**Producer/Consumer Problem (also known as the ‘bounded buffer’ problem)**:

from threading import Thread

from queue import Queue

q = Queue()

final\_results = []

def producer():

for i in range(100):

q.put(i)

def consumer():

while True:

number = q.get()

result = (number, number\*\*2)

final\_results.append(result)

q.task\_done()

for i in range(5):

t = Thread(target=consumer)

t.daemon = True

t.start()

producer()

q.join()

print (final\_results)

In the above code, the queue data structure is used to demonstrate how multithreading works. In simple terms, the producer function generates numbers from 1 to 99. Once a number is placed in a queue, the consumer immediately processes it on one of the available threads.

The purpose of q.put(i) is to place the number in queue, whereas q.get() method returns that number. At the end q.join() method blocks the execution of the program until all tasks in a queue are not completed.