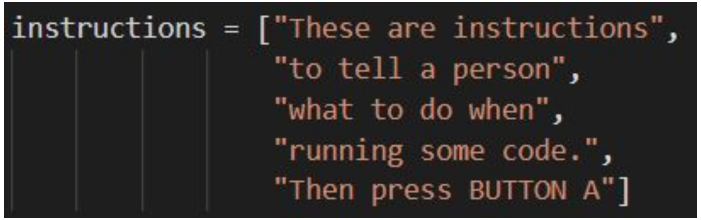
Answer:



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

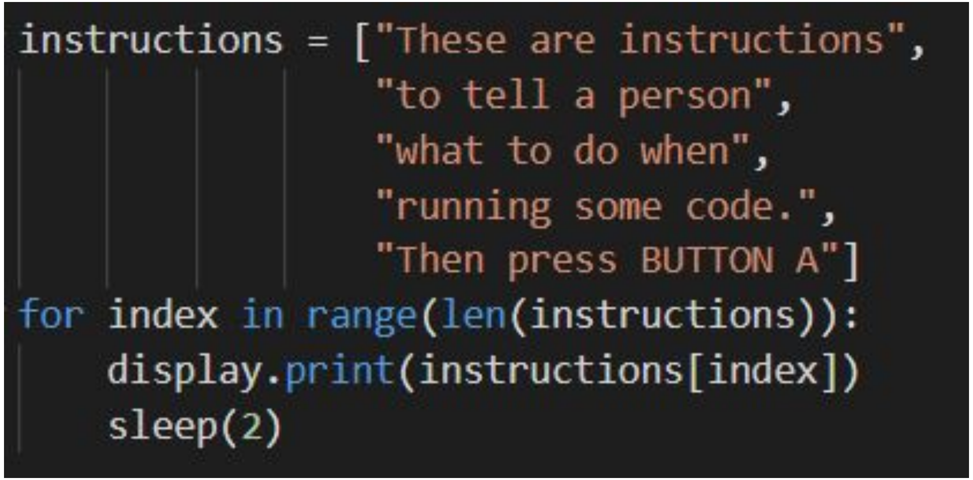
Traversing a list:

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"]
```

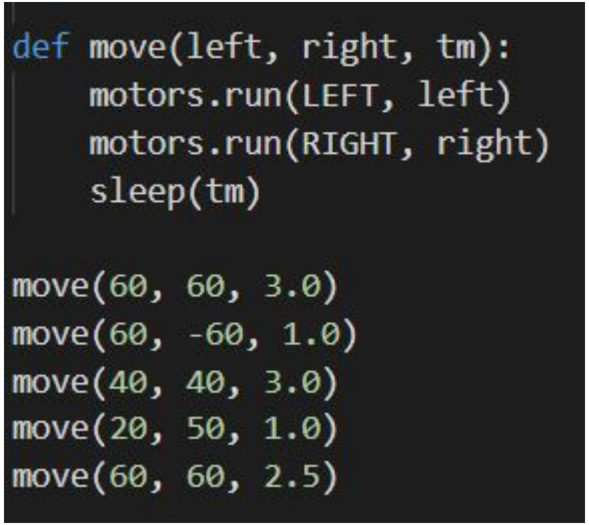
Answer:



```
instructions = ["These are instructions",
                "to tell a person",
                "what to do when",
                "running some code.",
                "Then press BUTTON A"]
for index in range(len(instructions)):
    display.print(instructions[index])
    sleep(2)
```

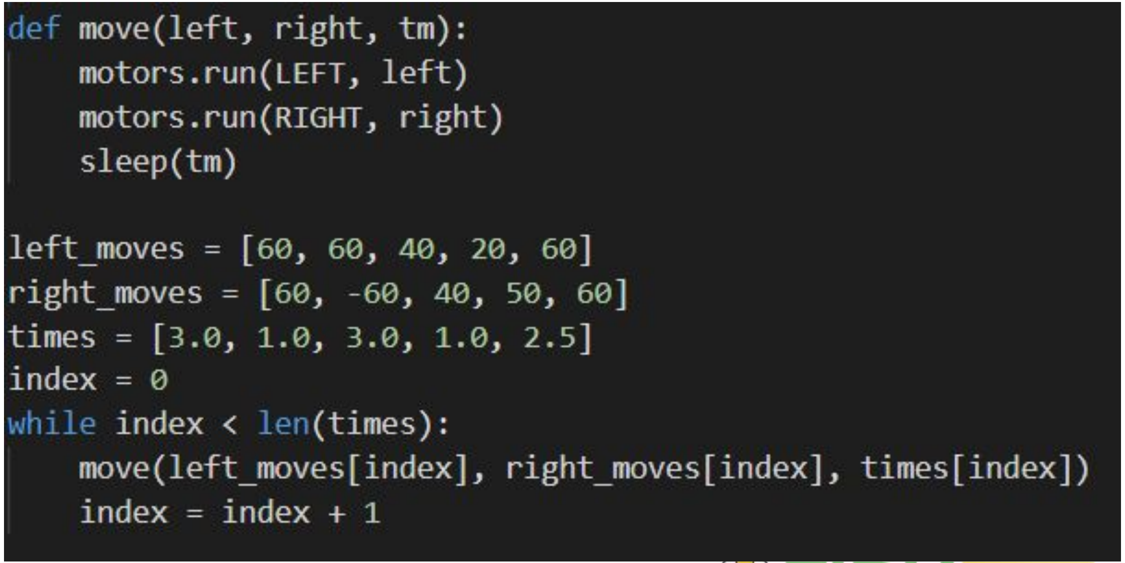
Traversing a list: Question

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)
```

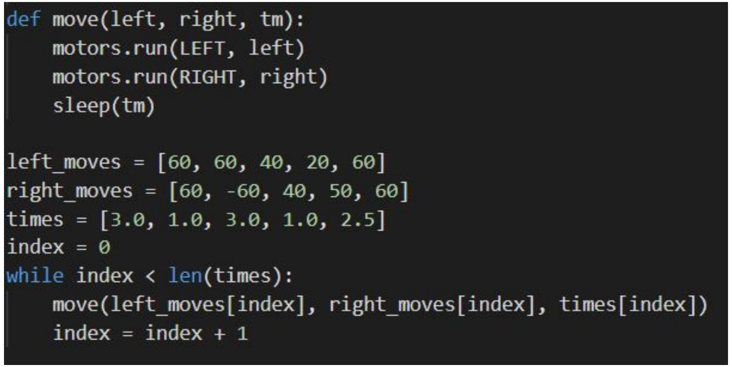


```
def move(left, right, tm):
    motors.run(LEFT, left)
    motors.run(RIGHT, right)
    sleep(tm)

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
```

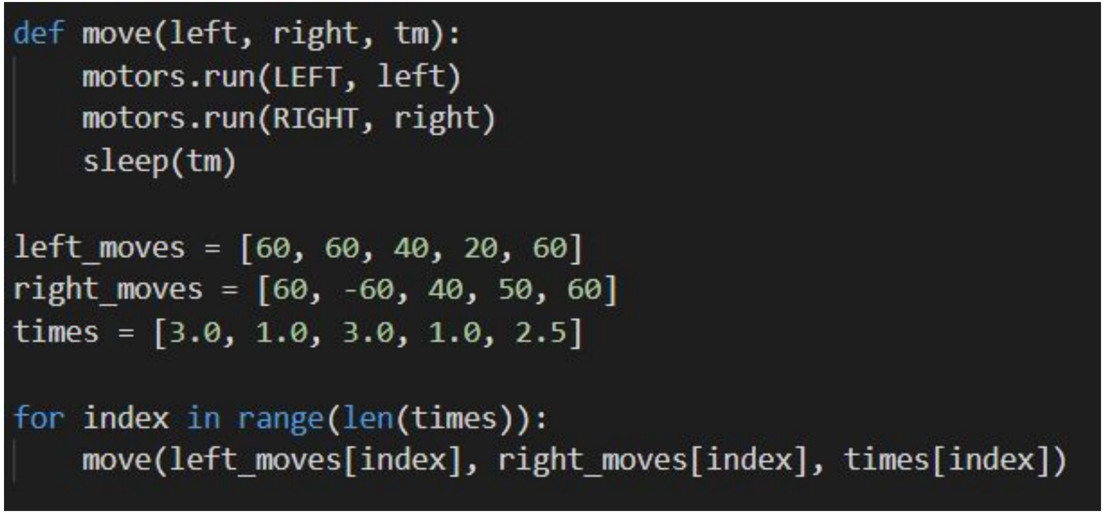
Traversing a list: Answer

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)

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
```



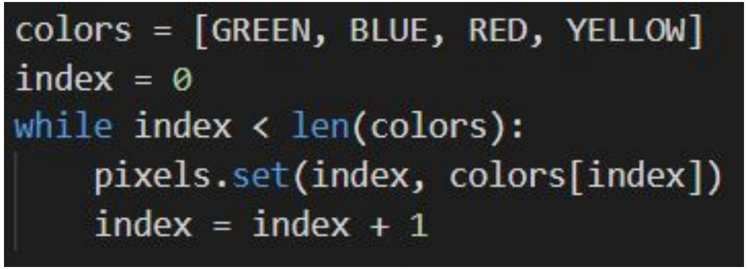
```
def move(left, right, tm):
    motors.run(LEFT, left)
    motors.run(RIGHT, right)
    sleep(tm)

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]

for index in range(len(times)):
    move(left_moves[index], right_moves[index], times[index])
```

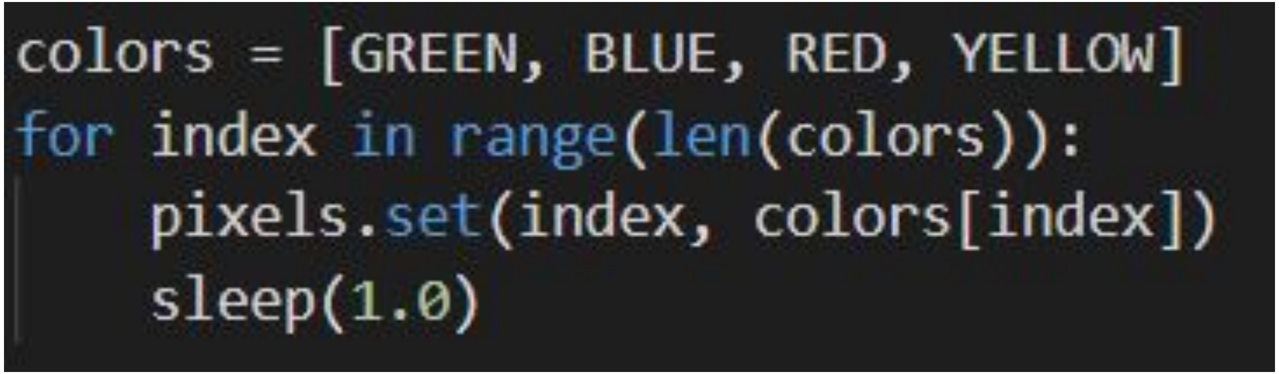
Traversing a list:

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
```

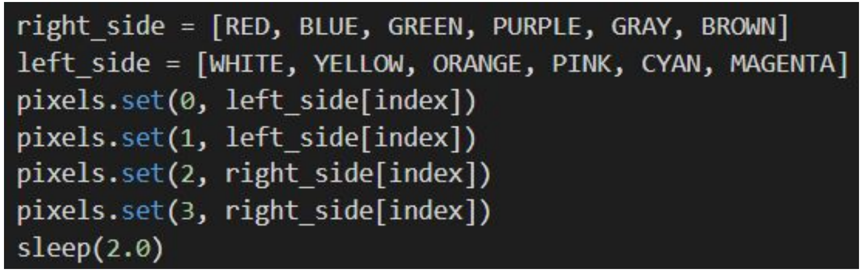
Answer:



```
colors = [GREEN, BLUE, RED, YELLOW]
for index in range(len(colors)):
    pixels.set(index, colors[index])
    sleep(1.0)
```

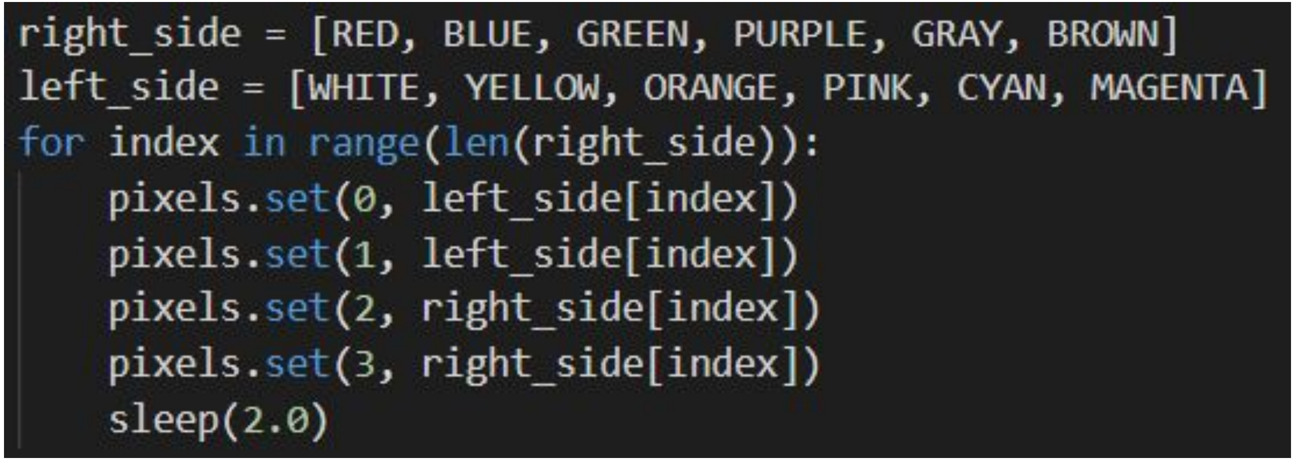
Traversing a list:

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)
```

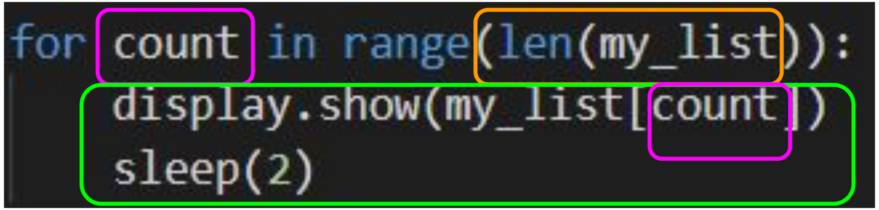
Answer:



```
right_side = [RED, BLUE, GREEN, PURPLE, GRAY, BROWN]
left_side = [WHITE, YELLOW, ORANGE, PINK, CYAN, MAGENTA]
for index in range(len(right_side)):
    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)
```

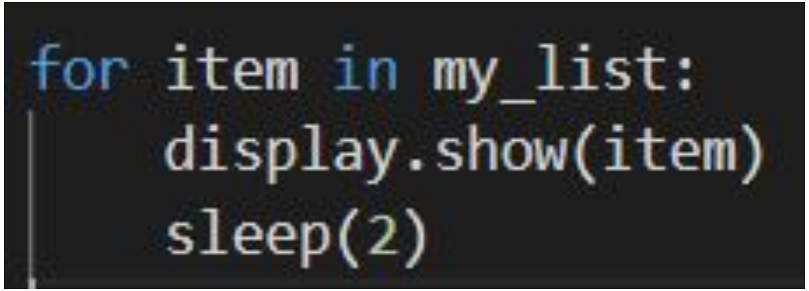
Traversing a list:

Change this for loop to a specialized for loop:



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

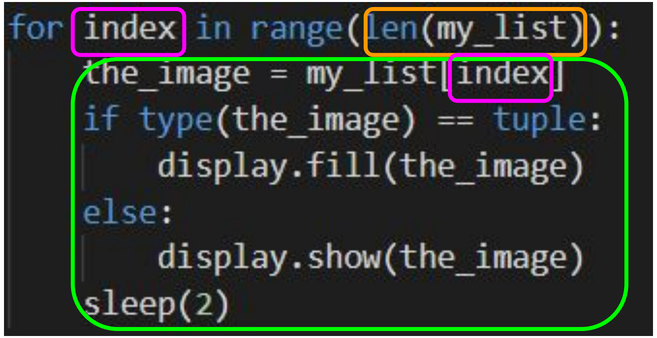
Answer:



```
for item in my_list:
    display.show(item)
    sleep(2)
```

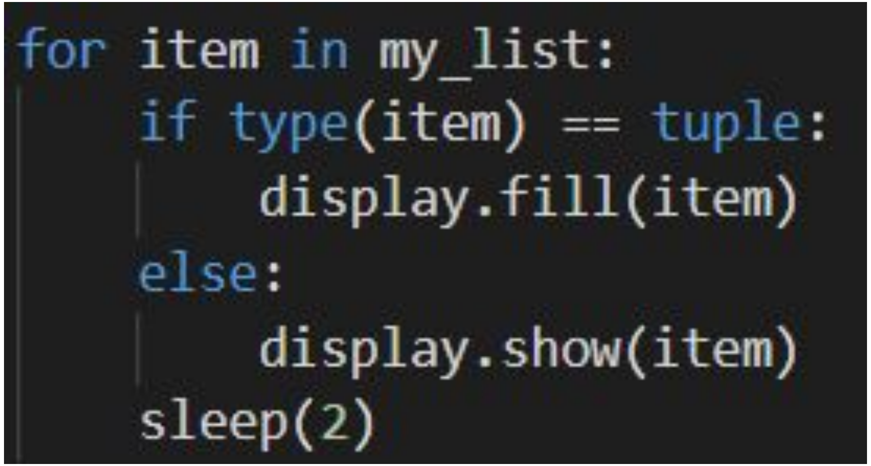
Traversing a list:

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)
```

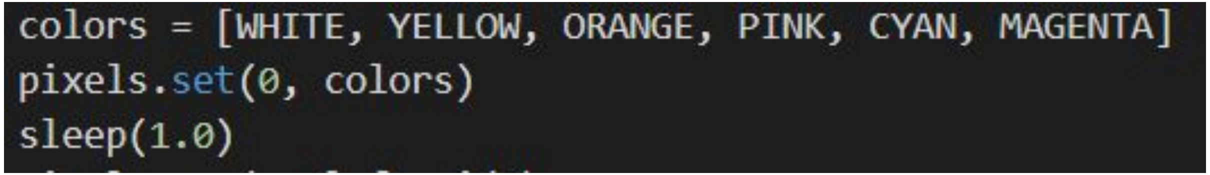
Answer:



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

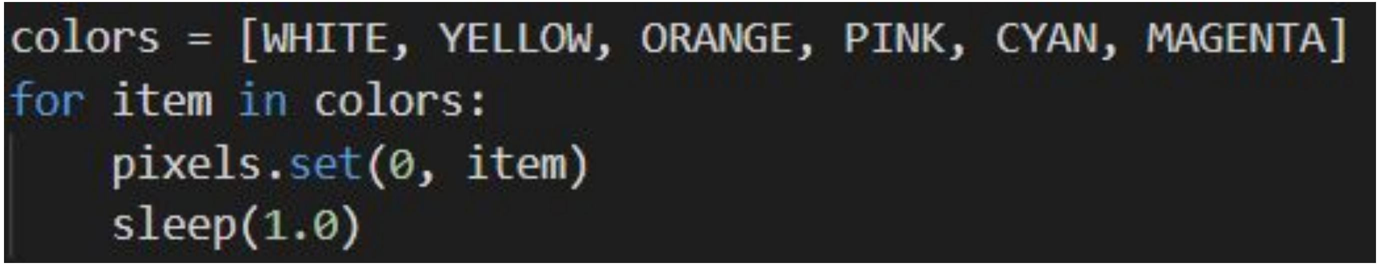
Traversing a list:

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)
```

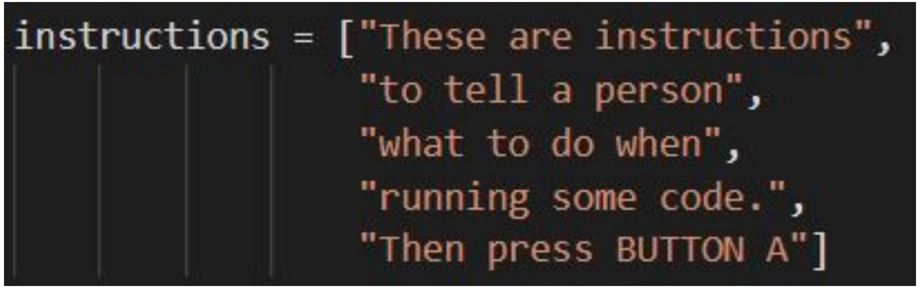
Answer:



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

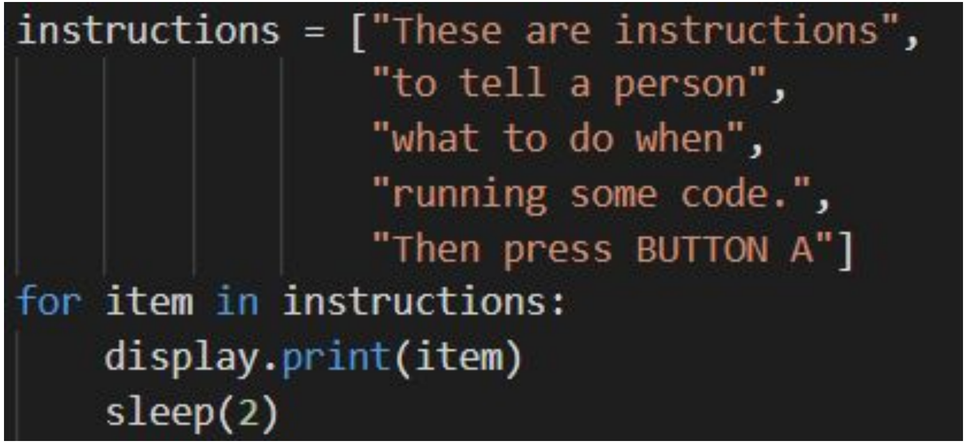
Traversing a list:

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"]
```

Answer:



```
instructions = ["These are instructions",
                "to tell a person",
                "what to do when",
                "running some code.",
                "Then press BUTTON A"]
for item in instructions:
    display.print(item)
    sleep(2)
```