Mawlana Bhashani Science And Technology University

Lab-Report

Lab Report No: 02

Lab Report Name: Programing with Python

Group member ID: IT-18019 and IT-18037

Date of Performance:

Date of Submission:

Submitted by

Name:Md.Shamim

ID: IT-18019

3rd Year 2nd Semester

Session: 2017-2018

Dept. of ICT,MBSTU

Submitted To

Nazrul Islam

Assistant Professor

Dept. of ICT

MBSTU.

1. Objectives:

Understand how python function works
Understand the use of global and local variables
Understand how python modules works
Learning the basis of networking programing with python

2. Theory:

Python functions: Functions are reusable pieces of programs. They allow you to give a name to a block of statements, allowing you to run that block using the specified name anywhere in the program and any number of times. This is known as calling the function.
Local Variables: Variables declared inside a function definition are not related in any way to other variables with the same names used outside the function (variable names are local to the function). This is called the scope of the variable. All variables have the scope of the block they are declared in starting from the point of definition of the name.
The global statement: Variables defined at the top level of the program are intended global.
Global variables are intended to be used in any functions or classes). Global statement allows defining global variables inside functions as well.

Modules: Modules allow reusing a number of functions in other programs.

Networking background for sockets

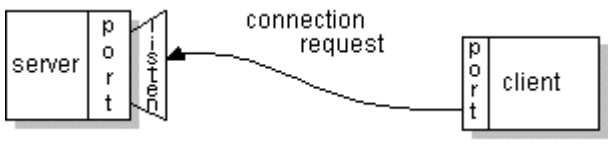
What is a socket and how use it? A socket is one endpoint of a two-way communication link between two programs running on the network or PC. A socket is bound to a port number so that the TCP layer can identify the application that data is destined to be sent to.

Endpoint: An endpoint is a combination of an IP address and a port number.

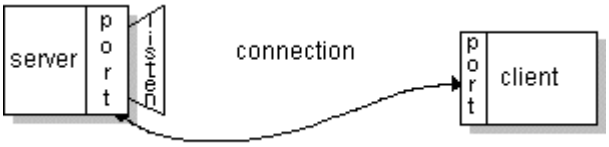
Server and Client: Normally, a server runs on a specific computer and has a socket that is bound to a specific port number.

On the server-side: The server just waits, listening to the socket for a client to make a connection request.

On the client-side: The client knows the hostname of the machine on which the server is running and the port number on which the server is listening. To make a connection request, the client tries to rendezvous with the server on the server's machine and port. The client also needs to identify itself to the server so it binds to a local port number that it will use during this connection. This is usually assigned by the system.



If everything goes well, the server accepts the connection. Upon acceptance, the server gets a new socket bound to the same local port and also has its remote endpoint set to the address and port of the client. It needs a new socket so that it can continue to listen to the original socket for connection requests while tending to the needs of the connected client.



On the client side, if the connection is accepted, a socket is successfully created and the client can use the socket to communicate with the server. The client and server can now communicate by writing to or reading from their sockets.

TCP: TCP stands for transmission control protocol. It is implemented in the transport layer of the IP/TCP model and is used to establish reliable connections. TCP is one of the protocols that encapsulate data into packets. It then transfers these to the remote end of the connection using the methods available on the lower layers. On the other end, it can check

for errors, request certain pieces to be resent, and reassemble the information into one logical piece to send to the application layer.
The protocol builds up a connection prior to data transfer using a system called a three-way handshake. This is a way for the two ends of the communication to acknowledge the request and agree upon a method of ensuring data reliability. After the data has been sent, the connection is torn down using a similar four-way handshake.
TCP is the protocol of choice for many of the most popular uses for the internet, including WWW, FTP, SSH, and email. It is safe to say that the internet we know today would not be here without TCP.

UDP: UDP stands for user datagram protocol. It is a popular companion protocol to TCP and is also implemented in the transport layer.
The fundamental difference between UDP and TCP is that UDP offers unreliable data transfer. It does not verify that data has been received on the other end of the connection.
This might sound like a bad thing, and for many purposes, it is. However, it is also extremely important for some functions.
Because it is not required to wait for confirmation that the data was received and forced to

resend data, UDP is much faster than TCP. It does not establish a connection with the remote host, it simply fires off the data to that host and doesn't care if it is accepted or not. Because it is a simple transaction, it is useful for simple communications like querying for network resources. It also doesn't maintain a state, which makes it great for transmitting data from one machine to many real-time clients. This makes it ideal for VOIP, games, and other applications that cannot afford delays

3. Methodology

Defining functions: Functions are defined using the def keyword. After this keyword comes an identifier name for the function, followed by a pair of parentheses which may enclose some names of variables, and by the final colon that ends the line

```
def XX_YY(variable1, varible2):
# block belonging to the function
# End of function
```

Defining local and global variables: Local and global variables can be defined using:

```
x = 50 #Local
global x
```

Defining modules: There are various methods of writing modules, but the simplest way is to create a file with a .py extension that contains functions and variables.

def xx_yy():
aa

Using modules: A module can be imported by another program to make use of its functionality. This is how we can use the Python standard library as well.

import xx_yy

4. Exercises:

Exercise 4.1.2:

```
print ("hello world");
```

```
 while.py
shamim@shamim-HP-ProBook-450-G5:~/Desktop$ python function.py
python: can't open file 'function.py': [Errno 2] No such file o
shamim@shamim-HP-ProBook-450-G5:~/Desktop$ python funtion.py
hello world
shamim@shamim-HP-ProBook-450-G5:~/Desktop$
```

Exercise 4.1.3:

```python
#Python function to calculate the s
#defining the function
def sum (a,b):
    return a+b;

#taking values from the user
a = int(input("Enter a: "))
b = int(input("Enter b: "))

#printing the sum of a and b
print("Sum = ",sum(a,b))
```

```
shamim@shamim-HP-ProBook-450-G5:~/Desktop$ python funtion2.py
Enter a: 10
Enter b: 20
('Sum = ', 30)
shamim@shamim-HP-ProBook-450-G5:~/Desktop$
```

Exercise 4.1.4:

```python
# This function has a variable with
# name same as s.
def f():
    s = "Me too."
    print(s)

# Global scope
s = "I love Geeksforgeeks"
f()
print(s)
```

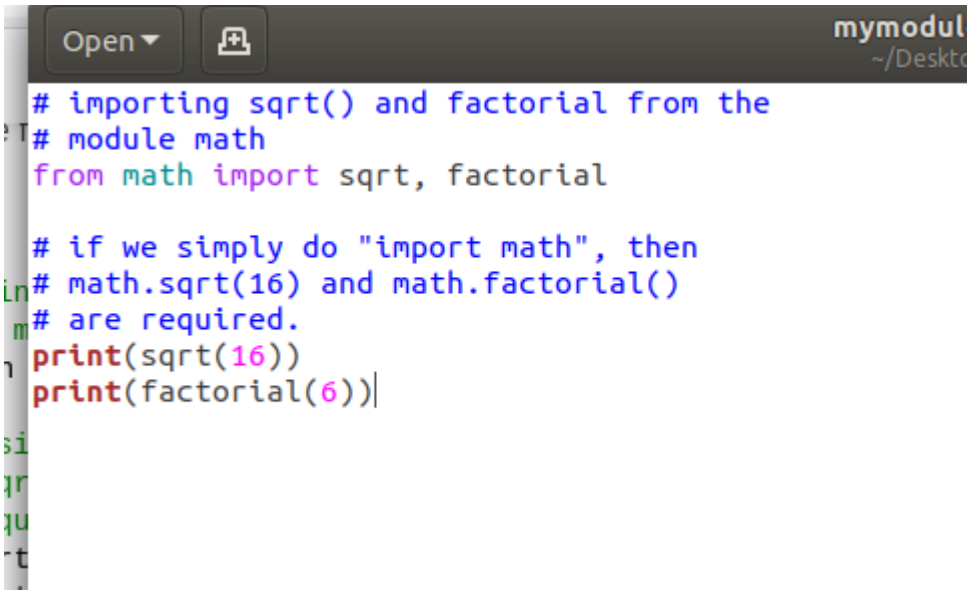
```
write.py
shamim@shamim-HP-ProBook-450-G5:~/Desktop$ python  local.py
Me too.
I love Geeksforgeeks
shamim@shamim-HP-ProBook-450-G5:~/Desktop$
```

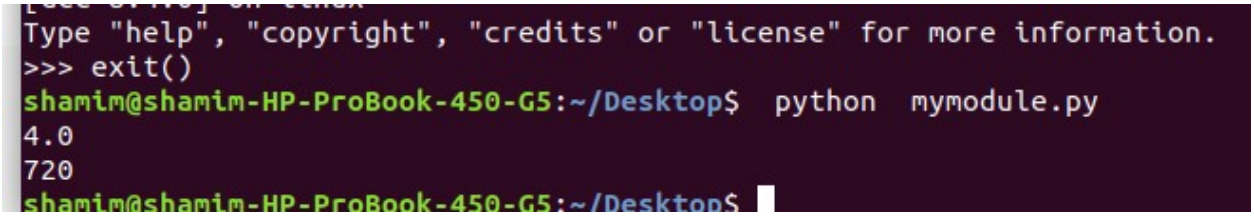
Exercise 4.1.5:

```
x = "awesome"

def myfunc():
  x = "fantastic"
  print("Python is " + x)

myfunc()

print("Python is " + x)
```

```
Enter b: 20
('Sum = ', 30)
shamim@shamim-HP-ProBook-450-G5:~/Desktop$ python global.py
Python is fantastic
Python is awesome
shamim@shamim-HP-ProBook-450-G5:~/Desktop$
```

Exercise 4.1.6:





Section 4.2:

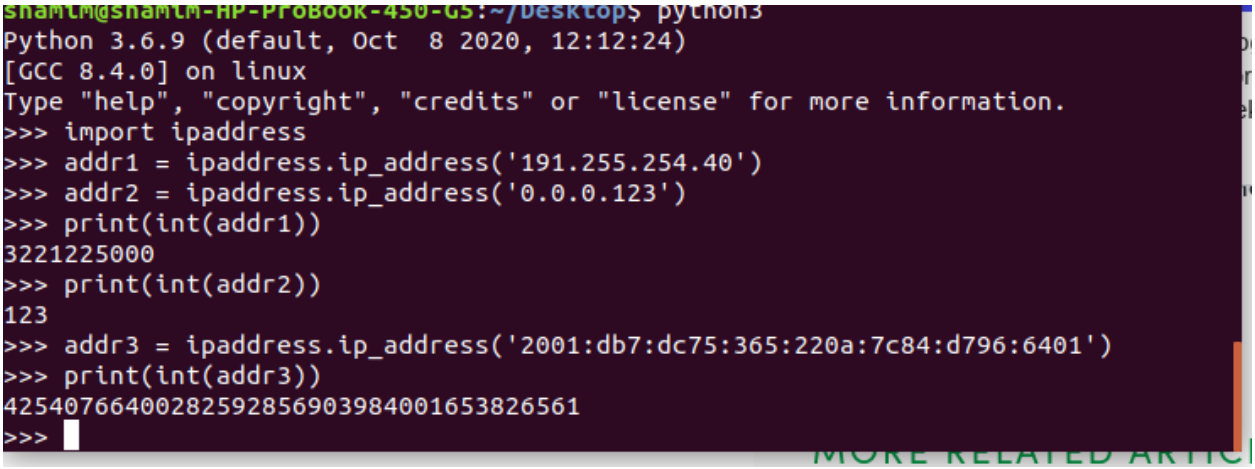
Exercise 4.2.1: Printing your machine's name and IPv4 address

```python
# Python3 code to display hostname and
# IP address

# Importing socket library
import socket

# Function to display hostname and
# IP address
def get_Host_name_IP():
        try:
                host_name = socket.gethostname()
                host_ip = socket.gethostbyname(host_name)
                print("Hostname : ",host_name)
                print("IP : ",host_ip)
        except:
                print("Unable to get Hostname and IP")

# Driver code
get_Host_name_IP() #Function call

#This code is conributed by "Sharad_Bhardwaj".
```

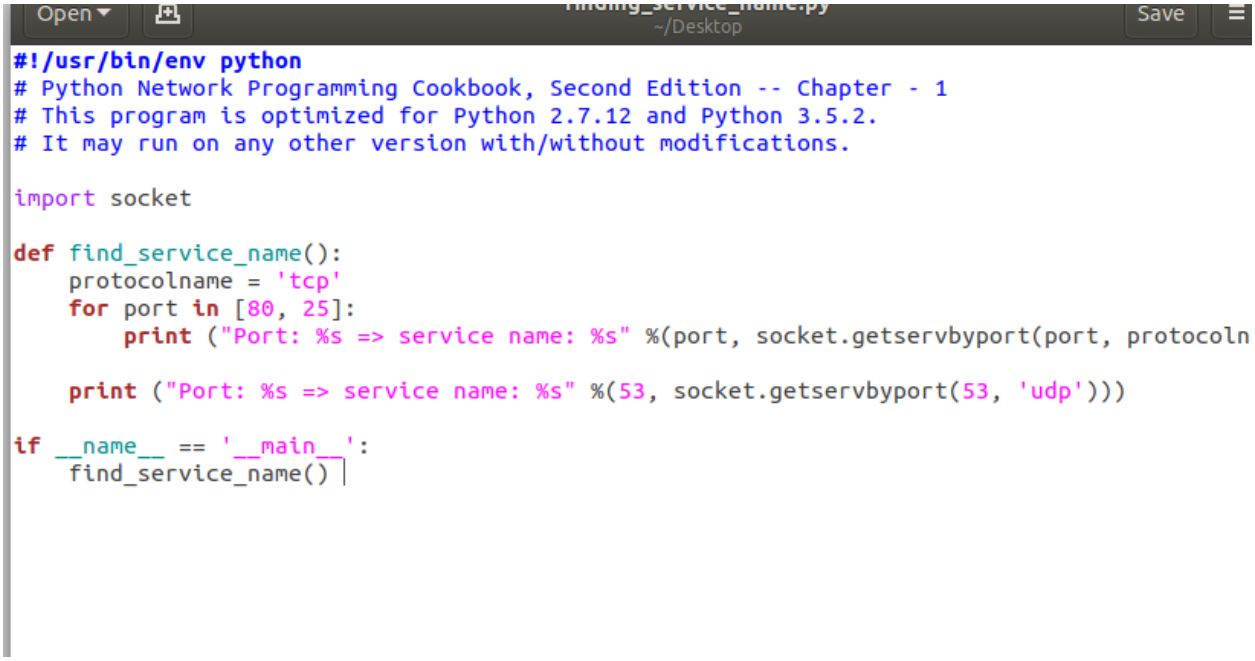
```
 while.py
shamim@shamim-HP-ProBook-450-G5:~/Desktop$ python local_machine_info.py
('Hostname : ', 'shamim-HP-ProBook-450-G5')
('IP : ', '127.0.1.1')
shamim@shamim-HP-ProBook-450-G5:~/Desktop$
```

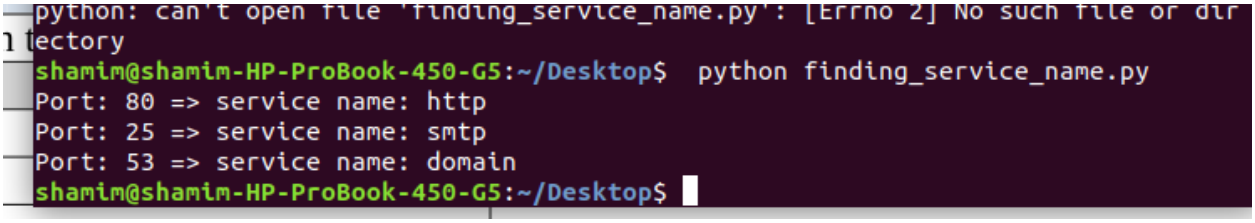
Exercise 4.2.3: Converting an IPv4 address to different formats

```python
# importing the module
import ipaddress

# converting IPv4 address to int
addr1 = ipaddress.ip_address('191.255.254.40')
addr2 = ipaddress.ip_address('0.0.0.123')
print(int(addr1))
print(int(addr2))

# converting IPv6 address to int
addr3 = ipaddress.ip_address('2001:db7:dc75:365:220a:7c84:d796:6401')
print(int(addr3))
```



Exercise 4.2.4: Finding a service name, given the port and protocol





5. Questions:

Question 5.1:

Python Function :

A function is a block of organized, reusable code that is used to perform a single, related action. Functions provide better modularity for your application and a high degree of code reusing.

Python gives you many built-in functions like print(), etc. but you can also create your own functions. These functions are called user-defined functions.

Python Module :

A module allows you to logically organize your Python code. Grouping related code into a module makes the code easier to understand and use. A module is a Python object with arbitrarily named attributes that you can bind and reference.

Simply, a module is a file consisting of Python code. A module can define functions, classes and variables. A module can also include runnable code.

Question 5.2:

Local Variable

Local Variable is defined as a type of variable declared within programming block or subroutines. It can only be used inside the subroutine or code block in which it is declared. The local variable exists until the block of the function is under execution. After that, it will be destroyed automatically.

Example of Local Variable

```
public int add(){
int a =4;
int b=5;
return a+b;
}
```

Global Variable

A Global Variable in the program is a variable defined outside the subroutine or function. It has a global scope means it holds its value throughout the lifetime of the program. Hence, it can be accessed throughout the program by any function defined within the program, unless it is shadowed.

Example:

```
int a =4;
int b=5;
public int add(){
return a+b;
}
```

Question 5.3:

Socket in Computer Network

A socket is one endpoint of a two way communication link between two programs running on the network. The socket mechanism provides a means of inter-process communication (IPC) by establishing named contact points between which the communication take place.

Like ‘Pipe’ is used to create pipes and sockets is created using ‘socket’ system call. The socket provides bidirectional FIFO Communication facility over the network. A socket connecting to the network is created at each end of the communication. Each socket has a specific address. This address is composed of an IP address and a port number.
Socket are generally employed in client server applications. The server creates a socket, attaches it to a network port addresses then waits for the client to contact it. The client creates a socket and then attempts to connect to the server socket. When the connection is established, transfer of data takes place.

Datagram Socket :
This is a type of network which has connection less point for sending and receiving packets. It is similar to mailbox. The letters (data) posted into the box are collected and delivered (transmitted) to a letterbox (receiving socket).

- Stream Socket

  In Computer operating system, a stream socket is type of **interprocess communications** socket or network socket which provides a connection-oriented, sequenced, and unique flow of data without record boundaries with well defined mechanisms for creating and destroying connections and for detecting errors. It is similar to phone. A connection is established between the phones (two ends) and a conversation (transfer of data) takes place.