Develop a threaded port scanner to check if particular ports are open or closed for a remote host. Determine which Port scanner is efficient

->here this method is efficient due to multitasking

1. importing moudles

```
import threading
import socket
from queue import Queue
import time
```

2. input the target and ports

```
target·=·input("Enter·the·Target")
low=int(input('Enter·the·lower_bound'))
high=int(input("Enter·the·higher_bound"))
```

+ Code + Text

3. creating the function that is used to scan ports

```
print_lock = threading.Lock()
# a print_lock is what is used to prevent "double" modification of shared variables.
def portscan(port):
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    try:
        con = s.connect((target, port))
        with print_lock:
            print('port', port)
        con.close()
    except:
        pass
```

4 .creating the threader function which takes port from queue and assign to thread

Double-click (or enter) to edit

```
def threader():
    while True:
        # gets an port from the queue
        port = q.get()
```

```
        # Run the example job with the avail port in queue (thread)
        portscan(port)
        # completed with the job
        q.task_done()
```

5. creating the threads

```
# Create the queue and threader
q = Queue()
no_thread=int(input("Enter·the·number·of·thereads"))
# how many threads are we going to allow for
for x in range(no_thread):
    t = threading.Thread(target=threader)
    # classifying as a daemon, so they will die when the main dies
    t.daemon = True
    # begins, must come after daemon definition
    t.start()
```

3. Scanning the ports

```
for port in range(low, high):
    q.put(port)

# wait until the thread terminates.
q.join()
```