**Codio Activity: The Producer-Consumer Mechanism**

**Task**

Run producer-consumer.py in the provided Codio workspace (**Producer-Consumer Mechanism**), where the queue data structure is used.

A copy of the code is available here for you.

# code source: https://techmonger.github.io/55/producer-consumer-python/

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)

Answer the following questions:

1. How is the queue data structure used to achieve the purpose of the code?
2. What is the purpose of q.put(I)?
3. What is achieved by q.get()?
4. What functionality is provided by q.join()?

**Answers:**

* 1. The queue data structure is used as a buffer. The producer queues numbers into it and the consumer gets a number from the queue. The producer is running on the main thread where the consumer is running on five different background threads. You can tell they are running as a background thread by reading the daemon keyword.
  2. The put keyword means that we are appending or adding a number into the queue.
  3. The get keyword means that we are retrieving an item from the queue and by retrieving it we also remove it from the queue.
  4. The join function prevents the program from exiting. The program will exit once all five background threads are done. The background threads are done when the task\_done function is called by the consumer.