Please write some code for this problem. Any kind of pseudocode is fine, as long as its meaning is clear. You may assume common primitives such as hash tables and linked lists: you do not need to implement them.

Write a concurrent program to issue and service disk requests. There are N requester threads to issue disk requests and one thread to service disk requests. Write procedures for the requester threads and service thread.

Each requester thread issues a series of requests for disk tracks: you may use any pseudocode for the requester to decide on its next request. Threads issue disk requests by queueing them at the disk scheduler. The disk scheduler queue can contain at most MAX requests: threads must wait if the queue is full.

Each request is synchronous; once a requester thread has issued a request, it must wait until the servicing thread finishes handling its last request before issuing its next request.

The service thread handles disk requests only when the queue has the “largest number of requests”. When at least MAX requester threads are alive, the “largest number of requests” is MAX. When fewer than MAX requester threads are alive, the largest number of requests in the queue is equal to the number of living requester threads.

You may use any pseudocode to select which request to service next in the service thread.