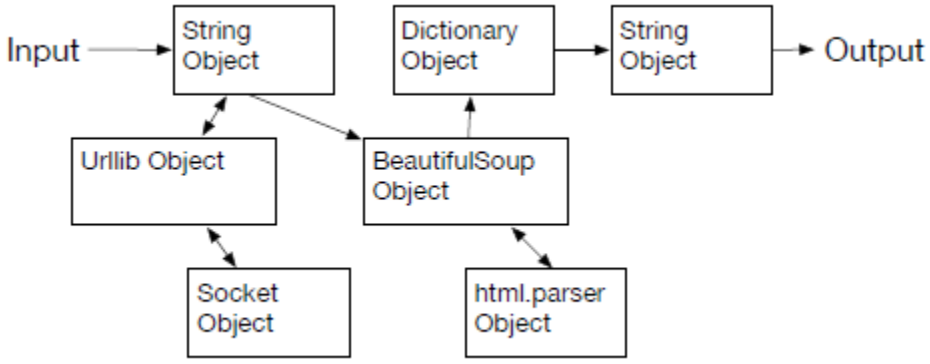
Solution/Model Answer
Python Application Programming (15CS664)
IAT-3, 6th Sem CSE
Dr. P. N. Singh, Professor(CSE)

---

1 a) What do you mean by "orchestrating the movement of data between objects". Discuss with example code. 5m

Ans:
In OOPs a program is "orchestrating the movement of data between objects". Back then it was just lines of code that got the job done. It is managing data in motion. It is how we build a network of interacting objects and orchestrate the movement of information between the objects to create a program.



A program as network of objects

```
#any simple example code for OOPs:
class song:
        def gao(self):
                print("Ek pyar ka nagma hai")

s1=song( )
s2=song( )
s1.gao( )
s2.gao( )
```

The same lyrics move within objects

1 b) How multiple inheritance can be used in Python programming. Explain with example program. 5m

Ans:
Python supports a limited form of multiple inheritance as well. A sub-class is derived from more than one base class. A class definition with multiple base classes looks like this:

```
class DerivedClassName(Base1, Base2, Base3):
     <statement-1>
                 ……
#multiple inheritance
class horse:
    def hsound(self):
        print("Neighss....Hin...n.n..")

class ass:
    def asound(self):
        print("bra....ys.......dhinchu.")

class donkey(horse,ass):
    def dsound(self):
        print("Neighss....Hin..Braysss... ss...Dhinchu")

d1 = donkey()
d1.hsound()
d1.asound()
d1.dsound()
```

2. Differentiate data hiding & data encapsulation. Implement data hiding in a OOPs program of Python 10m

Ans:

Data Encapsulation: Data and methods operate on data are wrapped-in(confined) within a class. Private members are not accessed directly outside of the class.

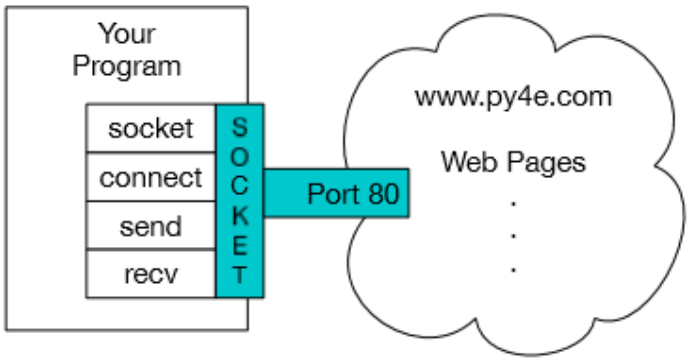
Data Hiding: An object's attributes may or may not be visible outside the class definition. Visibility of private members is available through public member functions. We need to name attributes with a double underscore prefix, and those attributes then are not be directly visible to outsiders.

```
#data hiding - protection - Example
class JustCounter:
   __secretCount = 0  # variable should preceed double underscore

   def count(self):
      self.__secretCount += 1
      print (self.__secretCount)

counter = JustCounter()
counter.count()
```

```
counter.count()
# print (counter.__secretCount)  error, yes it can't be accessed
outside
print (counter._JustCounter__secretCount) # works
```

3 a) What is a socket? How does http protocol work? Define parameters AF_INET & SOCK_STREAM. 5m

Ans:
Socket: The network protocol that powers the web to make network connections is classed socket. It is built-in feature in python which makes it very easy to make network connections and retrieve data over those sockets in a Python program.



The easiest way to show how the HTTP protocol works is to write a very simple Python program that makes a connection to a web server and follows the rules of the HTTP protocol to request a document and display what the server sends back.
Syntax of get request defined in above document:
GET http://data.pr4e.org/romeo.txt HTTP/1.0
80 is port number of www.py4e.org server
to calls the socket class from socket library to create a network end point

```
>>> import socket
>>> mysock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
```

The first parameter is **AF_INET** and the second one is **SOCK_STREAM**. AF_INET refers to the address family ipv4. The SOCK_STREAM means connection oriented TCP protocol.

3 b) Write a Python program using socket to connect 'data.pr4e.org' and to print the contents of 'romeo.txt'. 5m

Ans:

```
#socket program example
 import socket
 mysock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
 mysock.connect(('data.pr4e.org', 80))
 cmd = 'GET http://data.pr4e.org/romeo.txt
HTTP/1.0\r\n\r\n'.encode()
 mysock.send(cmd)
 while True:
             data=mysock.recv(20)
     if(len(data)<1):
          break
     print(data.decode(),end='')
```

4 a) Explain with example to retrieve a web-page with urlib library.
5m

Ans:
While we can manually send and receive data over HTTP using the socket library, there is a much simpler way to perform this common task in Python by using the urllib library.
Using urllib, you can treat a web page much like a file. You simply indicate which web page you would like to retrieve and urllib handles all of the HTTP protocol and header details.

```
>>> import urllib.request
>>> fhand= urllib.request.urlopen('http://data.pr4e.org/romeo.txt')
>>> for line in fhand:
     print(line.decode().strip())
```

Once the web page has been opened with urllib.urlopen, we can treat it like a file and read through it using a for loop.
When the program runs, we only see the output of the contents of the file. The headers are still sent, but the urllib code consumes the headers and only returns the data to us.
As an example, we can write a program to retrieve the data for romeo.txt and compute the frequency of each word in the file as follows:

```
>>> import urllib.request, urllib.parse, urllib.error
>>> fhand = urllib.request.urlopen('http://data.pr4e.org/romeo.txt')
>>> counts=dict()
>>> for line in fhand:
     words=line.decode().split()
     for word in words:
          counts[word]=counts.get(word,0)+1

>>> print(counts)
```

4 b) What is 'scrap the web' or web scraping? Write with Example code.
5m

Ans:

One of the common uses of the urllib capability in Python is to scrape the web. Web scraping is when we write a program that pretends to be a web browser and retrieves pages, then examines the data in those pages looking for patterns.

```
>>> import re
>>> url=input('Enter - ')
Enter - http://data.pr4e.org/romeo.txt
>>> html=urllib.request.urlopen(url).read()
>>> links=re.findall(b'A.+',html)
>>> for link in links:
     print(link.decode())
```

5 a) What is the use of XML(eXtensible Markup Language)? Write one sample XML document.
5m

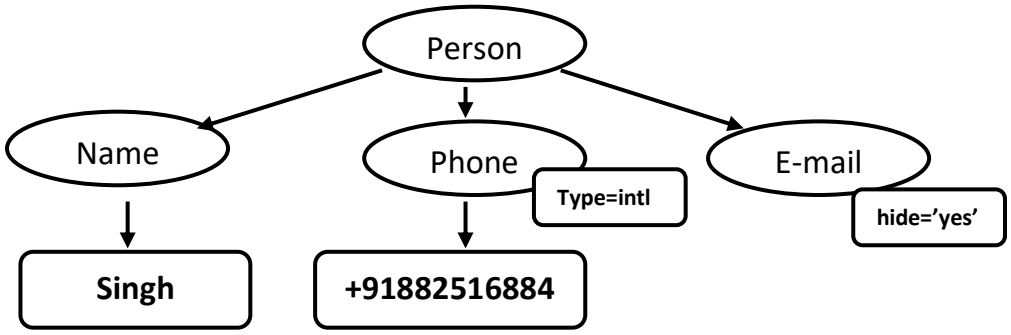
Ans:

There are formats that we use when exchanging data across the web. The "eXtensible Markup Language" or XML has been in use for a very long time and is best suited for exchanging document-style data.

XML(eXtensible Markup Language): XML looks very similar to HTML, but XML is more structured than HTML. Here is a sample of an XML document:

```
<person>
     <name>Singh</name>
     <phone type="intl">
          +918825168846
     </phone>
     <email hide="yes"/>
</person>
```

Often it is helpful to think of an XML document as a tree structure where there is a top tag person and other tags such as phone are drawn as children of their parent nodes.



Here is a simple application that parses some XML and extracts some data elements from the XML:

```
>>> import xml.etree.ElementTree as ET
>>> data='''
<person>
     <name>Singh</name>
     <phone type="intl">
          +918825168846
     </phone>
     <email hide="yes"/>
</person> '''
>>> tree=ET.fromstring(data)
>>> print('Name:', tree.find('name').text)
Name: Singh
>>> print('Attr:', tree.find('email').get('hide'))
Attr: yes
>>>
```

5 b) Often the XML has multiple nodes and we need to write a loop to process all of the nodes. Write a program to loop through all the user nodes.
5m

Ans:

 # In the following program, we loop through all of the user nodes:

```
 import xml.etree.ElementTree as ET
 input = '''
<stuff>
    <users>
       <user x="2">
           <id>12504</id>
           <name>Singh</name>
       </user>
       <user x="5">
           <id>12505</id>
           <name>Rani</name>
```

```
       </user>
    </users>
</stuff>'''
stuff=ET.fromstring(input)
lst=stuff.findall('users/user')
 for item in lst:
     print('Name',item.find('name').text)
     print('Id',item.find('id').text)
     print('Attribute',item.get("x"))
```

6 a) What is JSON or JSON format? Implement parsing the JSON & read through the data in a Python Program. 5m

Ans:

The JSON format was inspired by the object and array format used in the JavaScript language. But since Python was invented before JavaScript, Python's syntax for dictionaries and lists influenced the syntax of JSON. So the format of JSON is nearly identical to a combination of Python lists and dictionaries.

Parsing JSON: We construct our JSON by nesting dictionaries (objects) and lists as needed. In this example, we represent a list of users where each user is a set of key-value pairs (i.e., a dictionary). So we have a list of dictionaries.

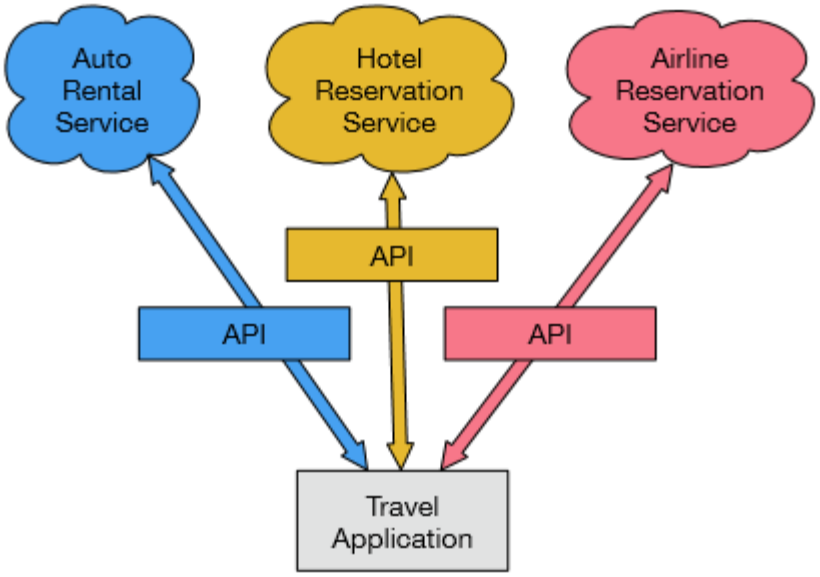
In the following program, we use the built-in json library to parse the JSON and read through the data.

```
import json

data = '''
[
  { "id" : "001",
    "x" : "2",
    "name" : "Dr. P. N. Singh"
  } ,
  { "id" : "009",
    "x" : "7",
    "name" : "Dr. Jhansi Rani"
  }
]'''

info = json.loads(data)
print('User count:', len(info))

for item in info:
    print('Name', item['name'])
    print('Id', item['id'])
    print('Attribute', item['x'])
```

6 b) Define API with diagram? What are 2 advantages of Service Oriented Architecture? 5m

Ans:
Application Programming Interface: "To define document "contracts" between applications"
The general name for these application-to-application contracts is Application Program Interfaces or APIs.

When we begin to build our programs where the functionality of our program includes access to services provided by other programs, we call the approach a Service-Oriented Architecture or SOA. A SOA approach is one where our overall application makes use of the services of other applications. A non-SOA approach is where the application is a single standalone application which contains all of the code necessary to implement the application.



Advantages of SOA:
(1) we always maintain only one copy of data (this is particularly important for things like hotel reservations where we do not want to over-commit) and
(2) the owners of the data can set the rules about the use of their data.

7. Write a program using geocoding API to find out search string from google map and to say about land marks nearby. 5m

Ans:

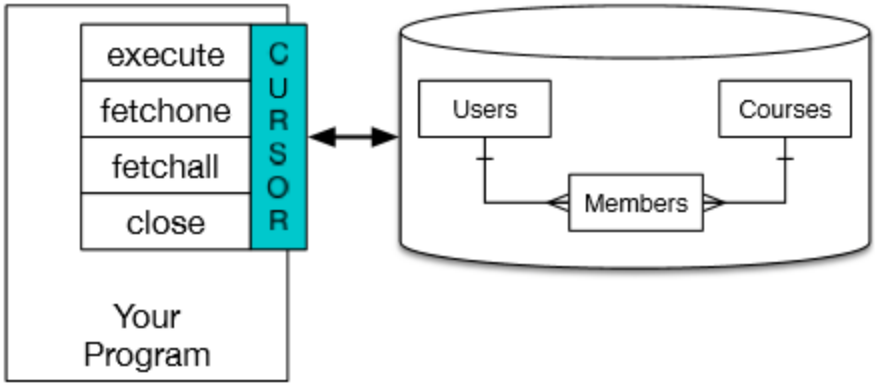
```
#Example program/application: geojson.py

import urllib.request, urllib.parse, urllib.error
import json

# Note that Google is increasingly requiring keys
# for this API
serviceurl = 'http://maps.googleapis.com/maps/api/geocode/json?'

while True:
    address = input('Enter location: ')
    if len(address) < 1: break

    url = serviceurl + urllib.parse.urlencode(
        {'address': address})

    print('Retrieving', url)
    uh = urllib.request.urlopen(url)
    data = uh.read().decode()
    print('Retrieved', len(data), 'characters')

    try:
        js = json.loads(data)
    except:
        js = None

    if not js or 'status' not in js or js['status'] != 'OK':
        print('==== Failure To Retrieve ====')
        print(data)
        continue

    print(json.dumps(js, indent=4))

    lat = js["results"][0]["geometry"]["location"]["lat"]
    lng = js["results"][0]["geometry"]["location"]["lng"]
    print('lat', lat, 'lng', lng)
    location = js['results'][0]['formatted_address']
    print(location)
```

8 a) What is a cursor( ) in terms of sqlite? Write an example program to create table student with fields roll, name and marks & insert 3 records into table.

5m

Ans:

A cursor is like a file handle that we can use to perform operations on the data stored in the database. Calling cursor() is very similar conceptually to calling open() when dealing with text files.



Once we have the cursor, we can begin to execute commands on the contents of the database using the execute() method.

```
#program to create table student
import sqlite3
conn = sqlite3.connect('mydata.db')
cur=conn.cursor()
cur.execute('CREATE TABLE student(roll integer, name text, marks integer)')
cur.execute('insert into student values(111,"Paras",78)')
cur.execute('insert into student values(222,"Kumar",85)')
cur.execute('insert into student values(111,"Amit",67)')
conn.commit()
cur.execute('Select * from students')
for row in cur:
    print(row)
```

8 b) Write a python program using SQL commands in python two join two tables.
5m

Ans:

```
import sqlite3

conn = sqlite3.connect('friends.sqlite')
cur = conn.cursor()

cur.execute('SELECT * FROM People')
count = 0
```

```
print('People:')
for row in cur:
    if count < 5: print(row)
    count = count + 1
print(count, 'rows.')

cur.execute('SELECT * FROM Follows')
count = 0
print('Follows:')
for row in cur:
    if count < 5: print(row)
    count = count + 1
print(count, 'rows.')

cur.execute('''SELECT * FROM Follows JOIN People
            ON Follows.to_id = People.id
            WHERE Follows.from_id = 2''')
count = 0
print('Connections for id=2:')
for row in cur:
    if count < 5: print(row)
    count = count + 1
print(count, 'rows.')

cur.close()
```