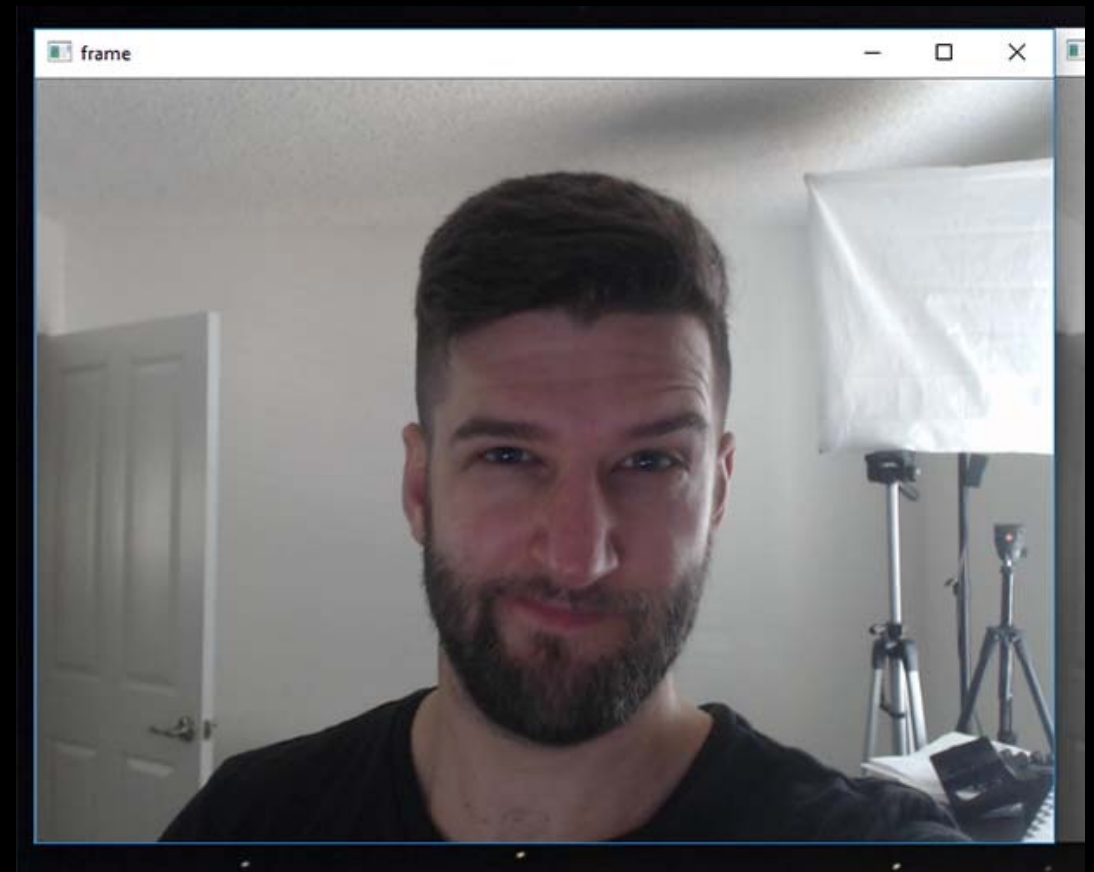
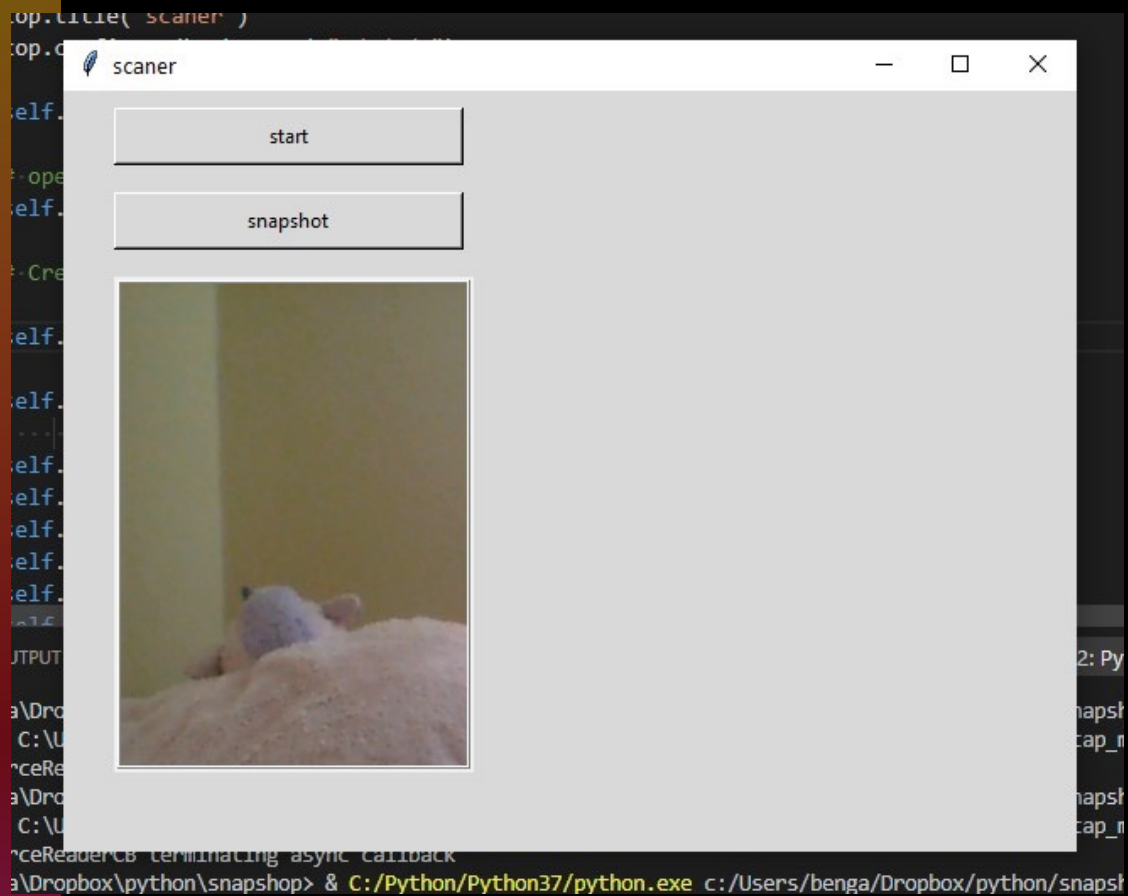


# **Object Detection Video Streaming on Web**

Mì AI

# Problems




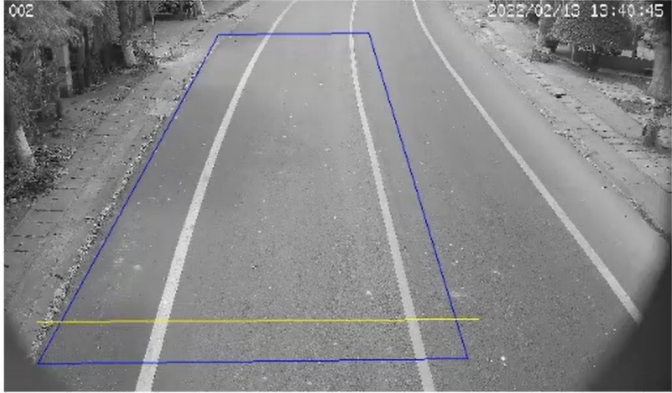
# Problems

Camera 1 Camera 2

TRANSLOC

TRG Traffic Counting

002 2022/02/19 13:40:45



License plate: 51H39418

CAR	TRUCK	BUS	TOTAL
1425	2503	0	3928

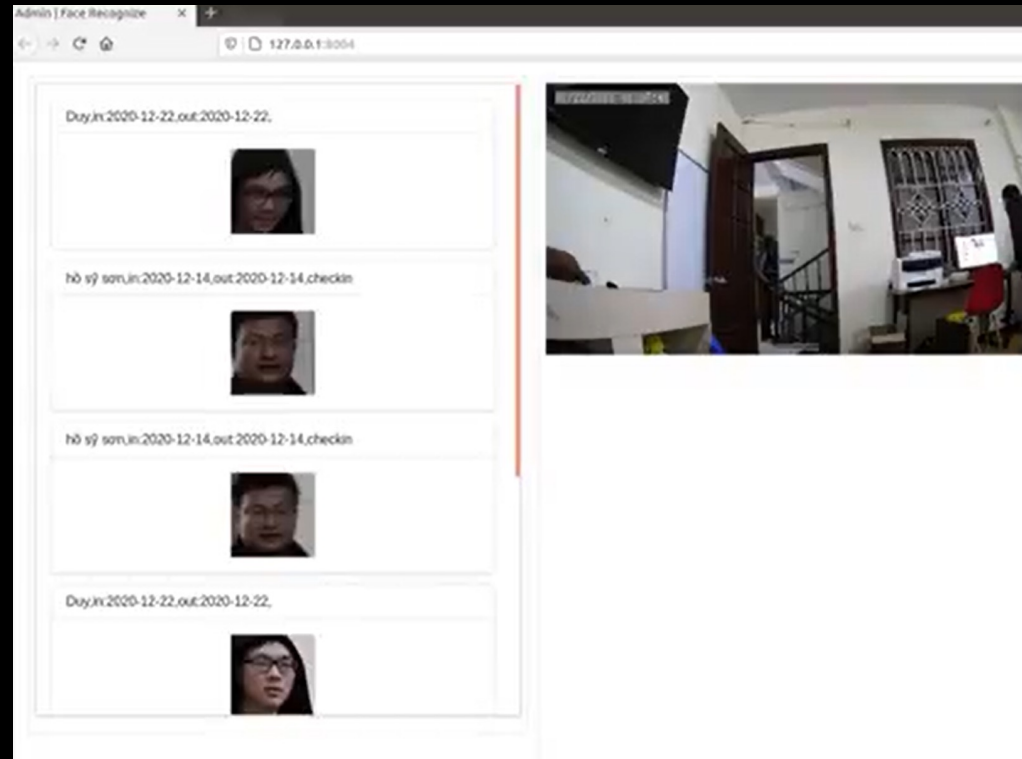
☒ Car, Truck, Bus ☐ Others ☒ auto refresh

mm/dd/yyyy

Export Download file

<https://www.facebook.com/sx.bk.it/videos/362390332396020>

# Problems

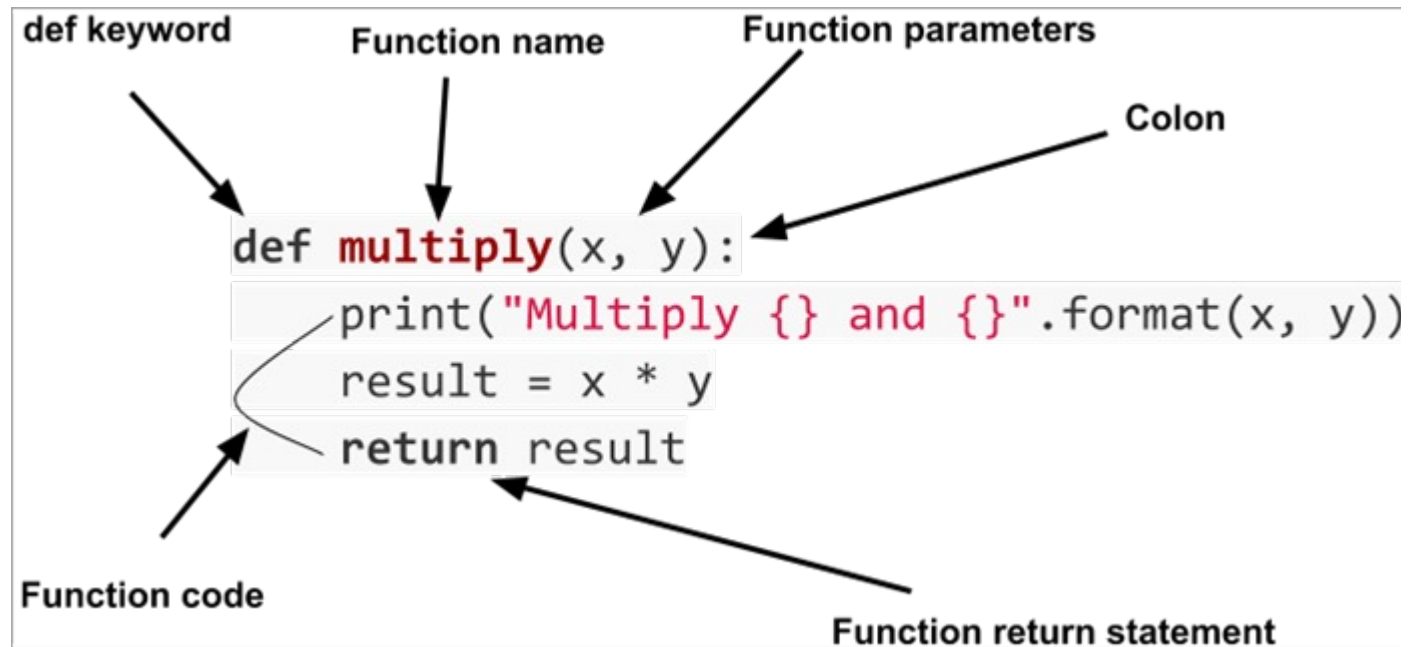


<https://www.facebook.com/sx.bk.it/videos/2744699025779393>

# Problems



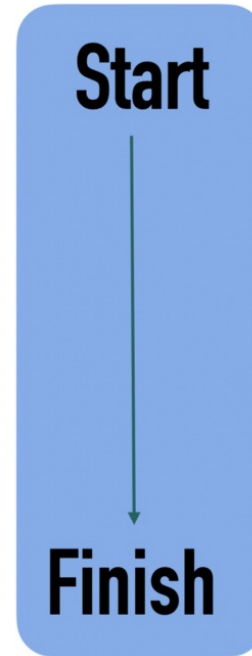
# Function return




The diagram shows a Python function definition with several labels and arrows pointing to specific parts of the code:

- def keyword**: Points to the `def` keyword.
- Function name**: Points to the word `multiply`.
- Function parameters**: Points to the parentheses and variables `(x, y)`.
- Colon**: Points to the colon `:` at the end of the first line.
- Function code**: A bracket on the left side of the function body points to the three lines of code inside.
- Function return statement**: Points to the `return result` line.

```
def multiply(x, y):  
    print("Multiply {} and {}".format(x, y))  
    result = x * y  
    return result
```



**Functions**



**But...video streaming need return  
multiple times**

# Yield statement

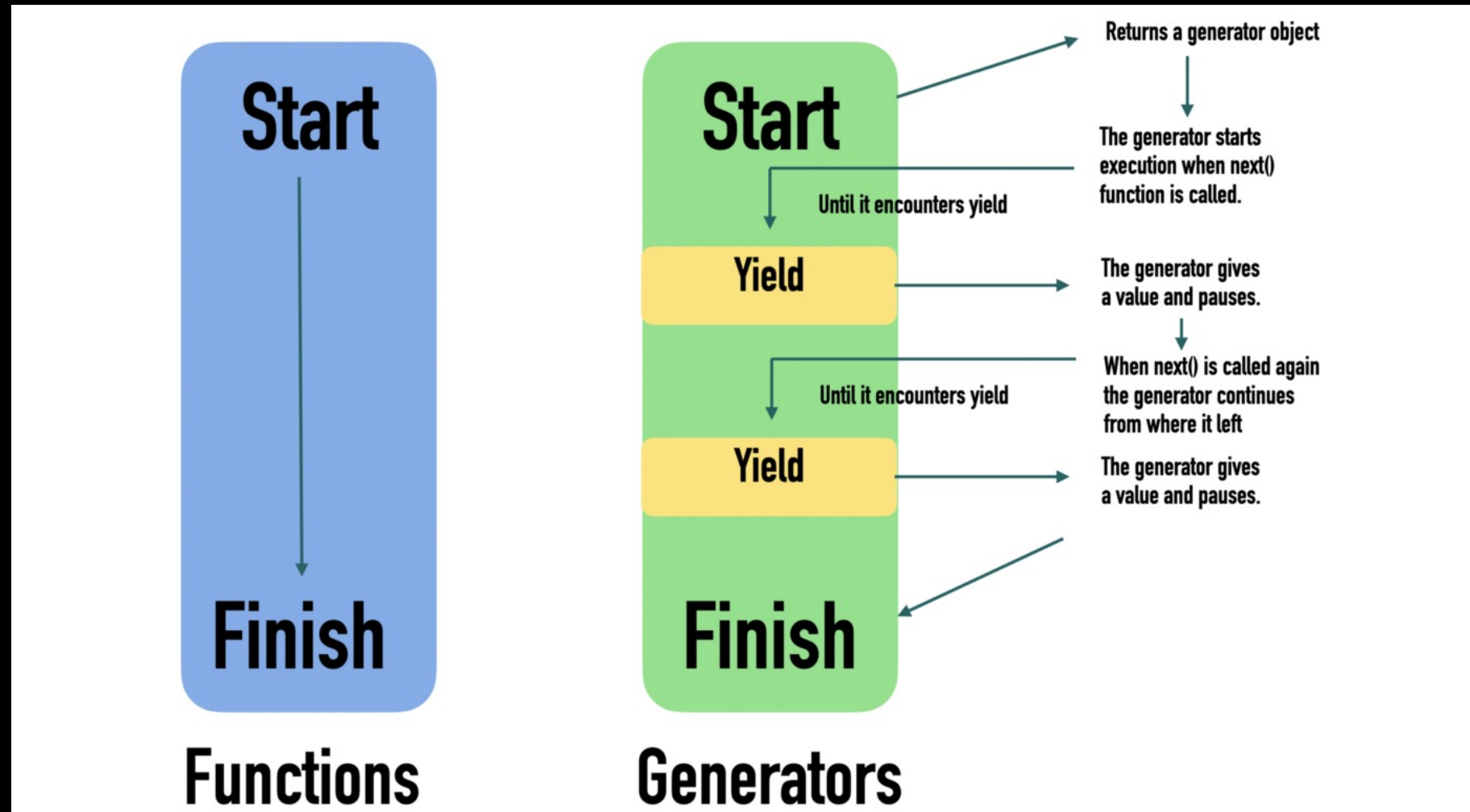
The yield statement suspends a function's execution and sends a value back to the caller, but retains enough state to enable the function to resume where it left off.

When the function resumes, it continues execution immediately after the last yield run.

This allows its code to produce a series of values over time, rather than computing them at once and sending them back like a list.



# Yield statement



# Yield statement

Python

```
# A Simple Python program to demonstrate working
# of yield

# A generator function that yields 1 for the first time,
# 2 second time and 3 third time

def simpleGeneratorFun():
    yield 1
    yield 2
    yield 3

# Driver code to check above generator function
for value in simpleGeneratorFun():
    print(value)
```

# Pipeline

