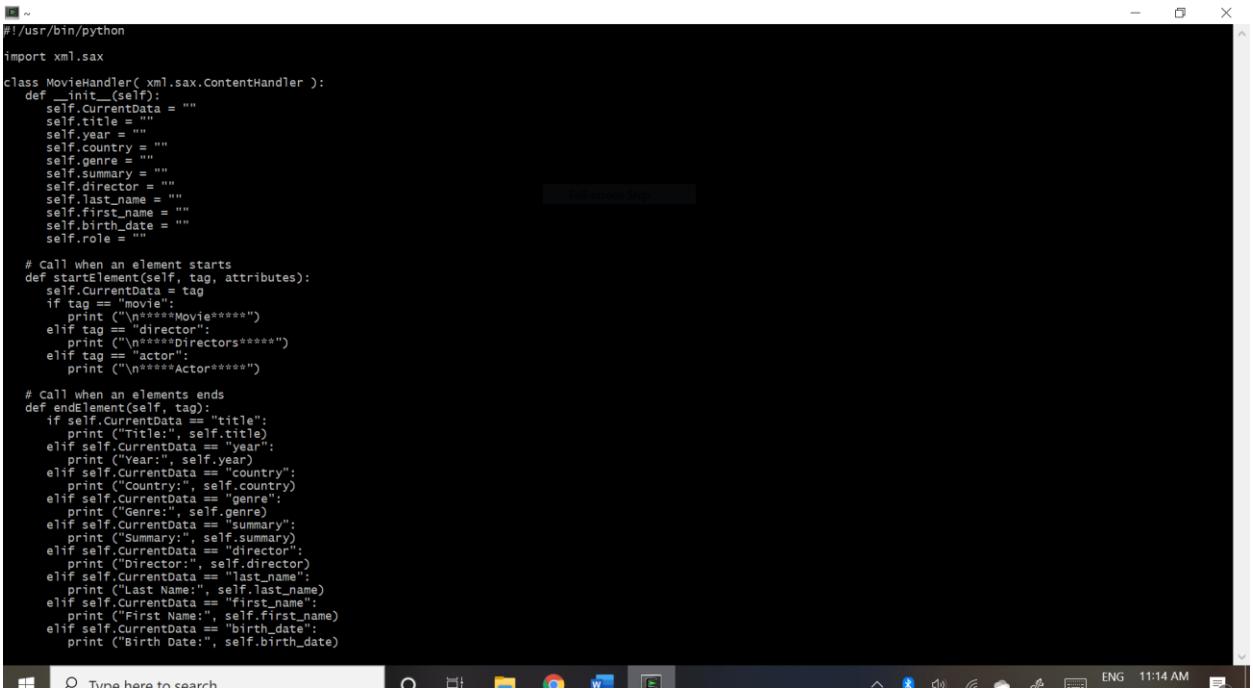
## Q2 ==> Parsing XML with SAX APIs

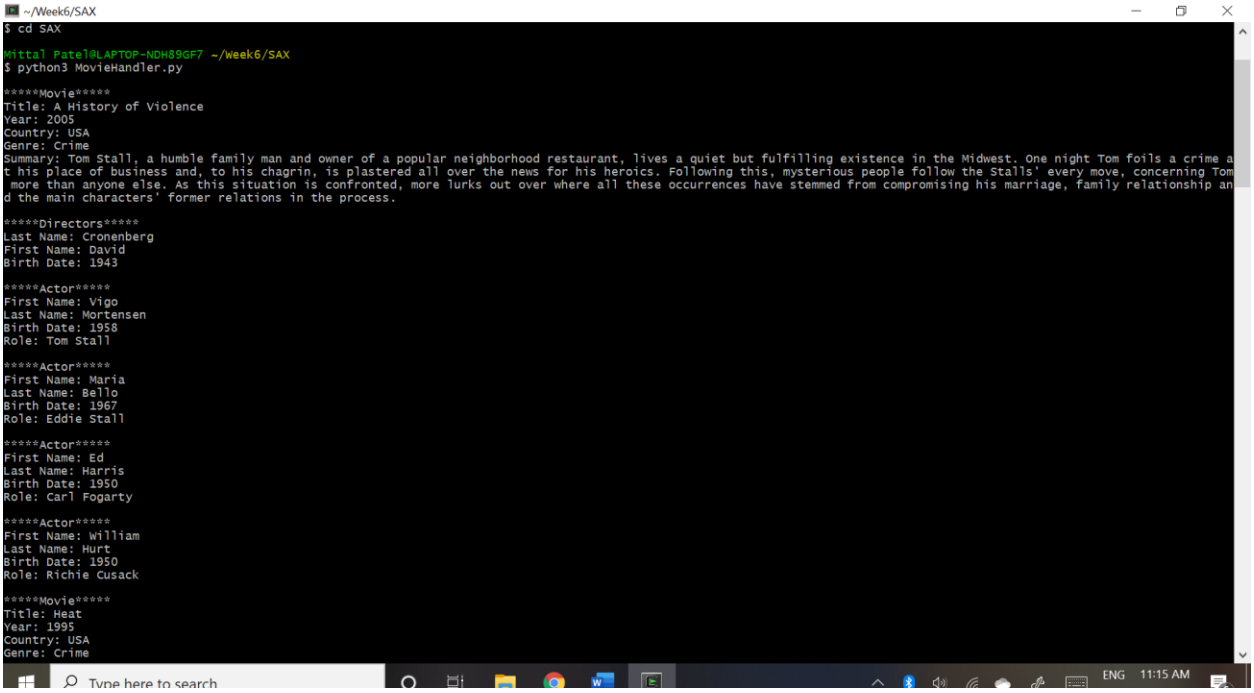
### MovieHandler.py

```
#!/usr/bin/python

import xml.sax

class MovieHandler( xml.sax.ContentHandler ):
  def __init__(self):
    self.CurrentData = ""
    self.title = ""
    self.year = ""
    self.country = ""
    self.genre = ""
    self.summary = ""
    self.director = ""
    self.last_name = ""
    self.first_name = ""
    self.birth_date = ""
    self.role = ""

  # Call when an element starts
  def startElement(self, tag, attributes):
    self.CurrentData = tag
    if tag == "movie":
      print ("\n*****Movie*****")
    elif tag == "director":
      print ("\n*****Directors*****")
    elif tag == "actor":
      print ("\n*****Actor*****")

  # Call when an elements ends
  def endElement(self, tag):
    if self.CurrentData == "title":
      print ("Title:", self.title)
    elif self.CurrentData == "year":
      print ("Year:", self.year)
    elif self.CurrentData == "country":
      print ("Country:", self.country)
```

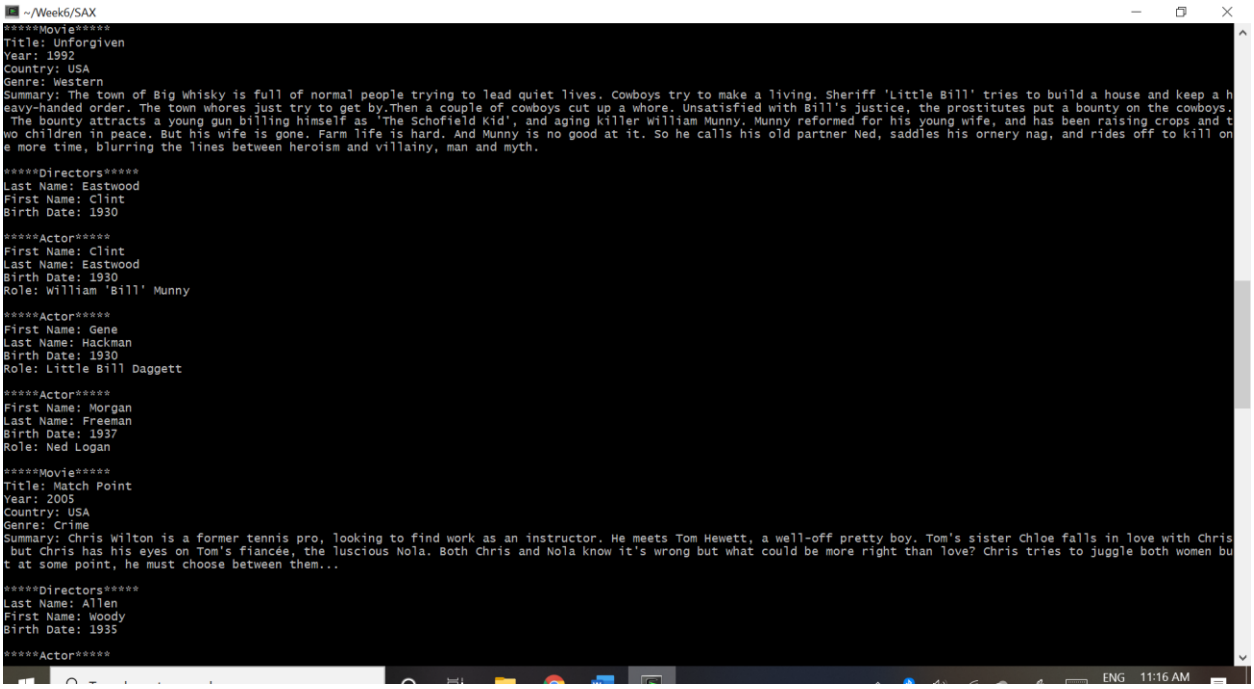
```
    elif self.CurrentData == "genre":
      print ("Genre:", self.genre)
    elif self.CurrentData == "summary":
      print ("Summary:", self.summary)
    elif self.CurrentData == "director":
      print ("Director:", self.director)
    elif self.CurrentData == "last_name":
      print ("Last Name:", self.last_name)
    elif self.CurrentData == "first_name":
      print ("First Name:", self.first_name)
    elif self.CurrentData == "birth_date":
      print ("Birth Date:", self.birth_date)
    elif self.CurrentData == "role":
      print ("Role:", self.role)
    self.CurrentData = ""

  # Call when a character is read
  def characters(self, content):
    if self.CurrentData == "title":
      self.title = content
    elif self.CurrentData == "year":
      self.year = content
    elif self.CurrentData == "country":
      self.country = content
    elif self.CurrentData == "genre":
      self.genre = content
    elif self.CurrentData == "summary":
      self.summary = content
    elif self.CurrentData == "director":
      self.director = content
    elif self.CurrentData == "last_name":
      self.last_name = content
    elif self.CurrentData == "first_name":
      self.first_name = content
    elif self.CurrentData == "birth_date":
      self.birth_date = content
    elif self.CurrentData == "role":
      self.role = content

if ( __name__ == "__main__"):
```

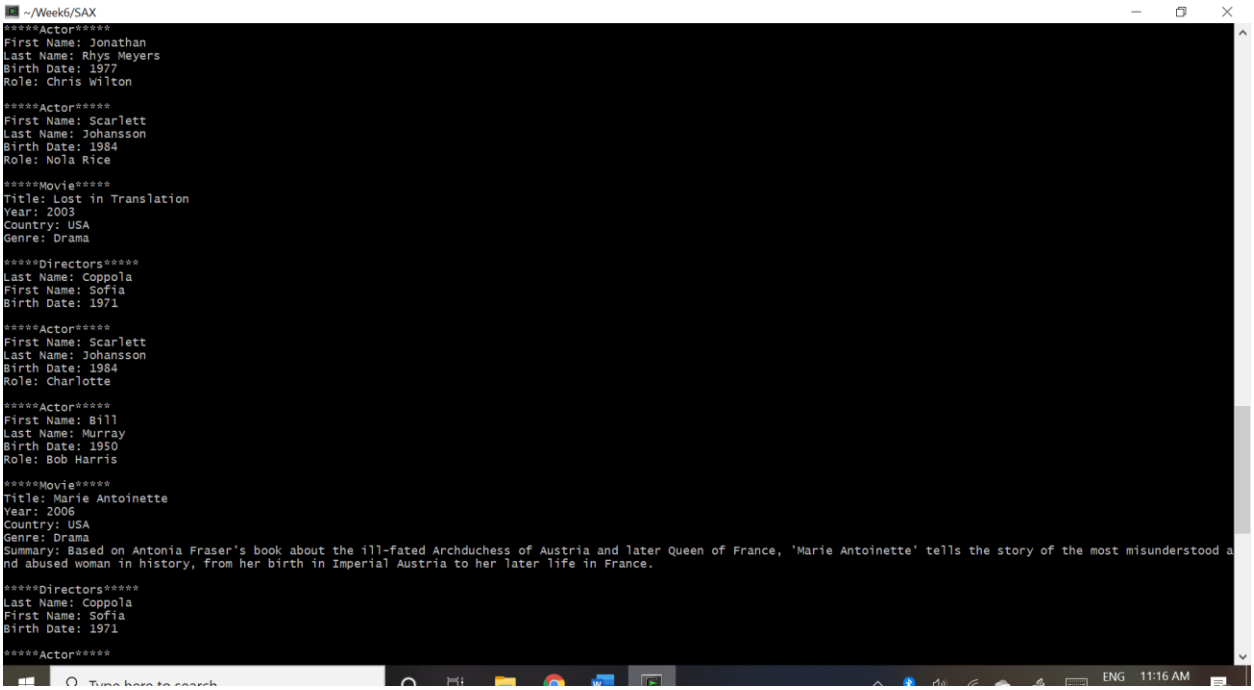
```
# create an XMLReader
parser = xml.sax.make_parser()
# turn off namepsaces
parser.setFeature(xml.sax.handler.feature_namespaces, 0)

# override the default ContextHandler
Handler = MovieHandler()
parser.setContentHandler( Handler )

parser.parse("movies_long.xml")
```

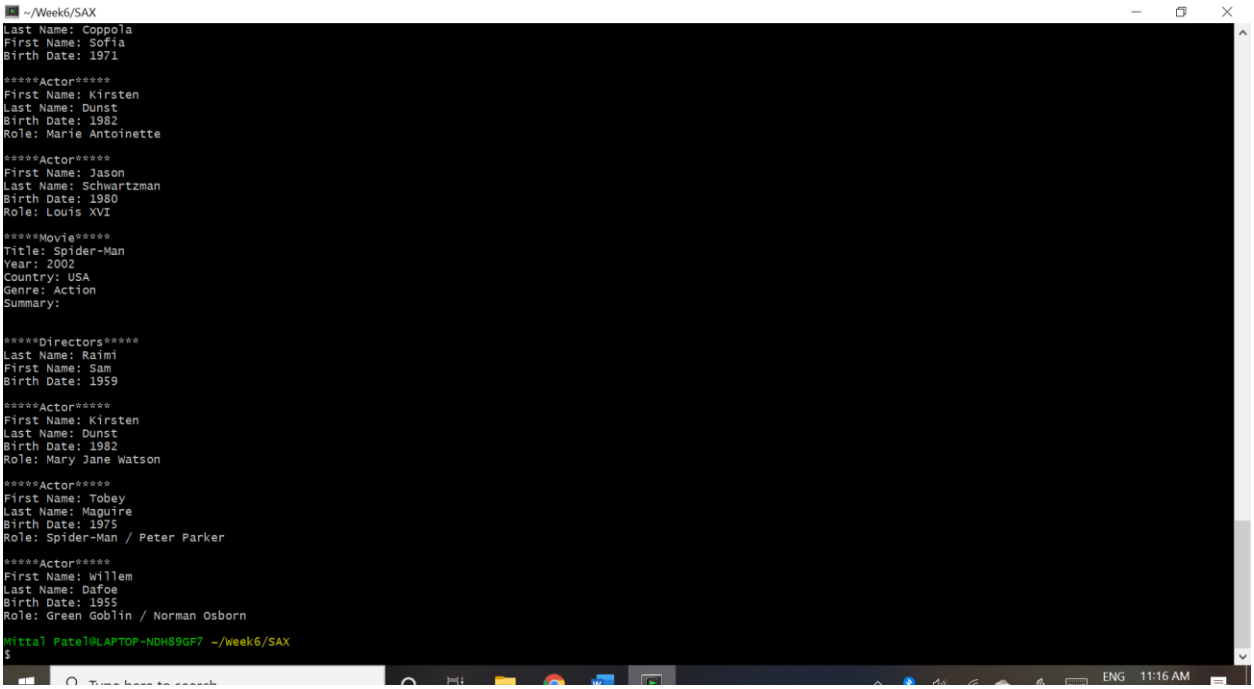




~/Week6/SAX
$ cd SAX
Mittal Patel@LAPTOP-NDH89GF7 ~/Week6/SAX
$ python3 MovieHandler.py
*****Movie*****
Title: A History of Violence
Year: 2005
Country: USA
Genre: Crime
Summary: Tom Stall, a humble family man and owner of a popular neighborhood restaurant, lives a quiet but fulfilling existence in the Midwest. One night Tom foils a crime a
t his place of business and, to his chagrin, is plastered all over the news for his heroics. Following this, mysterious people follow the Stalls' every move, concerning Tom
more than anyone else. As this situation is confronted, more lurks out over where all these occurrences have stemmed from compromising his marriage, family relationship an
d the main characters' former relations in the process.
*****Directors*****
Last Name: Cronenberg
First Name: David
Birth Date: 1943
*****Actor*****
First Name: Vigo
Last Name: Mortensen
Birth Date: 1958
Role: Tom Stall
*****Actor*****
First Name: Maria
Last Name: Bello
Birth Date: 1967
Role: Eddie Stall
*****Actor*****
First Name: Ed
Last Name: Harris
Birth Date: 1950
Role: Carl Fogarty
*****Actor*****
First Name: William
Last Name: Hurt
Birth Date: 1950
Role: Richie Cusack
*****Movie*****
Title: Heat
Year: 1995
Country: USA
Genre: Crime
Type here to search
ENG 11:15 AM


~/Week6/SAX
*****Movie*****
Title: Heat
Year: 1995
Country: USA
Genre: Crime
Summary: Hunters and their prey--Neil and his professional criminal crew hunt to score big money targets (banks, vaults, armored cars) and are, in turn, hunted by Lt. Vince
nt Hanna and his team of cops in the Robbery/Homicide police division. A botched job puts Hanna onto their trail while they regroup and try to put together one last big 're
tirement' score. Neil and Vincent are similar in many ways, including their troubled personal lives. At a crucial moment in his life, Neil disobeys the dictum taught to him
long ago by his criminal mentor--'Never have anything in your life that you can't walk out on in thirty seconds flat, if you spot the heat coming around the corner'--as he
falls in love. Thus the stage is set for the suspenseful ending....
*****Directors*****
Last Name: Mann
First Name: Michael
Birth Date: 1943
*****Actor*****
First Name: Al
Last Name: Pacino
Birth Date: 1940
Role: Lt. Vincent Hanna
*****Actor*****
First Name: Robert
Last Name: De Niro
Birth Date: 1943
Role: Neil McCauley
*****Actor*****
First Name: Val
Last Name: Kilmer
Birth Date: 1959
Role: Chris Shiherlis
*****Actor*****
First Name: Jon
Last Name: Voight
Birth Date: 1938
Role: Nate
*****Movie*****
Title: Unforgiven
Year: 1992
Country: USA
Genre: Western
Summary: The town of Big Whisky is full of normal people trying to lead quiet lives. Cowboys try to make a living. Sheriff 'Little Bill' tries to build a house and keep a h
eavy-handed order. The town whores just try to get by.Then a couple of cowboys cut up a whore. Unsatisfied with Bill's justice, the prostitutes put a bounty on the cowboys.
The bounty attracts a young gun billing himself as 'The Schofield Kid', and aging killer William Munny. Munny reformed for his young wife, and has been raising crops and t
wo children in peace. But his wife is gone. Farm life is hard. And Munny is no good at it. So he calls his old partner Ned, saddles his ornery nag, and rides off to kill on
ENG 11:16 AM


~/Week6/SAX
*****Movie*****
Title: Unforgiven
Year: 1992
Country: USA
Genre: Western
Summary: The town of Big Whisky is full of normal people trying to lead quiet lives. Cowboys try to make a living. Sheriff 'Little Bill' tries to build a house and keep a h
eavy-handed order. The town whores just try to get by.Then a couple of cowboys cut up a whore. Unsatisfied with Bill's justice, the prostitutes put a bounty on the cowboys.
The bounty attracts a young gun billing himself as 'The Schofield Kid', and aging killer William Munny. Munny reformed for his young wife, and has been raising crops and t
wo children in peace. But his wife is gone. Farm life is hard. And Munny is no good at it. So he calls his old partner Ned, saddles his ornery nag, and rides off to kill on
e more time, blurring the lines between heroism and villainy, man and myth.
*****Directors*****
Last Name: Eastwood
First Name: Clint
Birth Date: 1930
*****Actor*****
First Name: Clint
Last Name: Eastwood
Birth Date: 1930
Role: William 'Bill' Munny
*****Actor*****
First Name: Gene
Last Name: Hackman
Birth Date: 1930
Role: Little Bill Daggett
*****Actor*****
First Name: Morgan
Last Name: Freeman
Birth Date: 1937
Role: Ned Logan
*****Movie*****
Title: Match Point
Year: 2005
Country: USA
Genre: Crime
Summary: Chris Wilton is a former tennis pro, looking to find work as an instructor. He meets Tom Hewett, a well-off pretty boy. Tom's sister Chloe falls in love with Chris
but Chris has his eyes on Tom's fiancée, the luscious Nola. Both Chris and Nola know it's wrong but what could be more right than love? Chris tries to juggle both women bu
t at some point, he must choose between them...
*****Directors*****
Last Name: Allen
First Name: Woody
Birth Date: 1935
*****Actor*****
ENG 11:16 AM


~/Week6/SAX
*****Actor*****
First Name: Jonathan
Last Name: Rhys Meyers
Birth Date: 1977
Role: Chris Wilton
*****Actor*****
First Name: Scarlett
Last Name: Johansson
Birth Date: 1984
Role: Nola Rice
*****Movie*****
Title: Lost in Translation
Year: 2003
Country: USA
Genre: Drama
*****Directors*****
Last Name: Coppola
First Name: Sofia
Birth Date: 1971
*****Actor*****
First Name: Scarlett
Last Name: Johansson
Birth Date: 1984
Role: Charlotte
*****Actor*****
First Name: Bill
Last Name: Murray
Birth Date: 1950
Role: Bob Harris
*****Movie*****
Title: Marie Antoinette
Year: 2006
Country: USA
Genre: Drama
Summary: Based on Antonia Fraser's book about the ill-fated Archduchess of Austria and later Queen of France, 'Marie Antoinette' tells the story of the most misunderstood a
nd abused woman in history, from her birth in Imperial Austria to her later life in France.
*****Directors*****
Last Name: Coppola
First Name: Sofia
Birth Date: 1971
*****Actor*****
ENG 11:16 AM

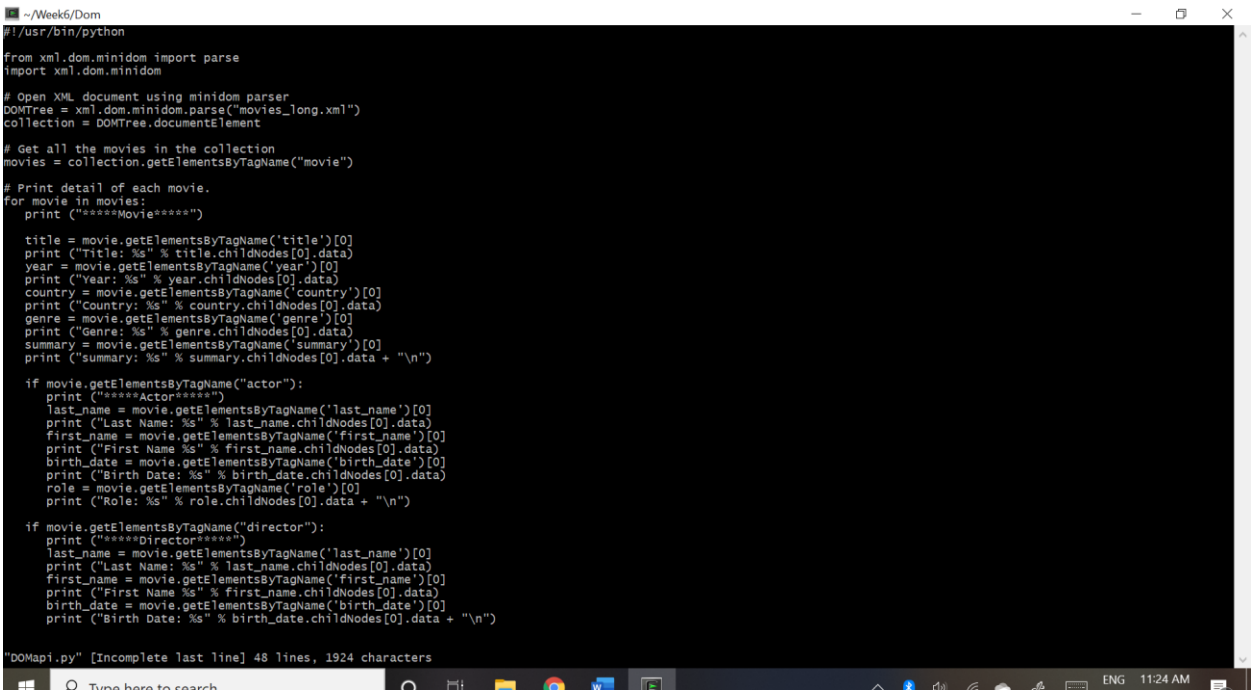

~/Week6/SAX
Last Name: Coppola
First Name: Sofia
Birth Date: 1971
*****Actor*****
First Name: Kirsten
Last Name: Dunst
Birth Date: 1982
Role: Marie Antoinette
*****Actor*****
First Name: Jason
Last Name: Schwartzman
Birth Date: 1980
Role: Louis XVI
*****Movie*****
Title: Spider-Man
Year: 2002
Country: USA
Genre: Action
Summary:
*****Directors*****
Last Name: Raimi
First Name: Sam
Birth Date: 1959
*****Actor*****
First Name: Kirsten
Last Name: Dunst
Birth Date: 1982
Role: Mary Jane Watson
*****Actor*****
First Name: Tobey
Last Name: Maguire
Birth Date: 1975
Role: Spider-Man / Peter Parker
*****Actor*****
First Name: Willem
Last Name: Dafoe
Birth Date: 1955
Role: Green Goblin / Norman Osborn
Mittal Patel@LAPTOP-NDH89GF7 ~/Week6/SAX
$
ENG 11:16 AM

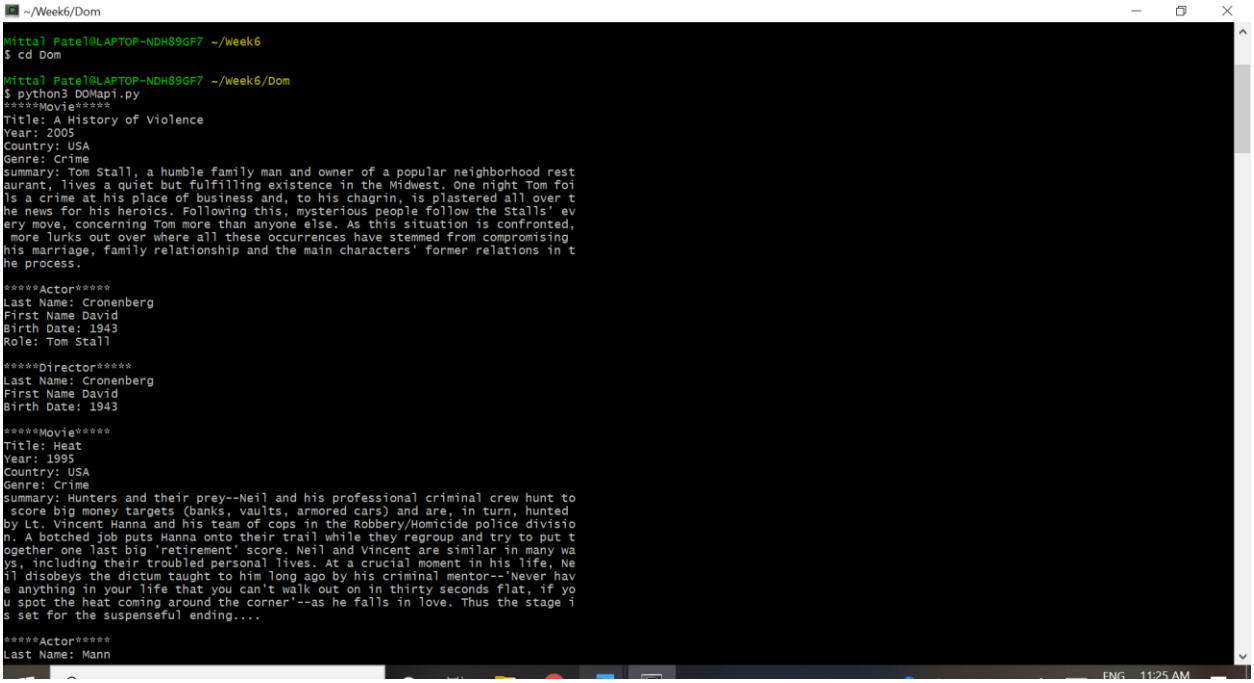
## Q3 ==> Parsing XML with DOM APIs

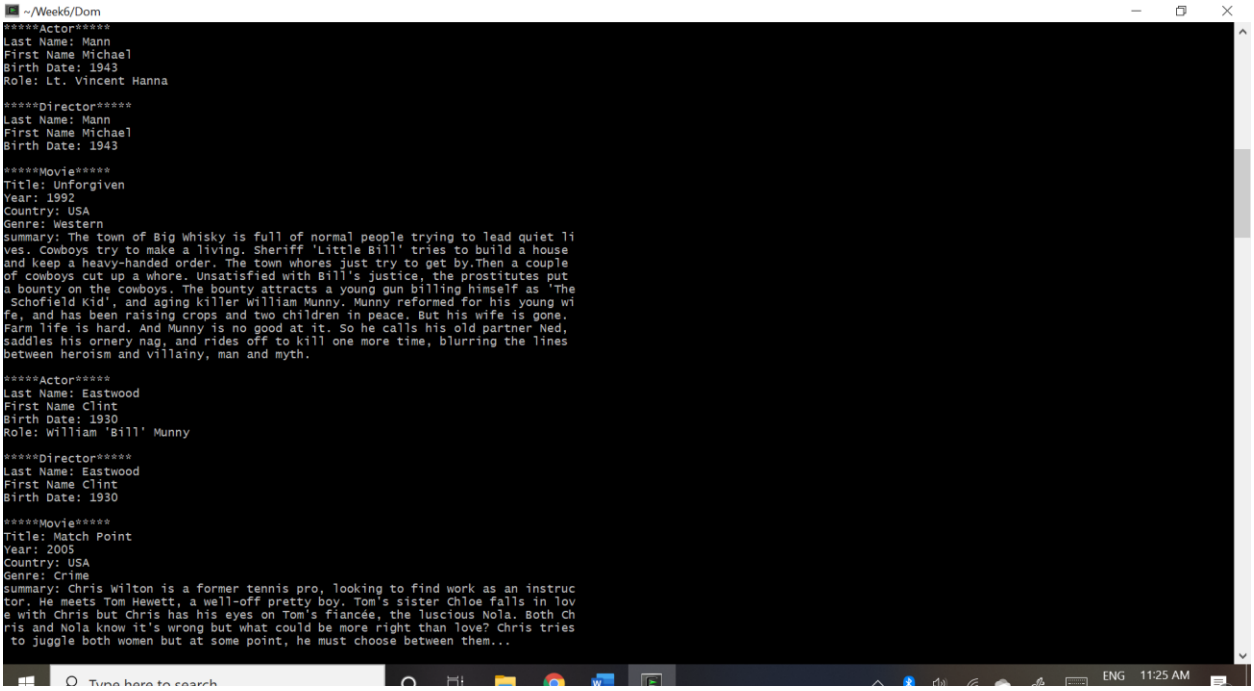
### DOMapi.py

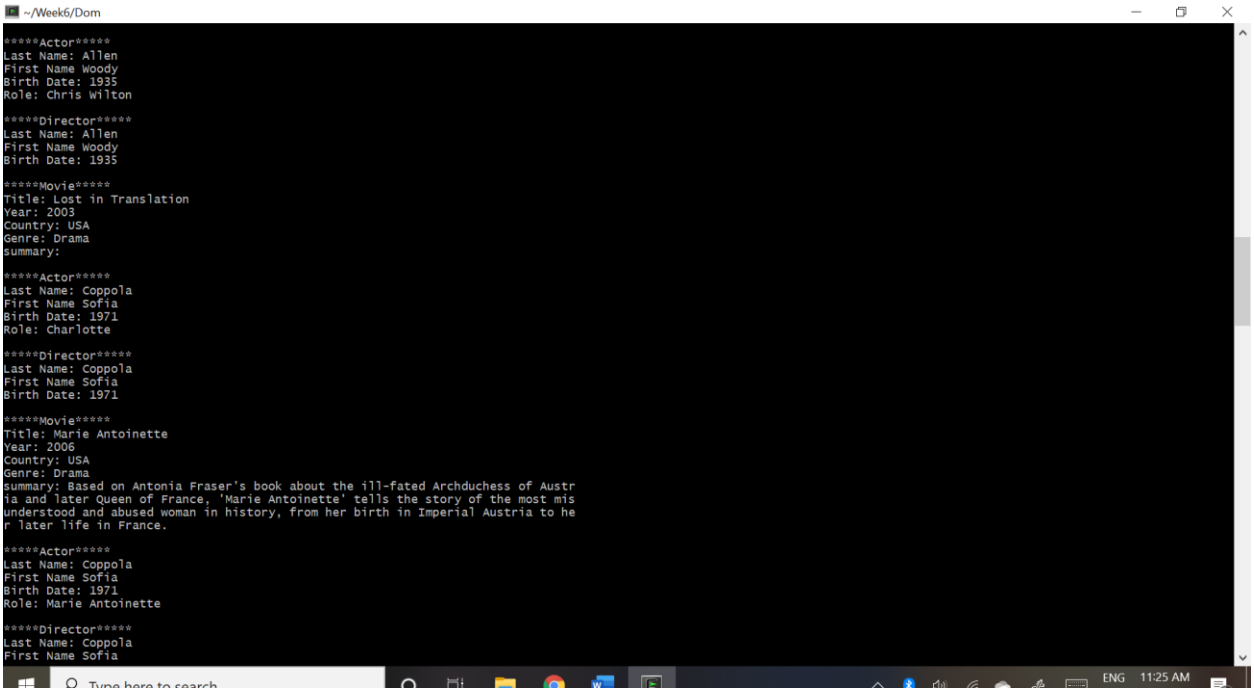
```
#!/usr/bin/python

from xml.dom.minidom import parse
import xml.dom.minidom

# Open XML document using minidom parser
DOMTree = xml.dom.minidom.parse("movies_long.xml")
collection = DOMTree.documentElement

# Get all the movies in the collection
movies = collection.getElementsByTagName("movie")

# Print detail of each movie.
for movie in movies:
  print ("*****Movie*****")

  title = movie.getElementsByTagName('title')[0]
  print ("Title: %s" % title.childNodes[0].data)
  year = movie.getElementsByTagName('year')[0]
  print ("Year: %s" % year.childNodes[0].data)
  country = movie.getElementsByTagName('country')[0]
  print ("Country: %s" % country.childNodes[0].data)
  genre = movie.getElementsByTagName('genre')[0]
  print ("Genre: %s" % genre.childNodes[0].data)
  summary = movie.getElementsByTagName('summary')[0]
  print ("summary: %s" % summary.childNodes[0].data + "\n")

  if movie.getElementsByTagName("actor"):
    print ("*****Actor*****")
    last_name = movie.getElementsByTagName('last_name')[0]
    print ("Last Name: %s" % last_name.childNodes[0].data)
    first_name = movie.getElementsByTagName('first_name')[0]
    print ("First Name %s" % first_name.childNodes[0].data)
    birth_date = movie.getElementsByTagName('birth_date')[0]
    print ("Birth Date: %s" % birth_date.childNodes[0].data)
    role = movie.getElementsByTagName('role')[0]
    print ("Role: %s" % role.childNodes[0].data + "\n")

  if movie.getElementsByTagName("director"):
```

```
print ("*****Director*****")
last_name = movie.getElementsByTagName('last_name')[0]
print ("Last Name: %s" % last_name.childNodes[0].data)
first_name = movie.getElementsByTagName('first_name')[0]
print ("First Name %s" % first_name.childNodes[0].data)
birth_date = movie.getElementsByTagName('birth_date')[0]
print ("Birth Date: %s" % birth_date.childNodes[0].data + "\n")
```




~/Week6/Dom
Mittal Patel@LAPTOP-NDH89GF7 ~/Week6
$ cd Dom
Mittal Patel@LAPTOP-NDH89GF7 ~/Week6/Dom
$ python3 DOMapi.py
*****Movie*****
Title: A History of Violence
Year: 2005
Country: USA
Genre: Crime
summary: Tom Stall, a humble family man and owner of a popular neighborhood rest
aurant, lives a quiet but fulfilling existence in the Midwest. One night Tom foi
ls a crime at his place of business and, to his chagrin, is plastered all over t
he news for his heroics. Following this, mysterious people follow the Stalls' ev
ery move, concerning Tom more than anyone else. As this situation is confronted,
more lurks out over where all these occurrences have stemmed from compromising
his marriage, family relationship and the main characters' former relations in t
he process.
*****Actor*****
Last Name: Cronenberg
First Name David
Birth Date: 1943
Role: Tom Stall
*****Director*****
Last Name: Cronenberg
First Name David
Birth Date: 1943
*****Movie*****
Title: Heat
Year: 1995
Country: USA
Genre: Crime
summary: Hunters and their prey--Neil and his professional criminal crew hunt to
score big money targets (banks, vaults, armored cars) and are, in turn, hunted
by Lt. Vincent Hanna and his team of cops in the Robbery/Homicide police divisio
n. A botched job puts Hanna onto their trail while they regroup and try to put t
ogether one last big 'retirement' score. Neil and Vincent are similar in many wa
ys, including their troubled personal lives. At a crucial moment in his life, Ne
il disobeys the dictum taught to him long ago by his criminal mentor--'Never hav
e anything in your life that you can't walk out on in thirty seconds flat, if yo
u spot the heat coming around the corner'--as he falls in love. Thus the stage i
s set for the suspenseful ending....
*****Actor*****
Last Name: Mann
ENG 11:25 AM


~/Week6/Dom
*****Actor*****
Last Name: Mann
First Name Michael
Birth Date: 1943
Role: Lt. Vincent Hanna
*****Director*****
Last Name: Mann
First Name Michael
Birth Date: 1943
*****Movie*****
Title: Unforgiven
Year: 1992
Country: USA
Genre: Western
summary: The town of Big Whisky is full of normal people trying to lead quiet li
ves. Cowboys try to make a living. Sheriff 'Little Bill' tries to build a house
and keep a heavy-handed order. The town whores just try to get by.Then a couple
of cowboys cut up a whore. Unsatisfied with Bill's justice, the prostitutes put
a bounty on the cowboys. The bounty attracts a young gun billing himself as 'The
Schofield Kid', and aging killer William Munny. Munny reformed for his young wi
fe, and has been raising crops and two children in peace. But his wife is gone.
Farm life is hard. And Munny is no good at it. So he calls his old partner Ned,
saddles his ornery nag, and rides off to kill one more time, blurring the lines
between heroism and villainy, man and myth.
*****Actor*****
Last Name: Eastwood
First Name Clint
Birth Date: 1930
Role: William 'Bill' Munny
*****Director*****
Last Name: Eastwood
First Name Clint
Birth Date: 1930
*****Movie*****
Title: Match Point
Year: 2005
Country: USA
Genre: Crime
summary: Chris Wilton is a former tennis pro, looking to find work as an instruc
tor. He meets Tom Hewett, a well-off pretty boy. Tom's sister Chloe falls in lov
e with Chris but Chris has his eyes on Tom's fiancée, the luscious Nola. Both Ch
ris and Nola know it's wrong but what could be more right than love? Chris tries
to juggle both women but at some point, he must choose between them...
Type here to search
ENG 11:25 AM


~/Week6/Dom
*****Actor*****
Last Name: Allen
First Name Woody
Birth Date: 1935
Role: Chris Wilton
*****Director*****
Last Name: Allen
First Name Woody
Birth Date: 1935
*****Movie*****
Title: Lost in Translation
Year: 2003
Country: USA
Genre: Drama
summary:
*****Actor*****
Last Name: Coppola
First Name Sofia
Birth Date: 1971
Role: Charlotte
*****Director*****
Last Name: Coppola
First Name Sofia
Birth Date: 1971
*****Movie*****
Title: Marie Antoinette
Year: 2006
Country: USA
Genre: Drama
summary: Based on Antonia Fraser's book about the ill-fated Archduchess of Austr
ia and later Queen of France, 'Marie Antoinette' tells the story of the most mis
understood and abused woman in history, from her birth in Imperial Austria to he
r later life in France.
*****Actor*****
Last Name: Coppola
First Name Sofia
Birth Date: 1971
Role: Marie Antoinette
*****Director*****
Last Name: Coppola
First Name Sofia
Type here to search
ENG
11:25 AM


~/Week6/Dom
*****Director*****
Last Name: Coppola
First Name Sofia
Birth Date: 1971
*****Movie*****
Title: Spider-Man
Year: 2002
Country: USA
Genre: Action
summary:
On a school field trip, Peter Parker (Maguire) is bitten by a genetically modifi
ed spider. He wakes up the next morning with incredible powers. After witnessing
the death of his uncle (Robertson), Parkers decides to put his new skills to us
e in order to rid the city of evil, but someone else has other plans. The Green
Goblin (Dafoe) sees Spider-Man as a threat and must dispose of him. Even if it m
eans the Goblin has to target Parker's Aunt (Harris) and the girl he secretly pi
nes for (Dunst)
*****Actor*****
Last Name: Raimi
First Name Sam
Birth Date: 1959
Role: Mary Jane Watson
*****Director*****
Last Name: Raimi
First Name Sam
Birth Date: 1959
Mittal Patel@LAPTOP-NDH89GF7 ~/Week6/Dom
$ vi DOMapi.py
Mittal Patel@LAPTOP-NDH89GF7 ~/Week6/Dom
$ python3 DOMapi.py
*****Movie*****
Title: A History of Violence
Year: 2005
Country: USA
Genre: Crime
summary: Tom Stall, a humble family man and owner of a popular neighborhood restaurant, lives a quiet but fulfilling existence in the Midwest. One night Tom foils a crime a
t his place of business and, to his chagrin, is plastered all over the news for his heroics. Following this, mysterious people follow the Stalls' every move, concerning Tom
more than anyone else. As this situation is confronted, more lurks out over where all these occurrences have stemmed from compromising his marriage, family relationship an
d the main characters' former relations in the process.
*****Actor*****
Last Name: Cronenberg
Type here to search
ENG
11:25 AM

## Q4 ==> Json + pychat

### Pychat_client.py

```
import select, socket, sys
from pychat_util import Room, Hall, Player
import pychat_util
READ_BUFFER = 4096
if len(sys.argv) < 2:
    print("Usage: Python3 client.py [hostname]", file = sys.stderr)
    sys.exit(1)
else:
    server_connection = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    server_connection.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
    server_connection.connect((sys.argv[1], pychat_util.PORT))

def prompt():
    print('>', end=' ', flush = True)
print("Connected to server\n")
msg_prefix = ''
socket_list = [sys.stdin, server_connection]
while True:
    read_sockets, write_sockets, error_sockets = select.select(socket_list, [], [])
```

```
for s in read_sockets:
    if s is server_connection: # incoming message
        msg = s.recv(READ_BUFFER)
        if not msg:
            print("Server down!")
            sys.exit(2)
        else:
            if msg == pychat_util.QUIT_STRING.encode():
                sys.stdout.write('Bye\n')
                sys.exit(2)
            else:
                sys.stdout.write(msg.decode())
                if 'Please tell us your name' in msg.decode():
                    msg_prefix = 'name: ' # identifier for name
                else:
                    msg_prefix = ''
                prompt()
    else:
        msg = msg_prefix + sys.stdin.readline()
        server_connection.sendall(msg.encode())
```

## Pychat_server.py

```
# implementing 3-tier structure: Hall --> Room --> Clients;

import select, socket, sys, pdb

from pychat_util import Hall, Room, Player

import pychat_util

READ_BUFFER = 4096

host = sys.argv[1] if len(sys.argv) >= 2 else ''

listen_sock = pychat_util.create_socket((host, pychat_util.PORT))

hall = Hall()

connection_list = []

connection_list.append(listen_sock)

while True:

   # Player.fileno()

   read_players, write_players, error_sockets = select.select(connection_list, [], [])

   for player in read_players:

      if player is listen_sock: # new connection, player is a socket

         new_socket, add = player.accept()

         new_player = Player(new_socket)

         connection_list.append(new_player)

         hall.welcome_new(new_player)

      else: # new message
```

```
            msg = player.socket.recv(READ_BUFFER)
            if msg:
                msg = msg.decode().lower()
                hall.handle_msg(player, msg)
            else:
                player.socket.close()
                connection_list.remove(player)
    for sock in error_sockets: # close error sockets
        sock.close()
        connection_list.remove(sock)
```

## Pychat_util.py

```
################################################################
# implementing 3-tier structure:
#       Hall --> Room --> Clients;
################################################################
import socket, pdb, json

MAX_CLIENTS = 30
PORT = 22222
QUIT_STRING = '<$quit$>'


def create_socket(address):
   s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
   s.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
   s.setblocking(0)
   s.bind(address)
   s.listen(MAX_CLIENTS)
   print("Now listening at ", address)
   return s

class Hall:
   def __init__(self):
      self.rooms = {} # {room_name: Room}
      self.room_player_map = {} # {playerName: roomName}

   def welcome_new(self, new_player):
      new_player.socket.sendall(b'Welcome to pychat.\nPlease tell us your name:\n')

   def list_rooms(self, player):

      if len(self.rooms) == 0:
         msg = 'Oops, no active rooms currently. Create your own!\n' \
            + 'Use [<join> room_name] to create a room.\n'
         player.socket.sendall(msg.encode())
      else:
         msg = 'Listing current rooms...\n'
         for room in self.rooms:
            msg += room + ": " + str(len(self.rooms[room].players)) + " player(s)\n"
         player.socket.sendall(msg.encode())
```

```
def handle_msg(self, player, msg):
    instructions = b'Instructions:\n'\
        + b'[<list>] to list all rooms\n'\
        + b'[<join> room_name] to join/create/switch to a room\n' \
        + b'[<manual>] to show instructions\n' \
        + b'[<json>] to send JSON\n' \
        + b'[<quit>] to quit\n' \
        + b'Otherwise start typing and enjoy!' \
        + b'\n'

    print(player.name + " says: " + msg)
    if "name:" in msg:
        name = msg.split()[1]
        player.name = name
        print("New connection from:", player.name)
        player.socket.sendall(instructions)

    elif "<join>" in msg:
        same_room = False
        if len(msg.split()) >= 2: # error check
            room_name = msg.split()[1]
            if player.name in self.room_player_map: # switching?
                if self.room_player_map[player.name] == room_name:
                    player.socket.sendall(b'You are already in room: ' + room_name.encode())
                    same_room = True
                else: # switch
                    old_room = self.room_player_map[player.name]
                    self.rooms[old_room].remove_player(player)
            if not same_room:
                if not room_name in self.rooms: # new room:
                    new_room = Room(room_name)
                    self.rooms[room_name] = new_room
                self.rooms[room_name].players.append(player)
                self.rooms[room_name].welcome_new(player)
                self.room_player_map[player.name] = room_name
        else:
            player.socket.sendall(instructions)

    elif "<list>" in msg:
        self.list_rooms(player)
```

```
        elif "<manual>" in msg:
            player.socket.sendall(instructions)

        elif "<json>" in msg:
            if len(msg.split()) >= 2: # error check
                json_string = msg.split()[1]
                print(json.dumps(json_string))
            else:
                player.socket.sendall(instructions)

        elif "<quit>" in msg:
            player.socket.sendall(QUIT_STRING.encode())
            self.remove_player(player)

        else:
            ###################################
            # check if in a room or not first
            ###################################
            if player.name in self.room_player_map:
                self.rooms[self.room_player_map[player.name]].broadcast(player, msg.encode())
            else:
                msg = 'You are currently not in any room! \n' \
                    + 'Use [<list>] to see available rooms! \n' \
                    + 'Use [<join> room_name] to join a room! \n'
                player.socket.sendall(msg.encode())

    def remove_player(self, player):
        if player.name in self.room_player_map:
            self.rooms[self.room_player_map[player.name]].remove_player(player)
            del self.room_player_map[player.name]
        print("Player: " + player.name + " has left\n")
class Room:
    def __init__(self, name):
        self.players = [] # a list of sockets
        self.name = name

    def welcome_new(self, from_player):
        msg = self.name + " welcomes: " + from_player.name + '\n'
        for player in self.players:
            player.socket.sendall(msg.encode())
```

```
    def broadcast(self, from_player, msg):
        msg = from_player.name.encode() + b":" + msg
        for player in self.players:
            player.socket.sendall(msg)

    def remove_player(self, player):
        self.players.remove(player)
        leave_msg = player.name.encode() + b"has left the room\n"
        self.broadcast(player, leave_msg)

class Player:
    def __init__(self, socket, name = "new"):
        socket.setblocking(0)
        self.socket = socket
        self.name = name

    def fileno(self):
        return self.socket.fileno()
```