1. Timing of incorrect incrementing Thread code:

Enter maximum number of threads: 4
Enter number of times all threads should increment: 10000000
#Threads 1 : Counter should be 10000000 but the counter = [10000000] with a time of 6.505928 sec.
#Threads 2 : Counter should be 10000000 but the counter = [7535587] with a time of 4.518007 sec.
#Threads 3 : Counter should be 10000000 but the counter = [5683160] with a time of 3.918050 sec.
#Threads 4 : Counter should be 10000000 but the counter = [4348169] with a time of 3.620550 sec.
Hit <Enter> to quit

Timing of correct incrementing Thread code which uses a Condition object to force mutually exclusive access to the counter:

Enter maximum number of threads: 4
Enter number of times all threads should increment: 10000000
#Threads 1 : Counter should be 10000000 but the counter = [10000000] with a time of 117.962377 sec.
#Threads 2 : Counter should be 10000000 but the counter = [10000000] with a time of 134.423648 sec.
#Threads 3 : Counter should be 10000000 but the counter = [10000000] with a time of 130.316724 sec.
#Threads 4 : Counter should be 10000000 but the counter = [10000000] with a time of 134.391264 sec.
Hit <Enter> to quit

a) Why is the correct code so much slower?

2. Consider the client/server code from section 10.2 for the time and day.

```python
# server.py  Server for providing day and time. 
from socket import *
from time import ctime

HOST = 'localhost'
PORT = 5000
ADDRESS = (HOST, PORT)

server = socket(AF_INET, SOCK_STREAM)   # Create a socket
server.bind(ADDRESS)                    # Bind server to host and port
server.listen(5)                        # Listen for up to 5 requests

while True:
    print 'Waiting for a connection...
    client, address = server.accept()  # Accept connection from a client
    print '... connected from:',address
    client.send(ctime()+'\nHave a nice day!')   # Send day and time
    client.close()                              # Close the connection

temp = raw_input("Hit <Enter> key to quit")
```

```python
# client.py Client for obtaining day and time. 
from socket import *

HOST = 'localhost'
PORT = 5000
BUFSIZE = 1024
ADDRESS = (HOST, PORT)

server = socket(AF_INET, SOCK_STREAM)   # Create a socket
server.connect(ADDRESS)                 # Connect it to a host

dayAndTime = server.recv(BUFSIZE)       # Read a string from it
print dayAndTime
server.close()                              # Close the connection

temp = raw_input("Hit <Enter> key to quit")
```
a) Explain the role that ports and IP addresses play in a client/server program.

b) What is a local host, and how is it used to develop networked applications?

c) If the service being provided by the server takes a long time, then why is it a good idea for a server to create threads to handle clients’ requests as in serverWithClientHandler.py?

```python
# serverWithClientHandler.py   Server with Client Handler for providing day and time.

from socket import *
from time import ctime
from threading import Thread

class ClientHandler(Thread):
    """ Handles a client request. """
    def __init__(self, client):
        Thread.__init__(self)
        self._client = client

    def run(self):
        self._client.send(ctime() + '\nHave a nice day!')
        client.close()  # Close the connection

HOST = 'localhost'
PORT = 5000
ADDRESS = (HOST, PORT)

server = socket(AF_INET, SOCK_STREAM)  # Create a socket
server.bind(ADDRESS)  # Bind server to host and port
server.listen(5)  # Listen for up to 5 requests

while True:
    print 'Waiting for a connection...'
    client, address = server.accept()  # Accept connection from a client
    print '... connected from:', address
    handler = ClientHandler(client)
    handler.start()
```