

Practical 1:

Aim: LINUX COMMANDS: EXPLORING THE RASPBIAN

LINUX COMMANDS

Here are some fundamental and common Linux commands with example usage:

FILESYSTEM commands:

- **LS** : The ls command lists the content of the current directory (or one that is specified). It can be used with the -l flag to display additional information (permissions, owner, group, size, date and timestamp of last edit) about each file and directory in a list format. The -a flag allows you to view files beginning with . (i.e. dotfiles).
- **CD** : Using cd changes the current directory to the one specified. You can use relative (i.e. cd directoryA) or absolute (i.e. cd /home/pi/directoryA) paths.
- **PWD** : The pwd command displays the name of the present working directory: on a Raspberry Pi, entering pwd will output something like /home/pi.
- **MKDIR** : You can use mkdir to create a new directory, e.g. mkdir newDir would create the directory newDir in the present working directory.
- **RMDIR** : To remove empty directories, use rmdir. So, for example, rmdir oldDir will remove the directory oldDir only if it is empty.
- **RM** : The command rm removes the specified file (or recursively from a directory when used with -r). Be careful with this command: files deleted in this way are mostly gone for good!
- **CP** : Using cp makes a copy of a file and places it at the specified location (this is similar to copying and pasting). For example, cp ~/fileA /home/otherUser/would copy the file fileA from your home directory to that of the user otherUser (assuming you have permission to copy it there). This command can either take FILE FILE (cp fileA fileB), FILE

`DIR(cp fileA /directoryB/)` or `-r DIR DIR` (which recursively copies the contents of directories) as arguments.

- **MV** : The mv command moves a file and places it at the specified location (so where cp performs a 'copy-paste', mv performs a 'cut-paste'). The usage is similar to cp. So `mv ~/fileA /home/otherUser/` would move the file fileA from your home directory to that of the user otherUser. This command can either take FILE FILE (`mv fileA fileB`), FILE DIR (`mv fileA /directoryB/`) or DIR DIR (`mv /directoryB /directoryC`) as arguments. This command is also useful as a method to rename files and directories after they've been created.
- **TOUCH** : The command touch sets the last modified time-stamp of the specified file(s) or creates it if it does not already exist.
- **CAT** : You can use cat to list the contents of file(s), e.g. `cat thisFile` will display the contents of thisFile. Can be used to list the contents of multiple files, i.e. `cat *.txt` will list the contents of all .txt files in the current directory.
- **HEAD** : The head command displays the beginning of a file. Can be used with `-n` to specify the number of lines to show (by default ten), or with `-c` to specify the number of bytes.
- **TAIL** : The opposite of head, tail displays the end of a file. The starting point in the file can be specified either through `-b` for 512 byte blocks, `-c` for bytes, or `-n` for number of lines.
- **CHMOD** : You would normally use chmod to change the permissions for a file. The chmod command can use symbols u (user that owns the file), g (the files group) , and o (other users) and the permissions r (read), w (write), and x (execute). Using `chmod u+x *filename*` will add execute permission for the owner of the file.
- **CHOWN** : The chown command changes the user and/or group that owns a file. It normally needs to be run as root using sudo e.g. `sudo chown pi:root *filename*` will change the owner to pi and the group to root.
- **SSH** : ssh denotes the secure shell. Connect to another computer using an encrypted network connection. For more details see SSH (secure shell)

- SCP : The scp command copies a file from one computer to another using ssh. For more details see SCP (secure copy)
- SUDO : The sudo command enables you to run a command as a superuser, or another user. Use sudo -s for a superuser shell. For more details see Root user / sudo
- DD : The dd command copies a file converting the file as specified. It is often used to copy an entire disk to a single file or back again. So, for example, dd if=/dev/sdd of=backup.img will create a backup image from an SD card or USB disk drive at /dev/sdd. Make sure to use the correct drive when copying an image to the SD card as it can overwrite the entire disk.
- DF : Use df to display the disk space available and used on the mounted filesystems. Use df -h to see the output in a human-readable format using M for MBs rather than showing number of bytes.
- UNZIP : The unzip command extracts the files from a compressed zip file.
- TAR : Use tar to store or extract files from a tape archive file. It can also reduce the space required by compressing the file similar to a zip file. To create a compressed file, use tar -cvzf *filename.tar.gz* *directory/* To extract the contents of a file, use tar -xvzf *filename.tar.gz*
- ssinput for another command. The pipe symbol is a vertical line |. For example, to only show the first ten entries of the ls command it can be piped through the head command ls | head
- TREE : Use the tree command to show a directory and all subdirectories and files indented as a tree structure. & Run a command in the background with &, freeing up the shell for future commands.
- WGET : Download a file from the web directly to the computer with wget. So wget <https://www.raspberrypi.org/documentation/linux/usage/commands.md> will download this file to your computer as commands.md
- CURL : Use curl to download or upload a file to/from a server. By default, it will output the file contents of the file to the screen.

- **MAN** : Show the manual page for a file with man. To find out more, run man man to view the manual page of the man command.

SEARCH COMMANDS

- **GREP** : Use grep to search inside files for certain search patterns. For example, grep "search" *.txt will look in all the files in the current directory ending with .txt for the string search. The find command searches a directory and subdirectories for files matching certain patterns.
- **WHEREIS** : Use whereis to find the location of a command. It looks through standard program locations until it finds the requested command.

NETWORKING COMMANDS:

- **PING** : The ping utility is usually used to check if communication can be made with another host. It can be used with default settings by just specifying a hostname (e.g. ping raspberrypi.org) or an IP address (e.g. ping 8.8.8.8). It can specify the number of packets to send with the -c flag.
- **NMAP** : nmap is a network exploration and scanning tool. It can return port and OS information about a host or a range of hosts. Running just nmap will display the options available as well as example usage.
- **HOSTNAME** : The hostname command displays the current hostname of the system. A privileged (super) user can set the hostname to a new one by supplying it as an argument (e.g. hostname new-host).
- **IFCONFIG** : Use ifconfig to display the network configuration details for the interfaces on the current system when run without any arguments (i.e. ifconfig). By supplying the command with the name of an interface (e.g. eth0 or lo) you can then alter the configuration.

Practical 2:

Aim : To Perform Bash Commands

1. Create a bash file with following structure
f1/f2/t1

GNU nano 2.0.6

File: file1

```
#!/bin/bash
mkdir f1
cd f1
mkdir f2
cd f2
touch t1
```

Rishabh:~ rishabhghodke\$./file1

Rishabh:~ rishabhghodke\$ chmod +x file1

Rishabh:~ rishabhghodke\$ ls

#.profile#	Library	addition
AndroidStudioProjects	Movies	enthought
Applications	Music	f1
Desktop	Pictures	file1
Documents	Projects	iCloud Drive (Archive)
Downloads	Public	s.py

2.Create a bash file with following

- a. man pwd
- b. man touch
- c. man cp

```
Rishabh:~ rishabhghodke$ nano file2
Rishabh:~ rishabhghodke$ chmod + file2
Rishabh:~ rishabhghodke$ chmod +x file2
Rishabh:~ rishabhghodke$ ./file2

[1]+  Stopped                  ./file2
Rishabh:~ rishabhghodke$
```

PWD(1)	BSD General Commands Manual	PWD(1)
NAME		
pwd -- return working directory name		
SYNOPSIS		
pwd [-L -P]		
DESCRIPTION		
The pwd utility writes the absolute pathname of the current working directory to the standard output.		

3.Create a bash file to perform Arithmetic Operations

```
Rishabh:- rishabhghodke$ nano file3
Rishabh:- rishabhghodke$ chmod +x file3
Rishabh:- rishabhghodke$ ./file3
Rishabh:- rishabhghodke$ cd f1
Rishabh:f1 rishabhghodke$ ls
f2      t1
Rishabh:f1 rishabhghodke$ █
```

```
GNU nano 2.0.6          File: file4

#!/bin/bash
echo "arithmetic operation"
echo "First number"
read a
echo "Second number"
read b
c=`expr $a + $b`
d=$((a-b))
echo "Addition is $c"
echo "Subtraction is $d"
█
```

```
[Rishabh:f1 rishabhghodke$ ./file4
arithmetic operation
First number
4
Second number
8
Addition is 12
Subtraction is -4
Rishabh:f1 rishabhghodke$ ]
```

4.Create a bash file to check weather numbers are same

```
#!/bin/bash
echo "arithmetic operation"
echo "First number"
read a
echo "Second number"
read b
if [ $a -eq $b ]
then
echo "same number"
else
echo "different number"
fi]
```

```
[Rishabh:f1 rishabhghodke$ ./file5
arithmetic operation
First number
6
Second number
9
different number
[Rishabh:f1 rishabhghodke$ ./file5
arithmetic operation
First number
6
Second number
6
same number
Rishabh:f1 rishabhghodke$ ]
```

5.Create a bash file to perform Multiplication

```
#!/bin/bash
echo "arithmetic operation"
echo "First number"
read a
echo "Second number"
read b
c= `expr "$a" \* "$b"`
echo "multiplication is $c"
[
```

```
[Rishabh:f1 rishabhghodke$ ./file6
arithmetic operation
First number
8
Second number
5
multiplication is 40
Rishabh:f1 rishabhghodke$ ]
```

6.Create a bash file to perform switch case operation

```
#!/bin/bash
echo "switch case"
echo "choose any random number from 1 to 5"
read no;
case $no in
    1)echo "good boy" ;;
    2)echo "hello World" ;;
    3)echo "hi" ;;
    4)echo "bye" ;;
    5)echo "talk to you later" ;;
    *)echo "good morning" ;;
esac
```

```
[Rishabh:f1 rishabhghodke$ ./file7
switch case
choose any random number from 1 to 5
5
talk to you later
Rishabh:f1 rishabhghodke$ ]
```

2.Create Bash File to create Touch file m1,m2,m3,m4

```
#!/bin/bash
touch m1
touch m2
touch m3
touch m4
```

```
Rishabh:f1 rishabhghodke$ chmod +x file9
Rishabh:f1 rishabhghodke$ ./file9
Rishabh:f1 rishabhghodke$ ls
f2      file5    file7    file9    m2        m4
file4    file6    file8    m1        m3        t1
Rishabh:f1 rishabhghodke$ █
```

3.Create Bash File to create cat file m1,m2,m3,m4

```
#!/bin/bash  
cat m1  
cat m2  
cat m3  
cat m4
```

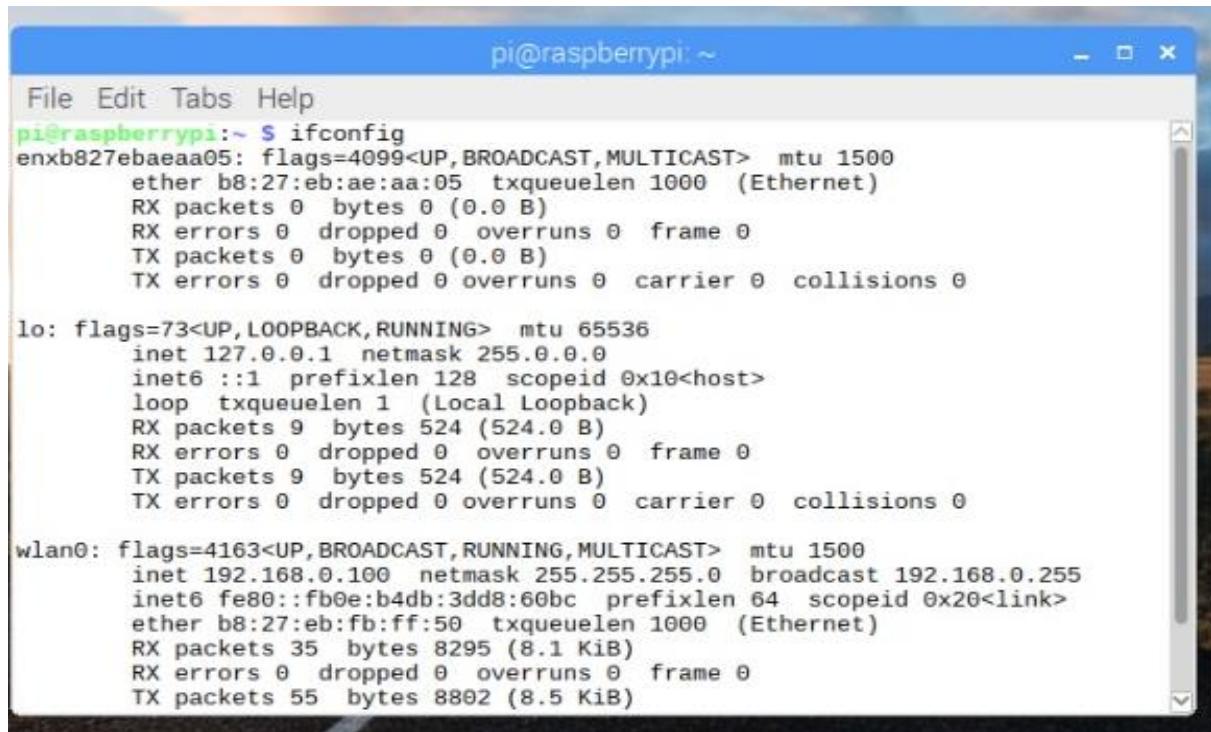
```
[Rishabh:f1 rishabhghodke$ chmod +x file10  
[Rishabh:f1 rishabhghodke$ ./file10  
[Rishabh:f1 rishabhghodke$ ls  
f2      file4    file6    file8    m1      m3      t1  
file10   file5    file7    file9    m2      m4  
Rishabh:f1 rishabhghodke$ █
```

Practical 3:

Aim : Learn to ssh into a raspberry pi using putty from a different computer on the network.

Steps for transferring file from windows to raspberry:

Step 1: Find out the IP address of raspberry - ifconfig



A terminal window titled "pi@raspberrypi: ~" displaying the output of the "ifconfig" command. The output shows three network interfaces: enxb827eba05 (Ethernet), lo (loopback), and wlan0 (wireless). The wlan0 interface is connected to the network with an IP of 192.168.0.100.

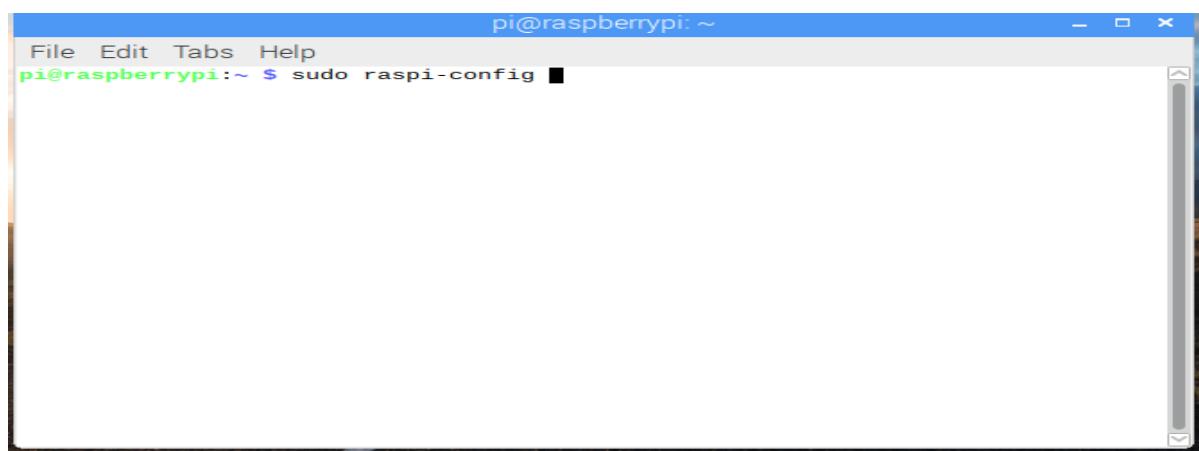
```
File Edit Tabs Help
pi@raspberrypi:~ $ ifconfig
enxb827eba05: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
    ether b8:27:eb:aa:05 txqueuelen 1000 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
        inet6 ::1 prefixlen 128 scopeid 0x10<host>
            loop txqueuelen 1 (Local Loopback)
            RX packets 9 bytes 524 (524.0 B)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 9 bytes 524 (524.0 B)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.0.100 netmask 255.255.255.0 broadcast 192.168.0.255
        inet6 fe80::fb0e:b4db:3dd8:60bc prefixlen 64 scopeid 0x20<link>
            ether b8:27:eb:fb:ff:50 txqueuelen 1000 (Ethernet)
            RX packets 35 bytes 8295 (8.1 KiB)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 55 bytes 8802 (8.5 KiB)
```

Step 2: Make the setting of SSH connection in raspberry

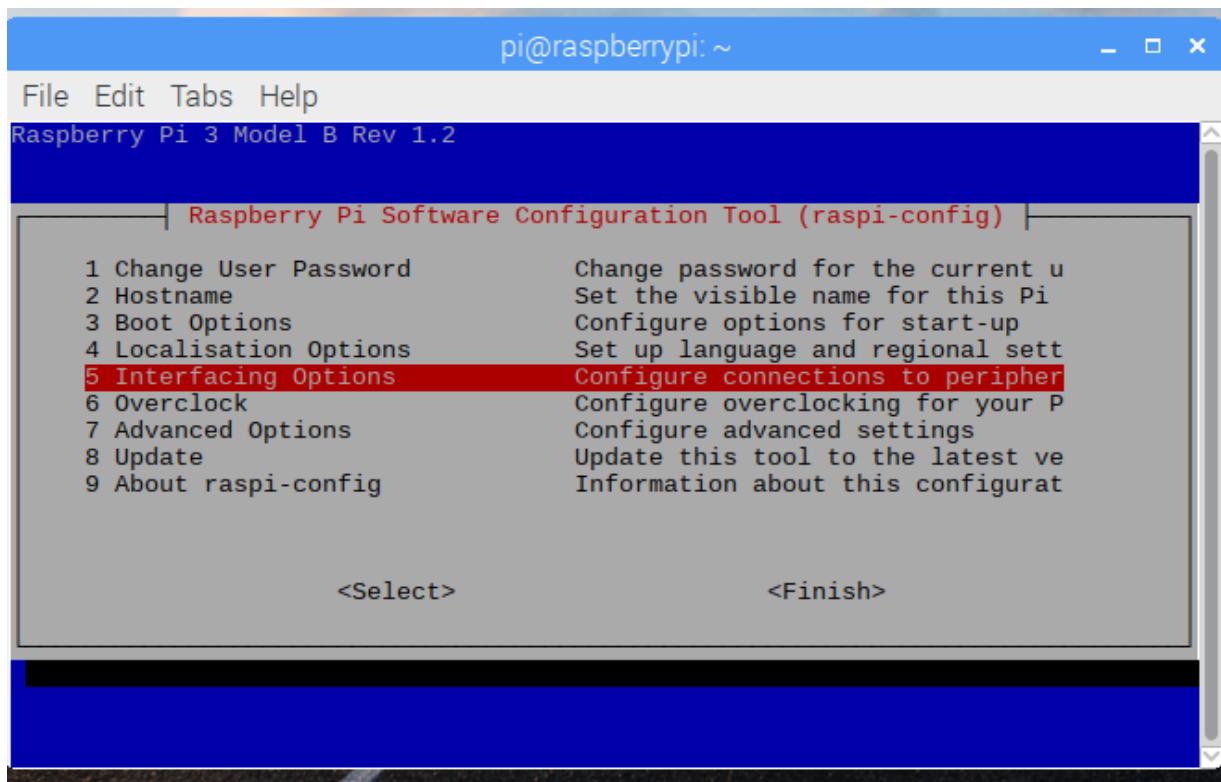
Type – sudo raspi -config



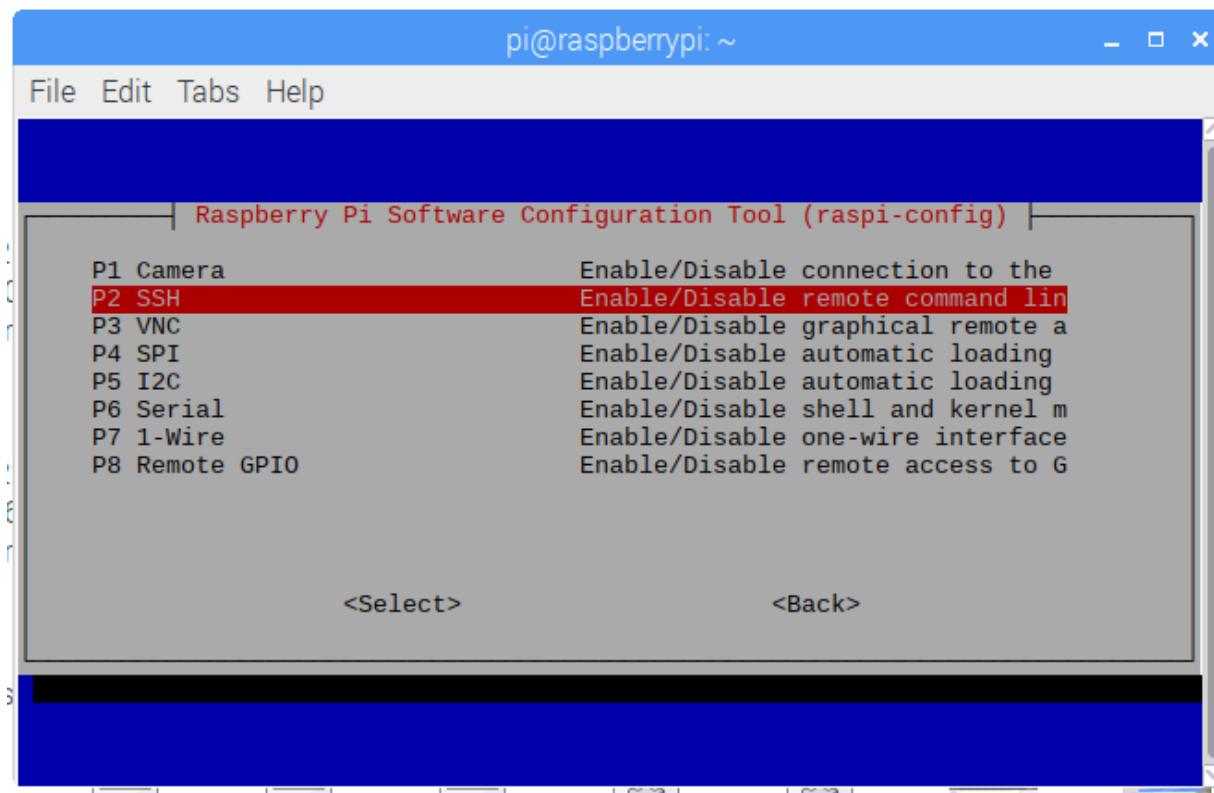
A terminal window titled "pi@raspberrypi: ~" displaying the command "sudo raspi-config". The window is mostly blank, indicating the configuration process is running.

```
File Edit Tabs Help
pi@raspberrypi:~ $ sudo raspi-config ■
```

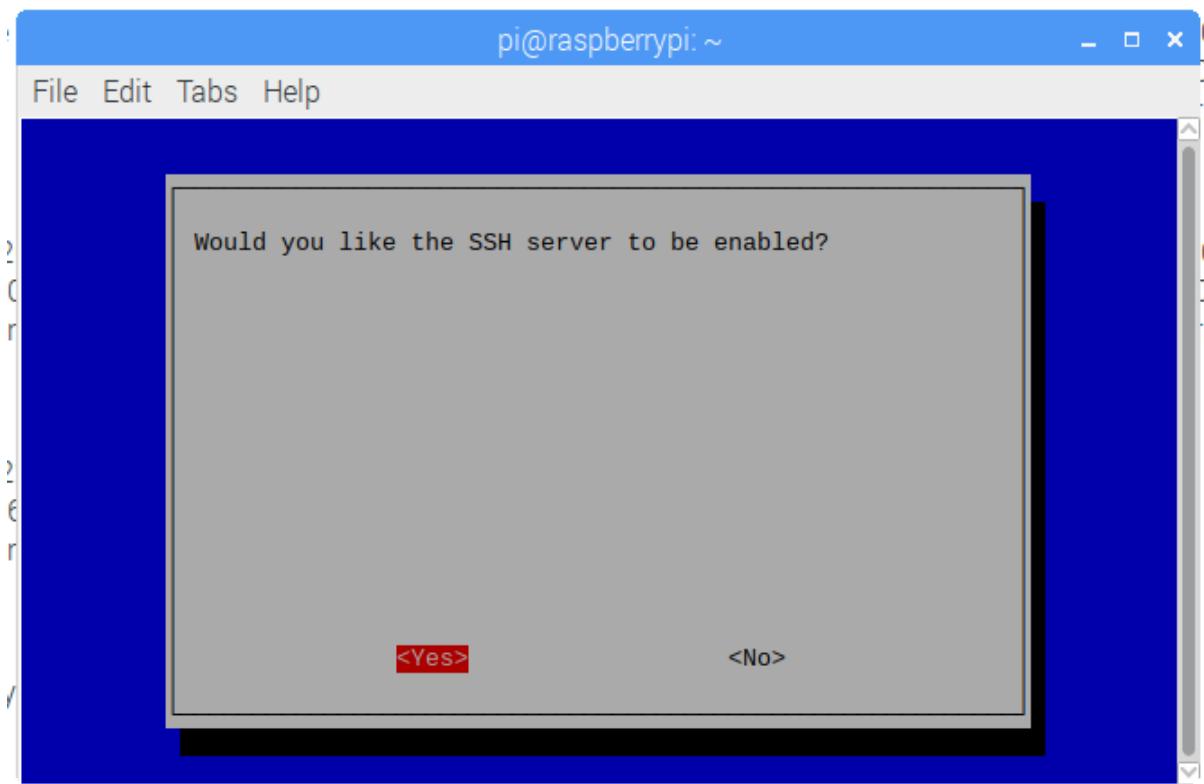
Select Interfacing Options



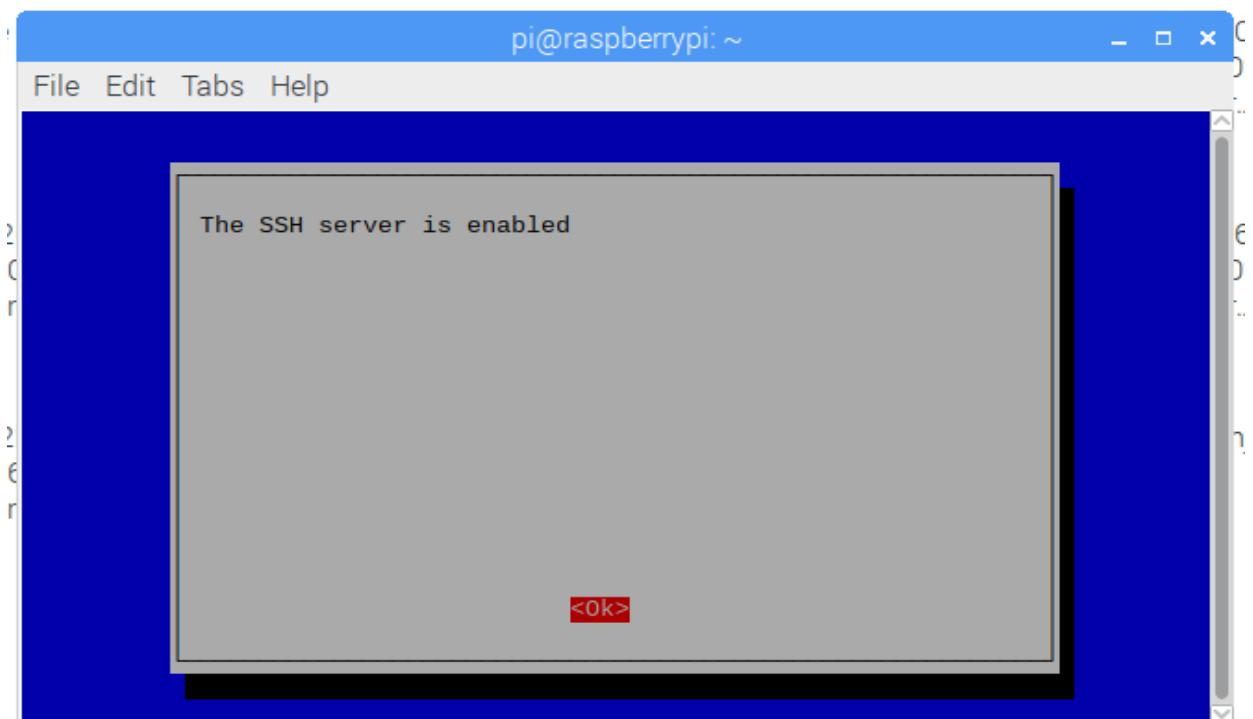
Select SSH



Enable SSH



Then Click on OK



Step 3: Download putty server on your desktop

Screenshot of a web browser showing the download page for PuTTY version 0.70. The page includes links for MSI installers (32-bit and 64-bit), a Unix source archive, and alternative binary files.

Package files

You probably want one of these. They include all the PuTTY utilities.
(Not sure whether you want the 32-bit or the 64-bit version? Read the [FAQ entry](#).)

MSI ('Windows Installer')

32-bit:	putty-0.70-installer.msi	(or by FTP)	(signature)
64-bit:	putty-64bit-0.70-installer.msi	(or by FTP)	(signature)

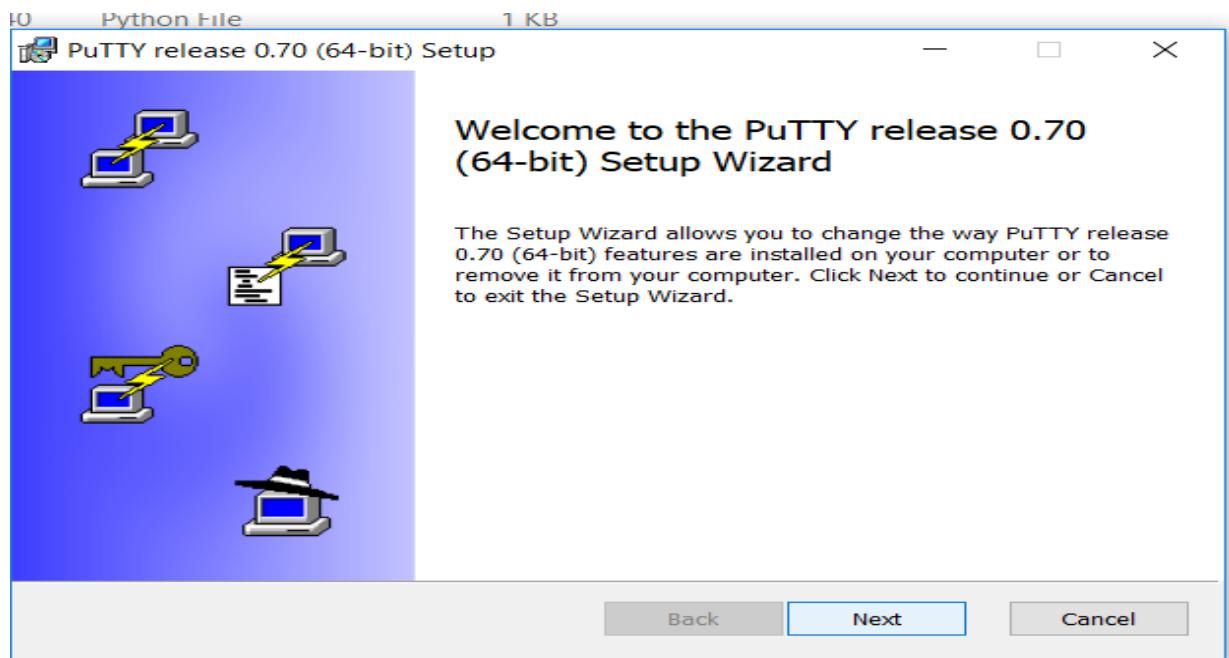
Unix source archive

.tar.gz:	putty-0.70.tar.gz	(or by FTP)	(signature)
----------	-----------------------------------	------------------------------	-------------

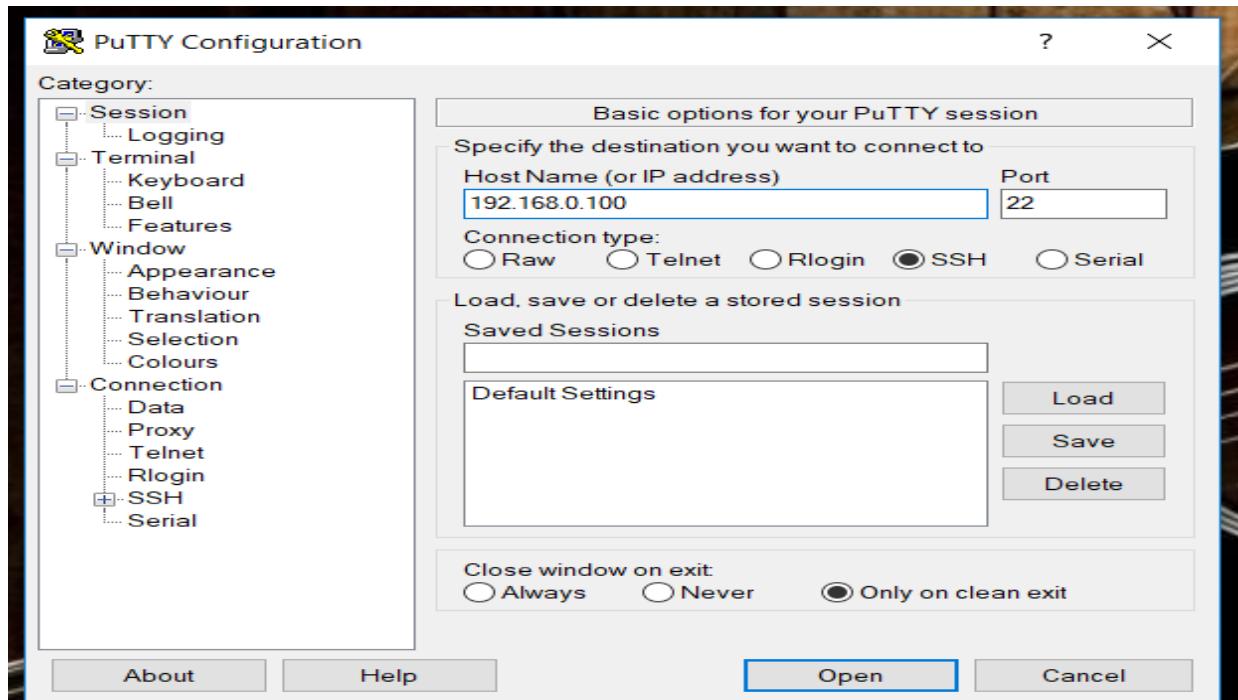
Alternative binary files

The installer packages above will provide all of these (except PuTTYtel), but you can download them one by one if you prefer.
<https://theearth.li/~sgtatham/putty/latest/w64/putty-64bit-0.70-installer.msi> ... version? Read the [FAQ entry](#).

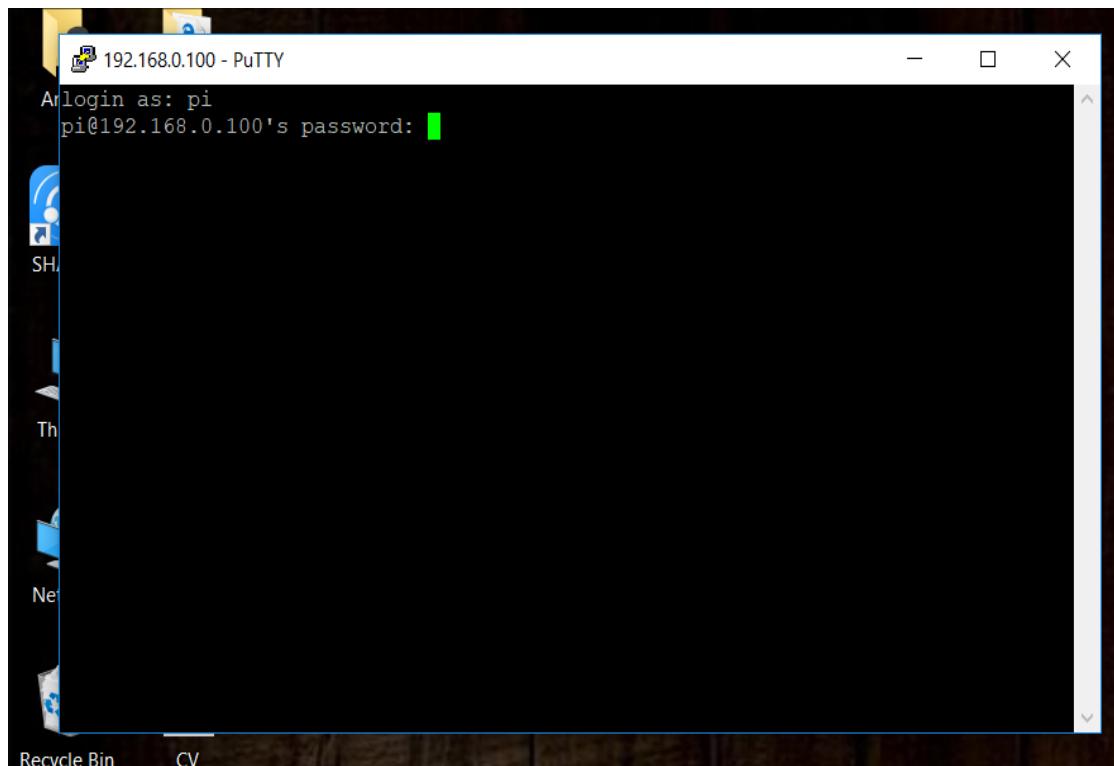
Step 4: Install putty server



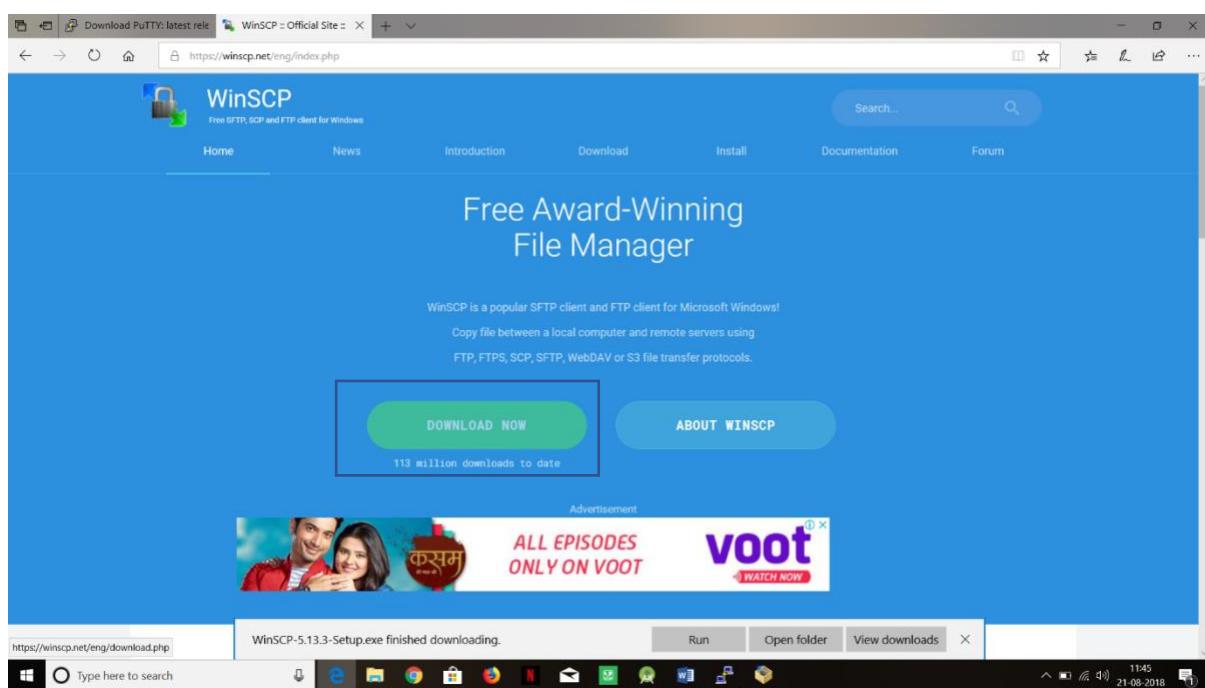
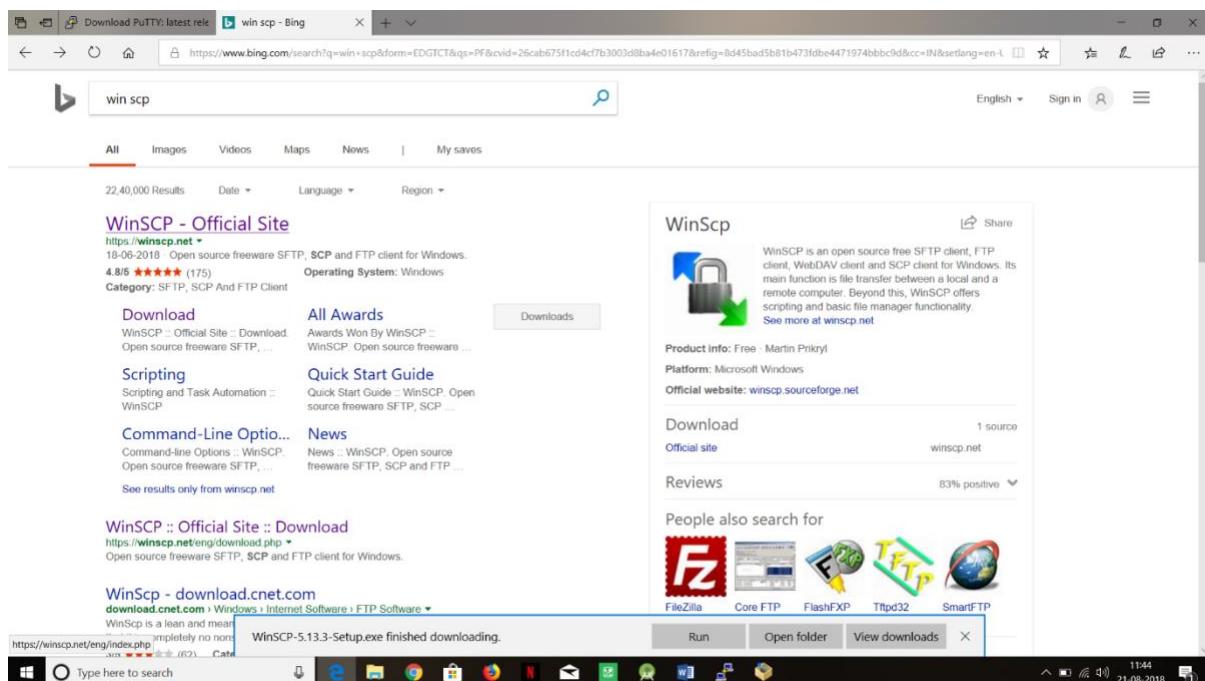
Step 5: Open putty server, then write the Hostname or IP address of raspberry



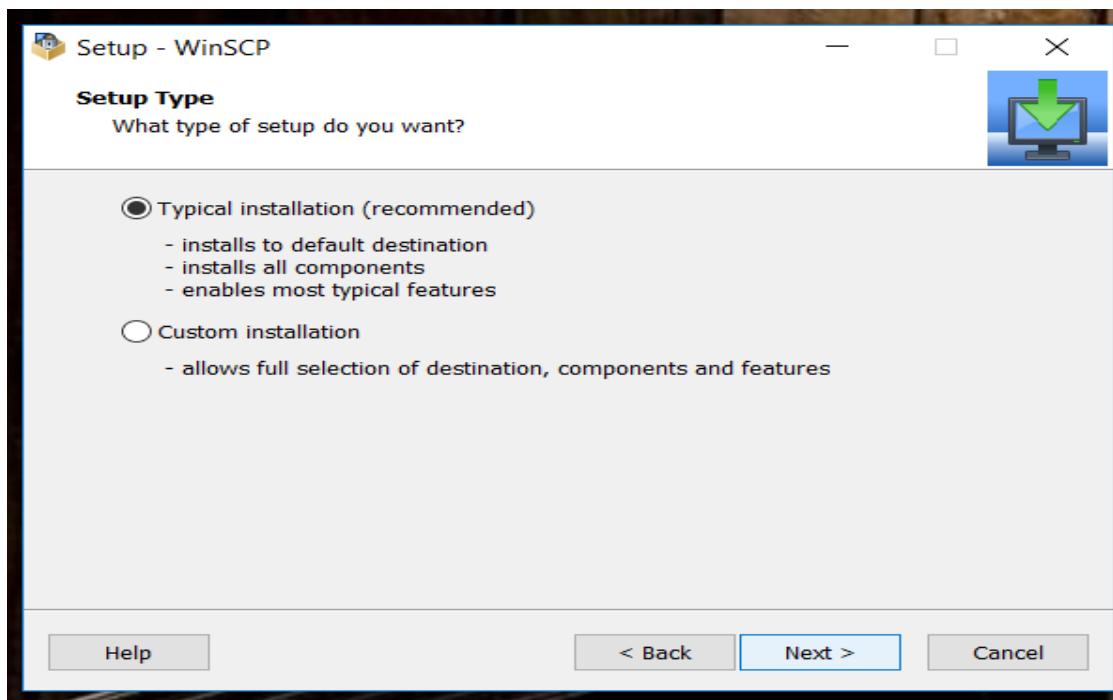
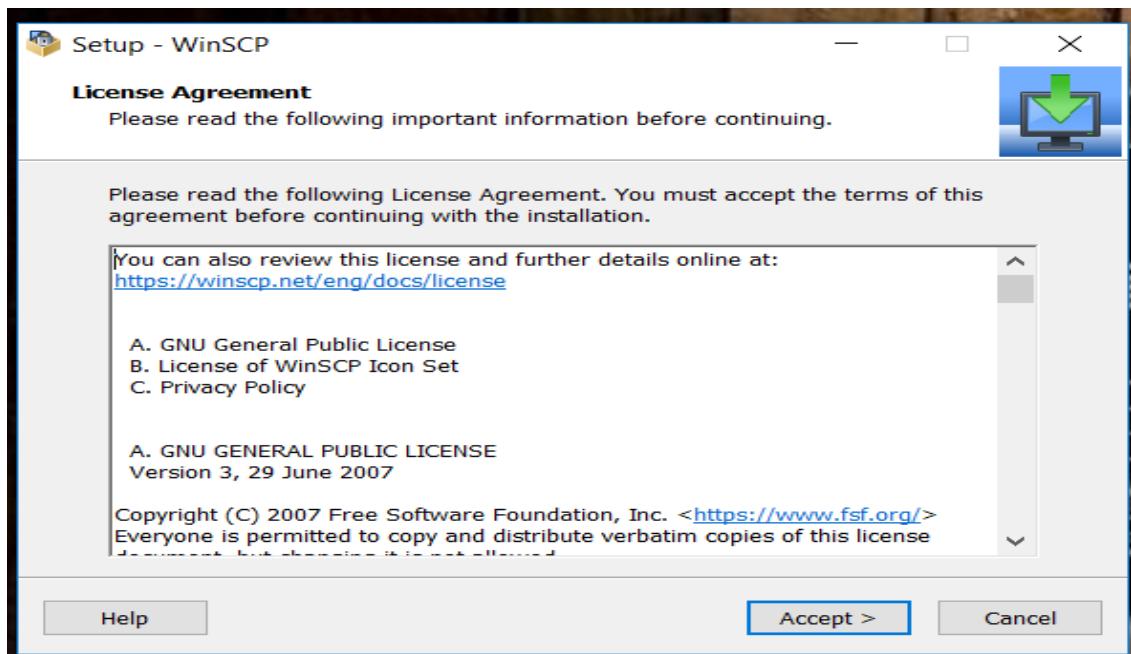
Step 6: Once it is connected it will ask for login & password of raspberry which is “pi” & “raspberry” respectively

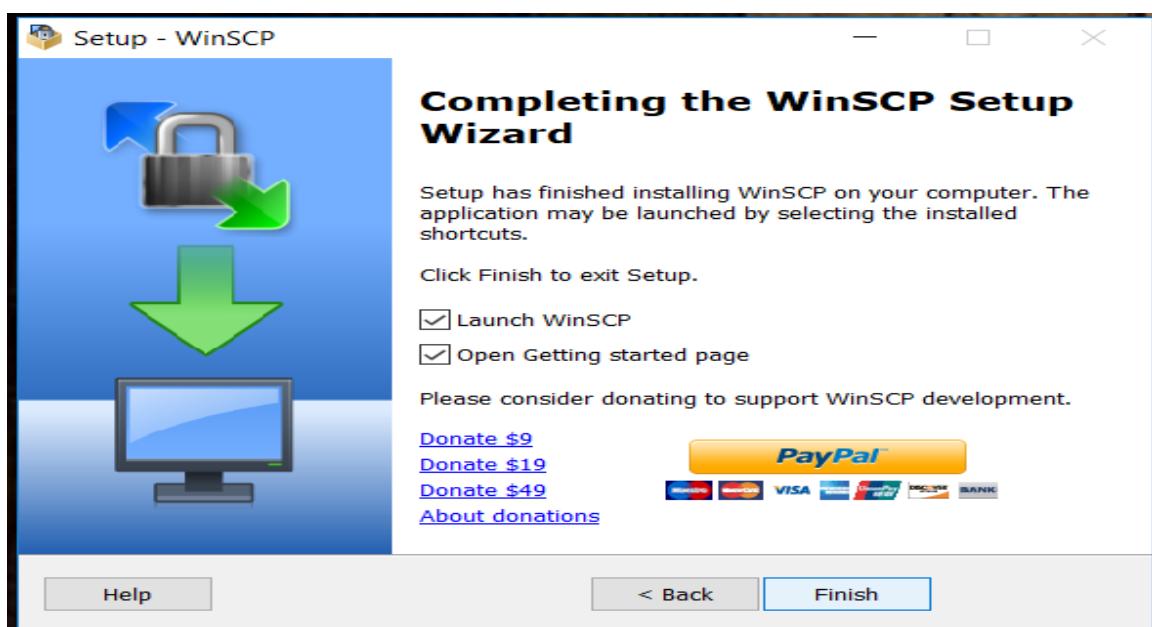
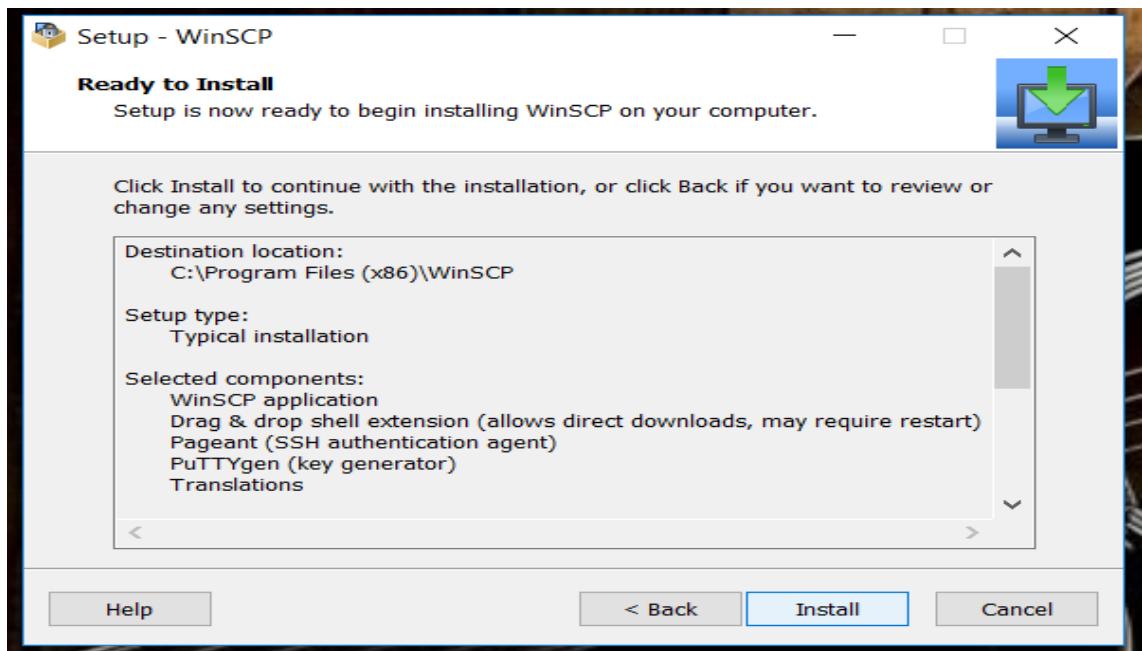


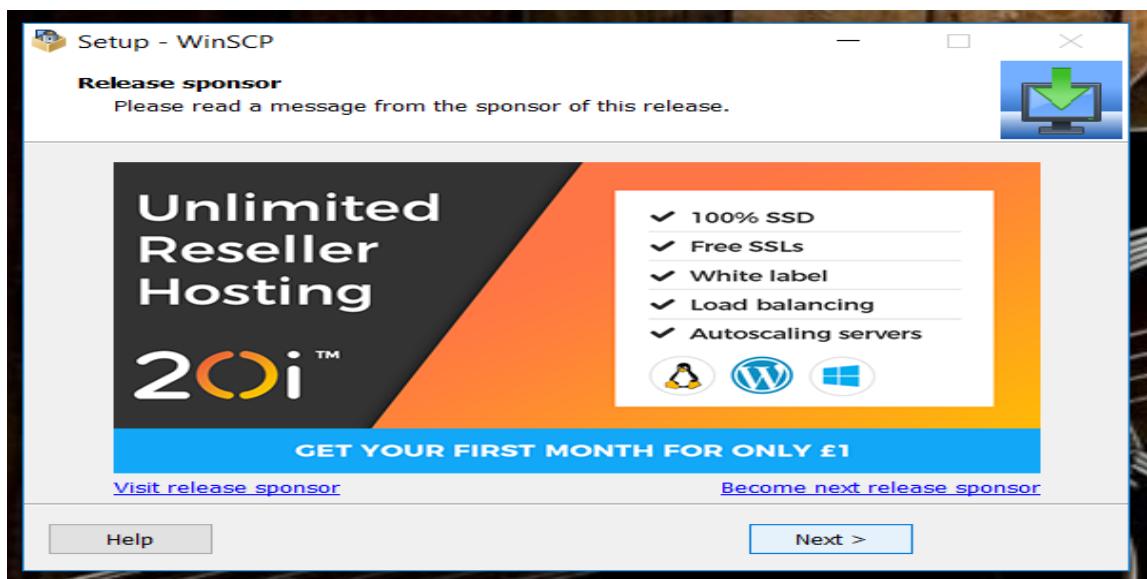
Step 7: Download WinSCP to transfer the file from your desktop to raspberry & vice versa



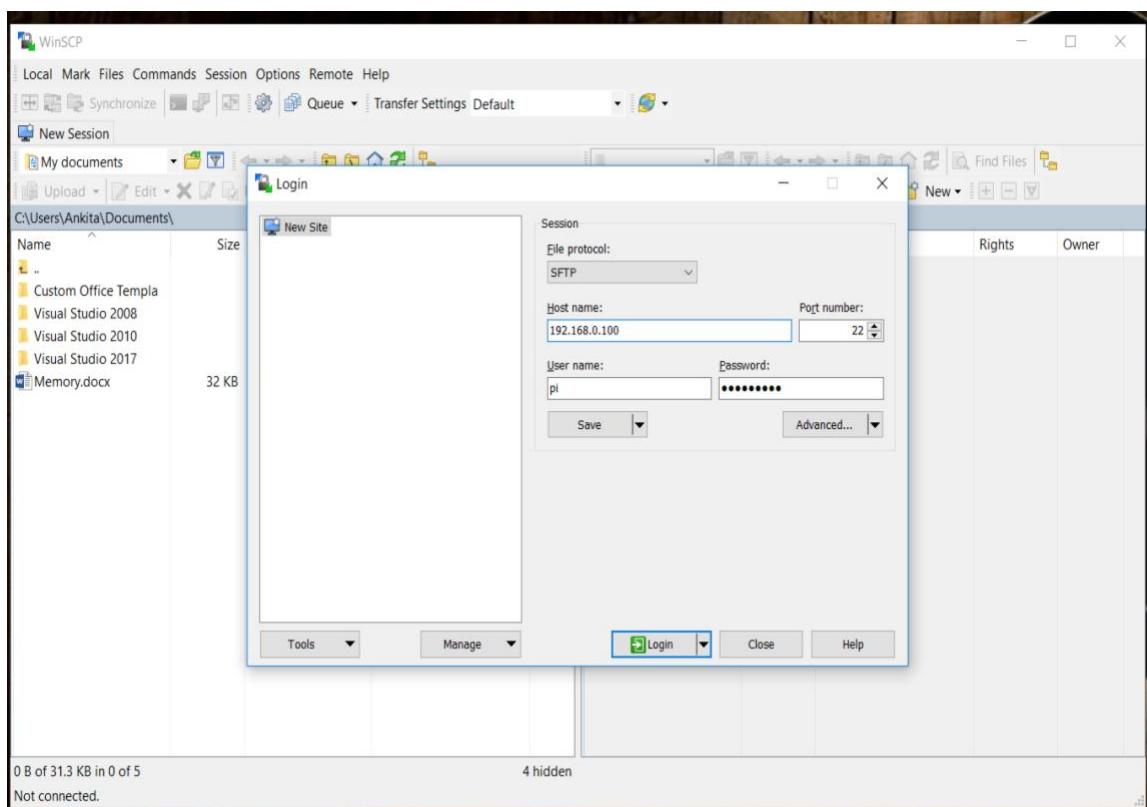
Step 8: Install WinSCP



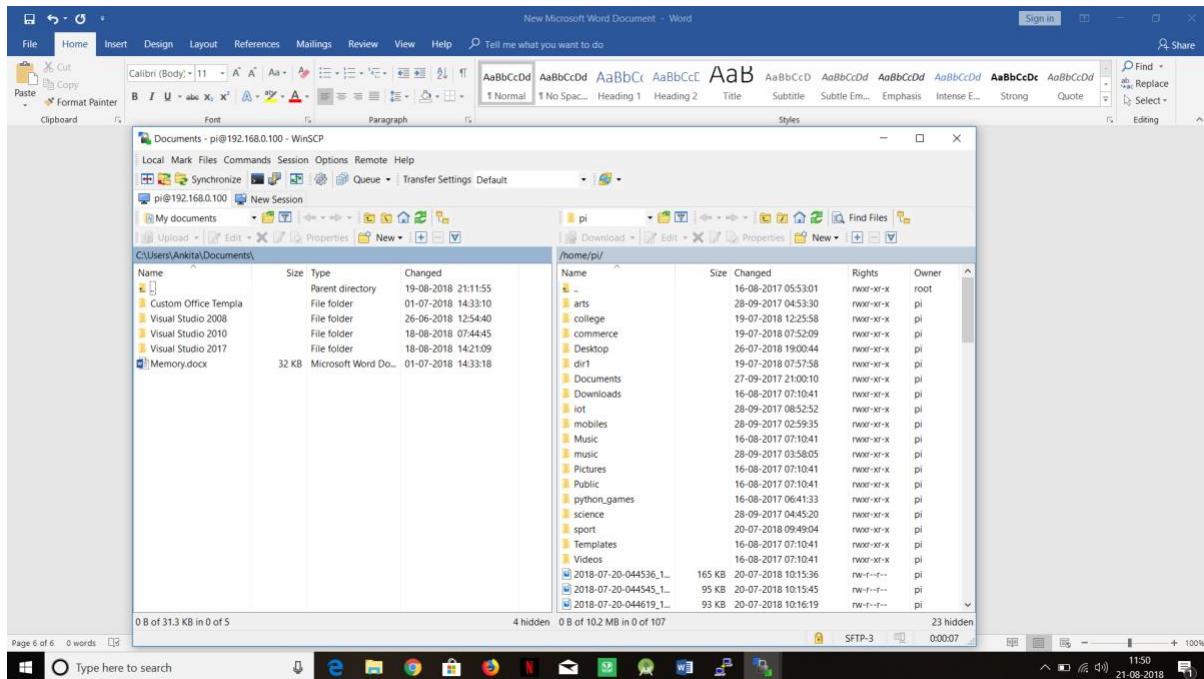




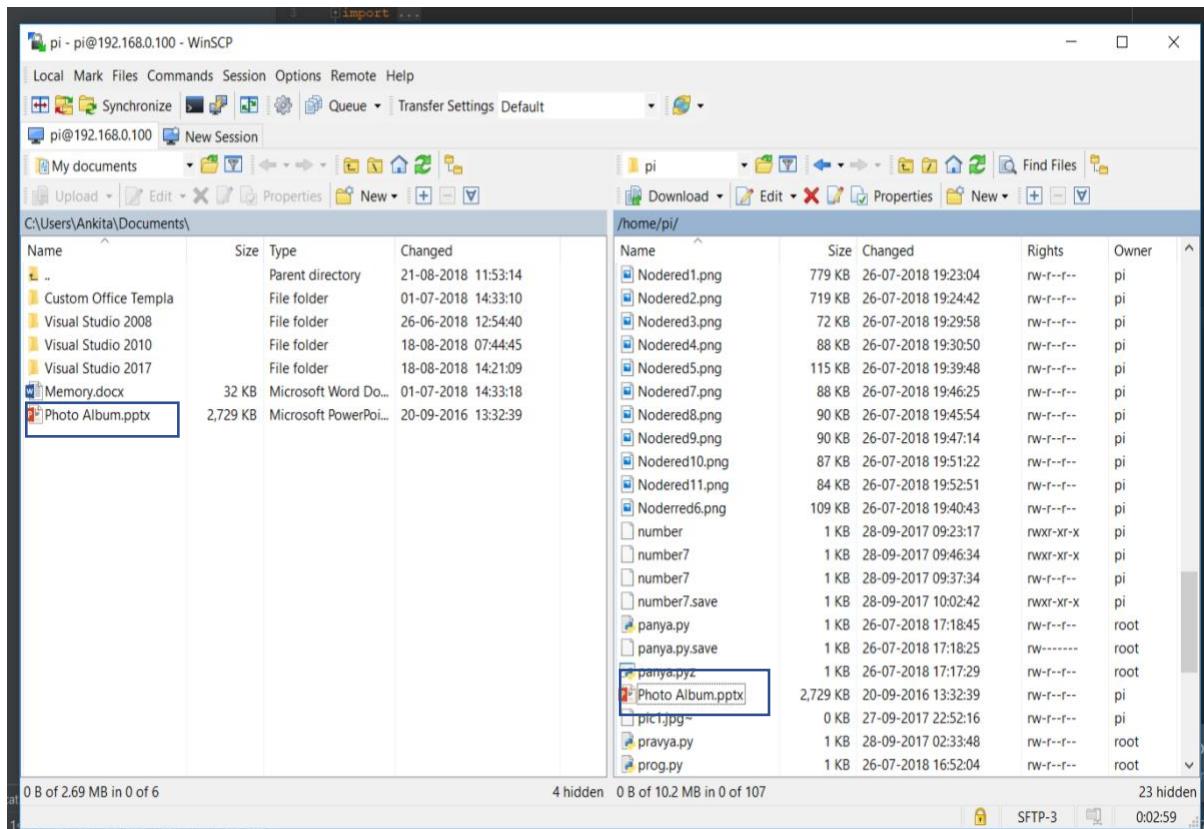
Step 9: Open WinSCP then enter the IP address, username & password



Step 10: You can view two screens on desktop first is of desktop & other is of raspberry



Step 11: Now you can easily transfer the files from one end to another end

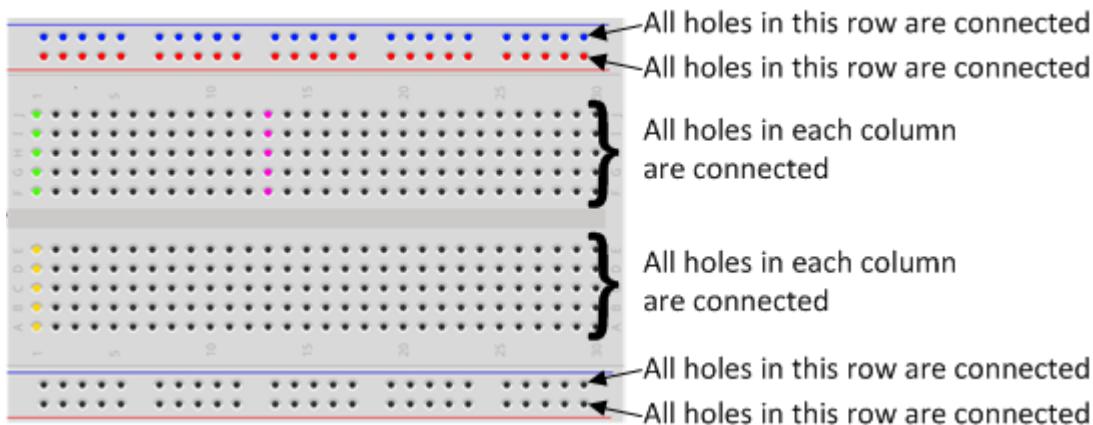


Practical 4:

AIM:GPIO : Light The LED With Python on Breadboard.

The Breadboard

The breadboard is a way of connecting electronic components to each other without having to solder them together. The holes on the breadboard are connected in a pattern.



With the breadboard in the CamJamEduKit, the top row of holes are all connected together – marked with red dots. And so are the second rows of holes – marked with blue dots. The same goes for the two rows of holes at the bottom of the breadboard.

In the middle, the columns of wires are connected together with a break in the middle. So for.

The LED

LED stands for Light Emitting Diode, and glows when electricity is passed through it.

When you pick up the LED, you will notice that one leg is longer than the other. The longer leg (known as the ‘anode’), is always connected to the positive supply of the circuit. The shorter leg (known as the ‘cathode’) is connected to the negative side of the power supply, known as ‘ground’.

The Resistor

You must **ALWAYS** use resistors to connect LEDs up to the GPIO pins of the Raspberry Pi. The Raspberry Pi can only supply a small current (about 60mA). The LEDs will want to draw more, and if allowed to they will burn out the

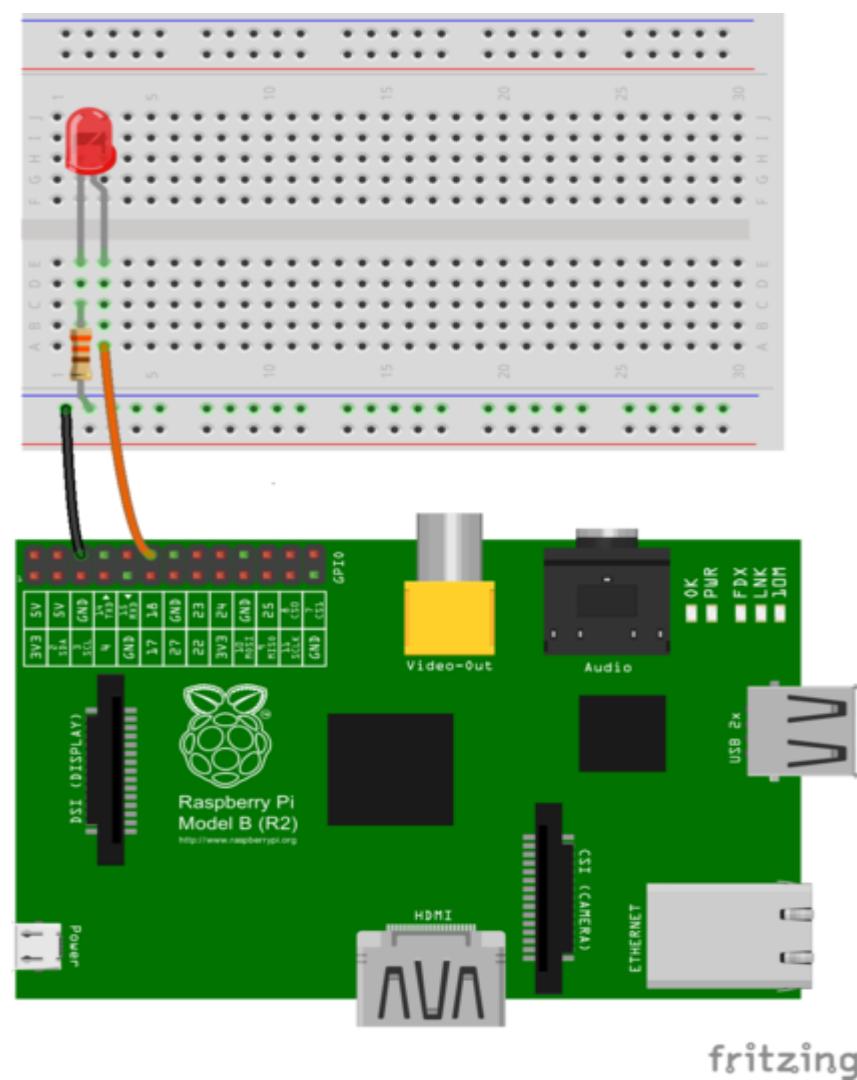
Raspberry Pi. Therefore putting the resistors in the circuit will ensure that only this small current will flow and the Pi will not be damaged.

Jumper Wires

Jumper wires are used on breadboards to ‘jump’ from one connection to another. The ones you will be using in this circuit have different connectors on each end. The end with the ‘pin’ will go into the Breadboard. The end with the piece of plastic with a hole in it will go onto the Raspberry Pi’s GPIO pins.

The Raspberry Pi's GPIO Pins

GPIO stands for **General Purpose Input Output**. It is a way the Raspberry Pi can control and monitor the outside world by being connected to electronic circuits.



You should turn your Pi off for the next bit, just in case you accidentally short something out.

- Use one of the jumper wires to connect a ground pin to the rail, marked with blue, on the breadboard. The female end goes on the Pi's pin, and the male end goes into a hole on the breadboard.
- Then connect the resistor from the same row on the breadboard to a column on the breadboard, as shown above.
- Next, push the LEDs legs into the breadboard, with the long leg (with the kink) on the right.

Write a Python Code to glow an LED on (Using BOARD mode) :->

Step 1 : Source Code

Nano LED.py

```
import time
import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BOARD)
GPIO.setup(8, GPIO.OUT)
GPIO.output(8,False)
while True:
    GPIO.output(8,False)
```

Once you have typed all the code and checked it, save and exit the text editor with “Ctrl + x” then “y” then “enter”.

Step 2 : Running the Code

To run this code type:

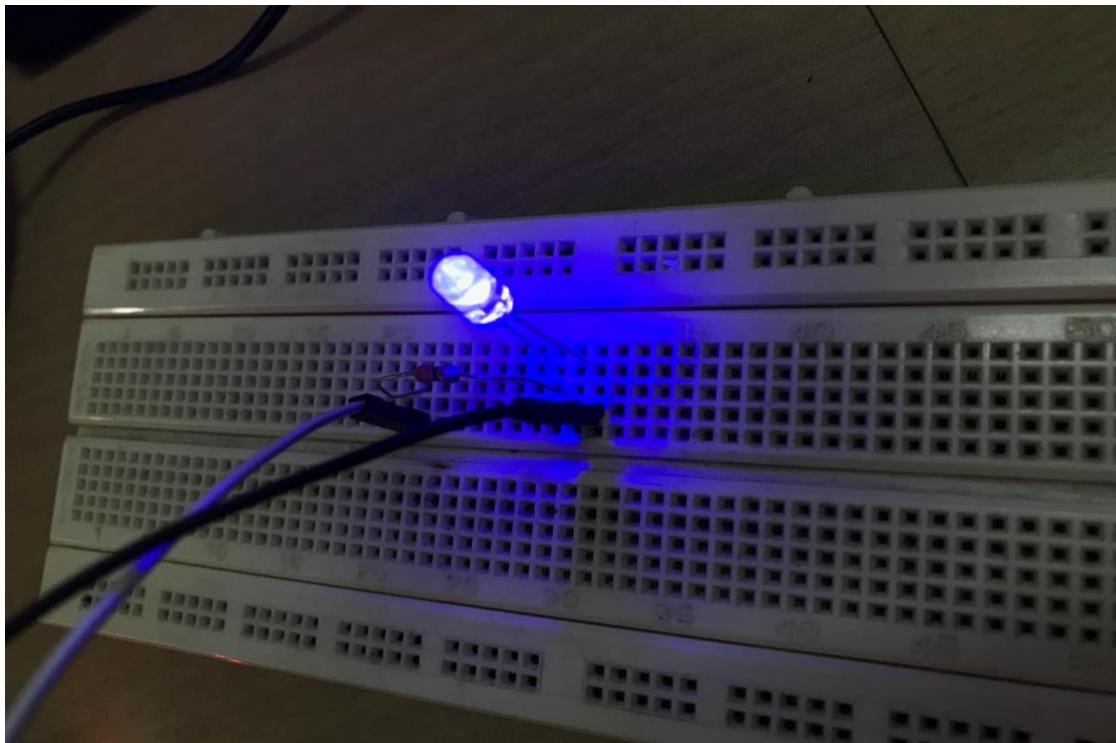
sudo python LED.py

You will see the LED turn on for a second and then turn off.

Explanation

import RPi.GPIO as GPIO	The first line tells the Python interpreter (the thing that runs the Python code) that it will be using a ‘library’ that will tell it how to work with the Raspberry Pi’s GPIO pins. A ‘library’ gives a programming language extra commands that can be used to do something different that it previously did not know how to do. This is like adding a new channel to your TV so you can watch something different.
import time	Imports the Time library so that we can pause the script later on.
GPIO.setmode(GPIO.BOARD)	Each pin on the Pi has several different names, so you need to tell the program which naming convention is to be used.
GPIO.setwarnings(False)	This tells Python not to print GPIO warning messages to the screen.
GPIO.setup(8, GPIO.OUT)	This line tells the Python interpreter that pin 18 is going to be used for outputting information, which means you are going to be able to turn the pin ‘on’ and ‘off’.
GPIO.output(8,True)	This turns the GPIO pin ‘on’. What this actually means is that the pin is made to provide power of 3.3volts. This is enough to turn the LED in our circuit on.
time.sleep(1)	Pauses the Python program for 1 second
GPIO.output(8,False)	This turns the GPIO pin ‘off’, meaning that the pin is no longer supplying any power.

The output look like below image :



Write a Python Code to glow an LED on (Using BCM mode) :->

```
import time
import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BCM)
GPIO.setup(23, GPIO.OUT)
GPIO.output(23,False)
while True:
    GPIO.output(23,False)
```

Write a program to blink LED using BOARD mode . Take pin as a input from user :->

```
import time
import RPi.GPIO as GPIO
a=int(input("Enter pin number:"))
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BOARD)
GPIO.setup(a, GPIO.OUT)
GPIO.output(a,True)
```

while True:

```
    GPIO.output(a, False)
    time.sleep(1)
    GPIO.output(a, True)
    time.sleep(1)
```

Write a program to blink LED. Take time as a input from user (using BOARD mode):->

```
import time
import RPi.GPIO as GPIO
a=float(input("Enter time:"))
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BOARD)
GPIO.setup(11, GPIO.OUT)
GPIO.output(11,True)
```

while True:

```
    GPIO.output(11, False)
    time.sleep(a)
    GPIO.output(11, True)
    time.sleep(a)
```

Write a program to blink the LED number of times requested by user (Using BCM Mode) :->

```
import time
import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BCM)
GPIO.setup(23, GPIO.OUT)
GPIO.output(23,True)
y = int (input ("Enter Number of times to blink an LED :"))
for i in range(0,y):
    GPIO.output(23,False)
    time.sleep(1)
    GPIO.output(23,True)
    time.sleep(1)
```

Write a program to Glow the LED .If condition is satisfied. Take choice as a input from user. (Using if-elif loop) :->

```

import time
import RPi.GPIO as GPIO
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BOARD)
GPIO.setup(8, GPIO.OUT)
GPIO.output(8,True)
y = int (input ("Enter your choice 1.To Glow an LED 2. To switch off an LED
3. To blink an LED:"))
if (y==1):
    a=int(input("Enter a pin number:"))
    while True:
        GPIO.output(a,True)
elif (y==2):
    b=int(input("Enter a pin number:"))
    while False:
        GPIO.output(b,False)
elif (y==3):
    while True:
        c=float(input("Enter time:"))
        GPIO.output(8,True)
        time.sleep(c)
        GPIO.output(8,False)
        time.sleep(c)

```

Write a program to blink 3 LED's 8 times :->

```

import time
import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BOARD)
GPIO.setup(7, GPIO.OUT)
GPIO.output(7,True)
GPIO.setup(8, GPIO.OUT)
GPIO.output(8,True)
GPIO.setup(10, GPIO.OUT)
GPIO.output(10,True)
for i in range(1,9);
    GPIO.output(7,False)
    time.sleep(1)
    GPIO.output(7,True)
    time.sleep(1)
    GPIO.output(8,False)
    time.sleep(1)
    GPIO.output(8,True)

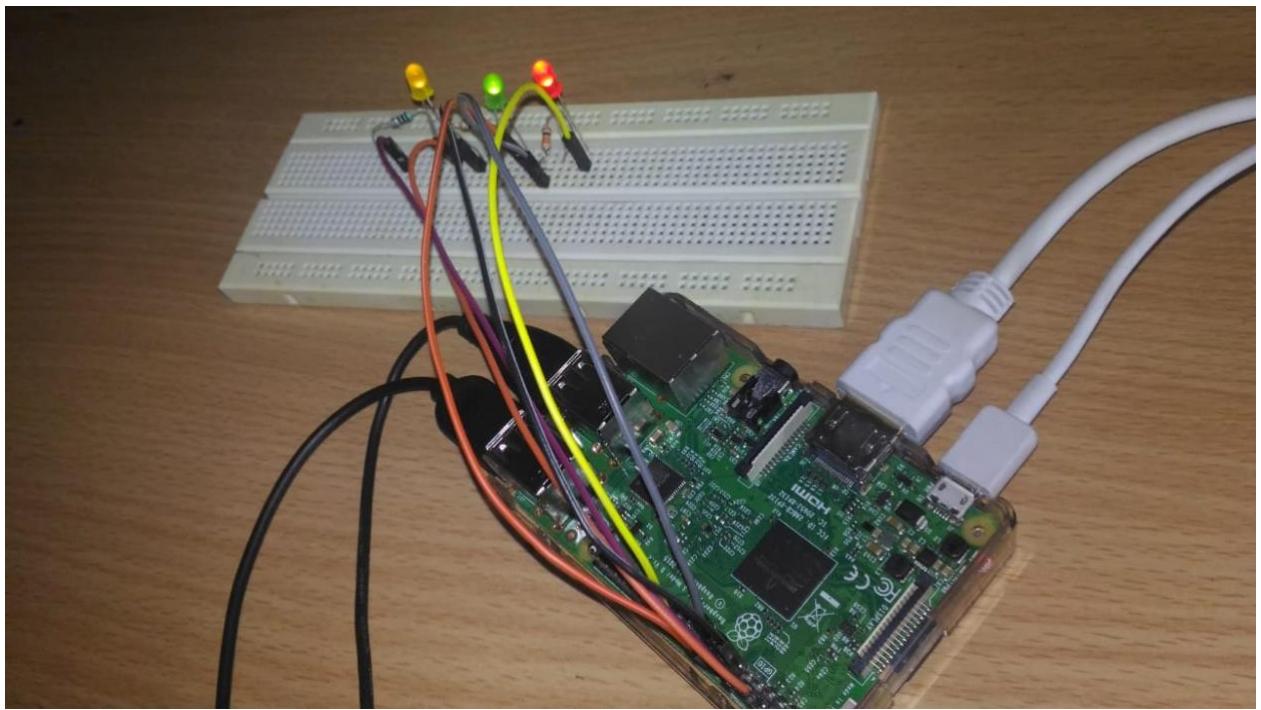
```

```
time.sleep(1)
GPIO.output(10,False)
time.sleep(1)
GPIO.output(10,True)
time.sleep(1)
```

Write a program to blink 3 LED's. Take time as a input from user :->

```
import time
import RPi.GPIO as GPIO
a=float(input("Enter time:"))
b=float(input("Enter time:"))
c=float(input("Enter time:"))
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BOARD)
GPIO.setup(8, GPIO.OUT)
GPIO.setup(11, GPIO.OUT)
GPIO.setup(15, GPIO.OUT)
while True:
    GPIO.output(8, False)
    time.sleep(a)
    GPIO.output(8, True)
    time.sleep(a)
    GPIO.output(11, False)
    time.sleep(b)
    GPIO.output(11, True)
    time.sleep(b)
    GPIO.output(15, False)
    time.sleep(c)
    GPIO.output(15, True)
    time.sleep(c)
```

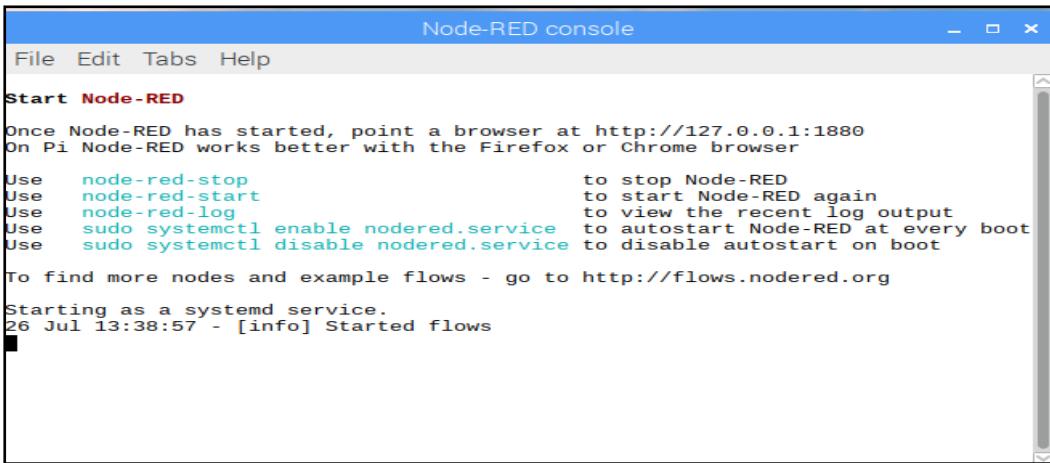
The output look like below image :



Practical 5:

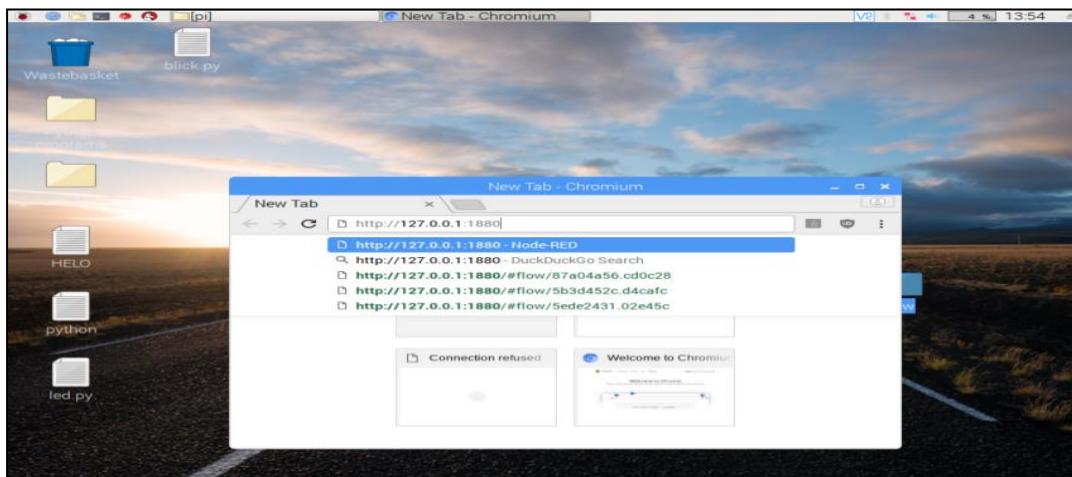
AIM: Run the node red editor and run simple programs and trigger GPIO.
Use basic nodes such as inject, debug, GPIO.

1. Copy the below given link

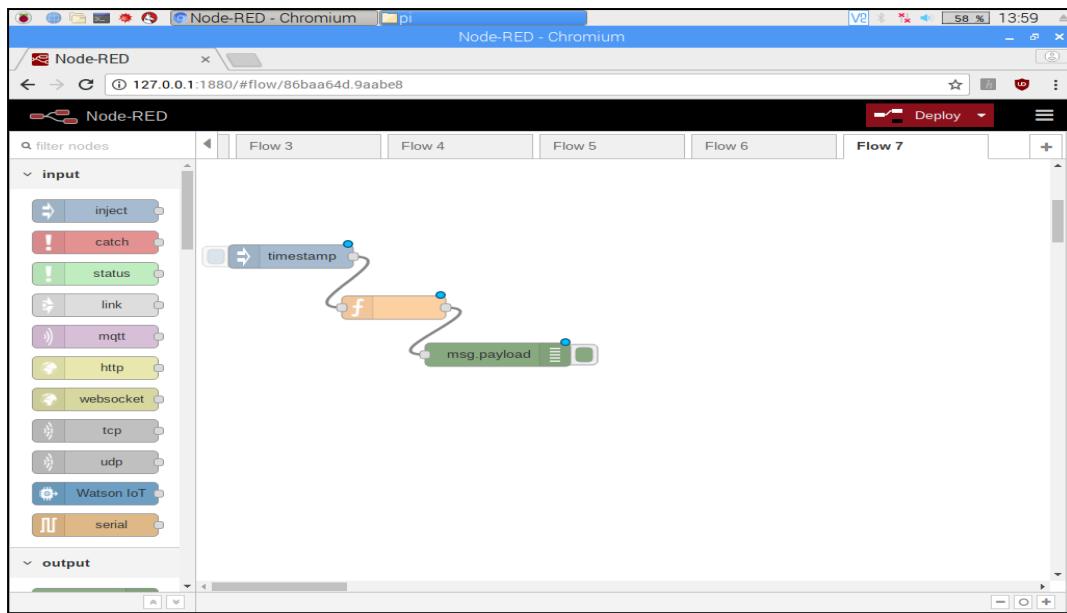


```
Node-RED console
File Edit Tabs Help
Start Node-RED
Once Node-RED has started, point a browser at http://127.0.0.1:1880
On Pi Node-RED works better with the Firefox or Chrome browser
Use node-red-stop to stop Node-RED
Use node-red-start to start Node-RED again
Use node-red-log to view the recent log output
Use sudo systemctl enable nodered.service to autostart Node-RED at every boot
Use sudo systemctl disable nodered.service to disable autostart on boot
To find more nodes and example flows - go to http://flows.nodered.org
Starting as a systemd service.
26 Jul 13:38:57 - [info] Started flows
```

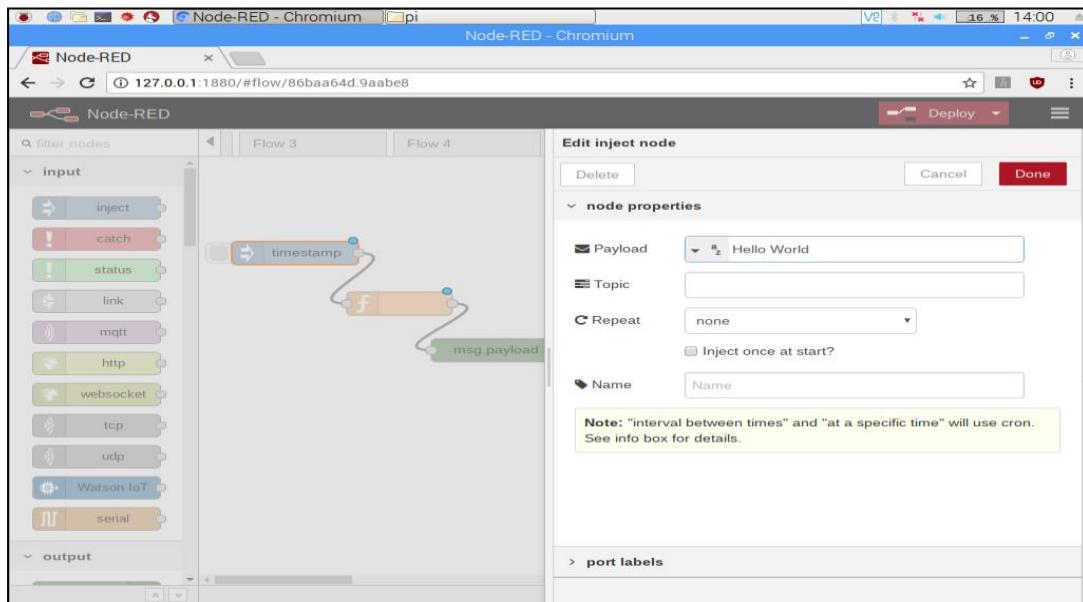
2. Paste it in the browser



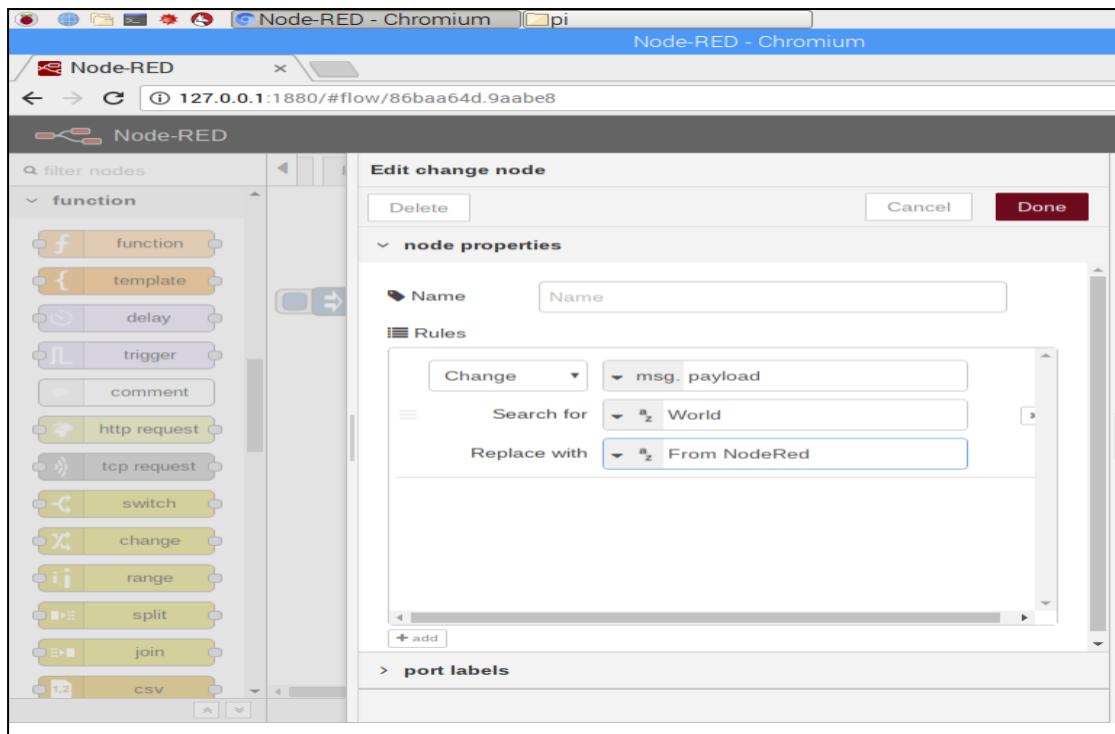
3. There are some readymade functions given in node red Eg:Change function. Create a flow as given below



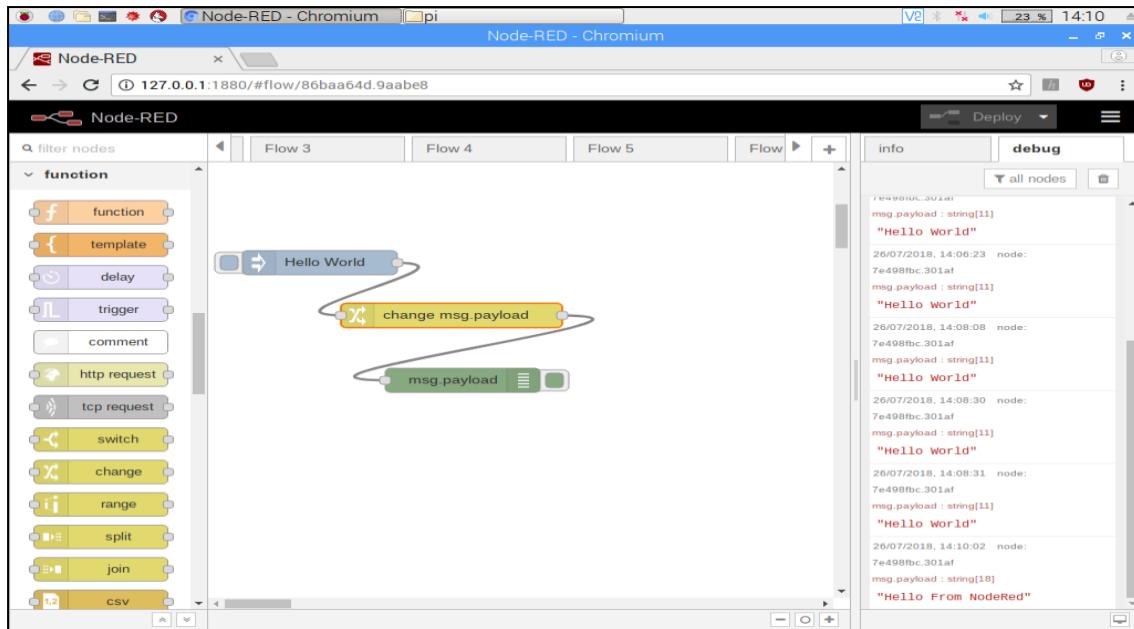
4. Double click on time stamp and write it as “Hello World” in payload



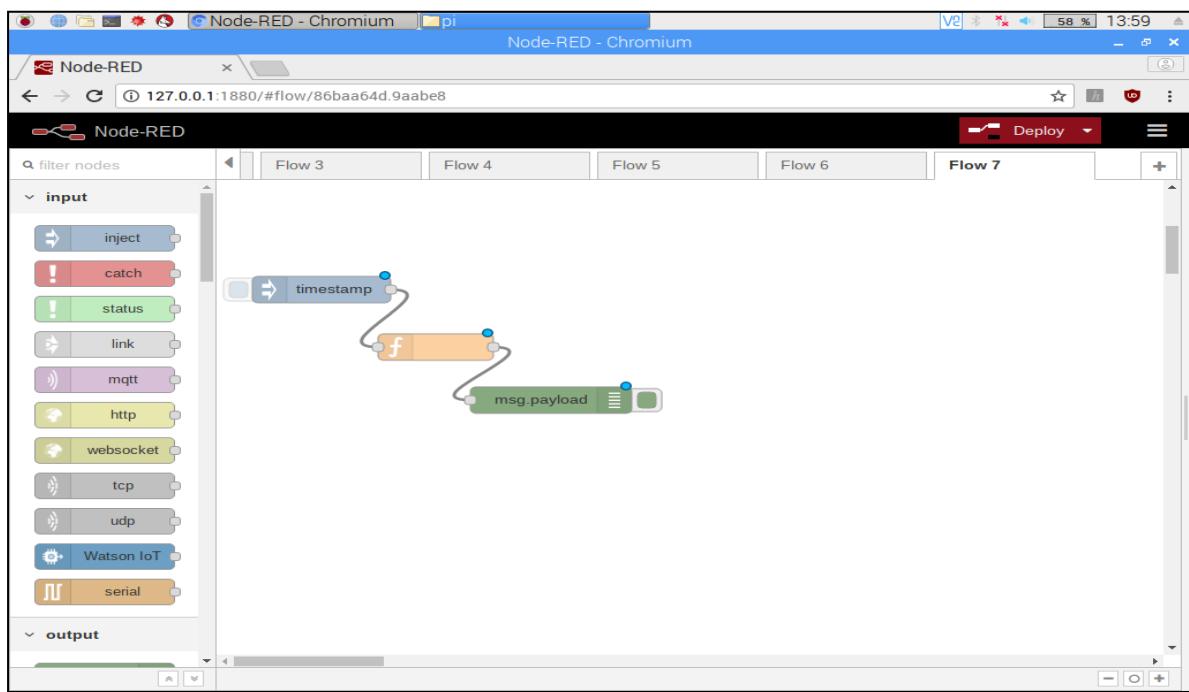
5. Double click on change function and search for “world” and replace with “From Nodered”.



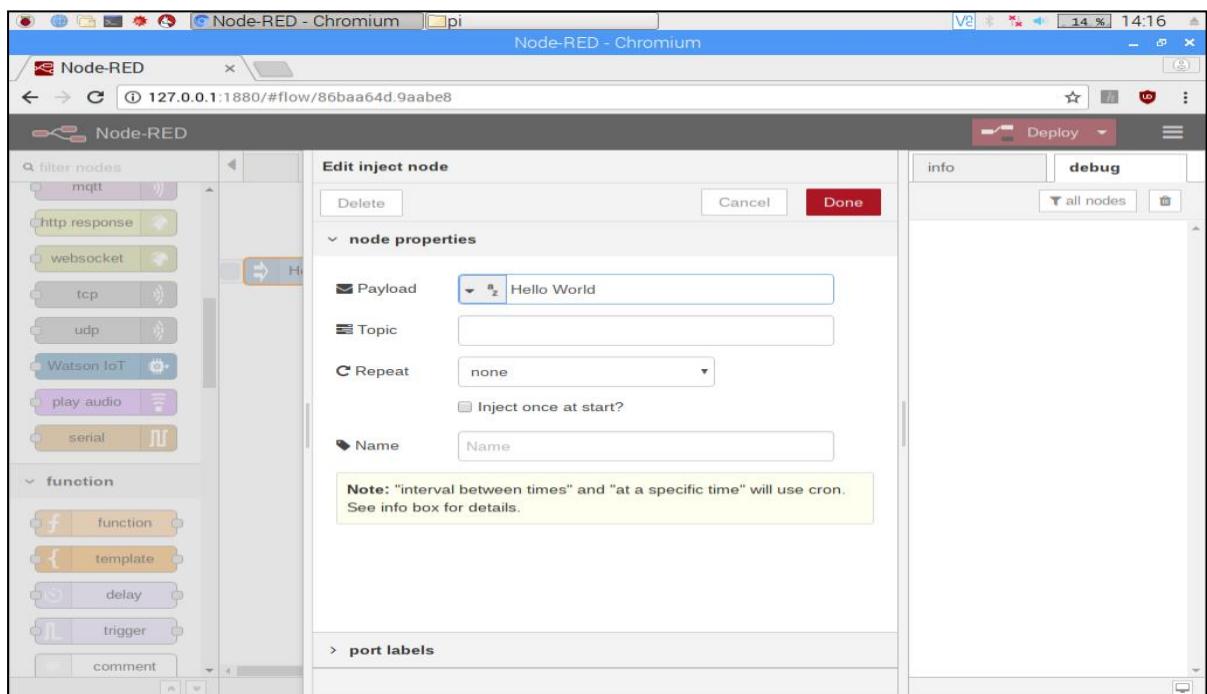
6. Then deploy and check the result.



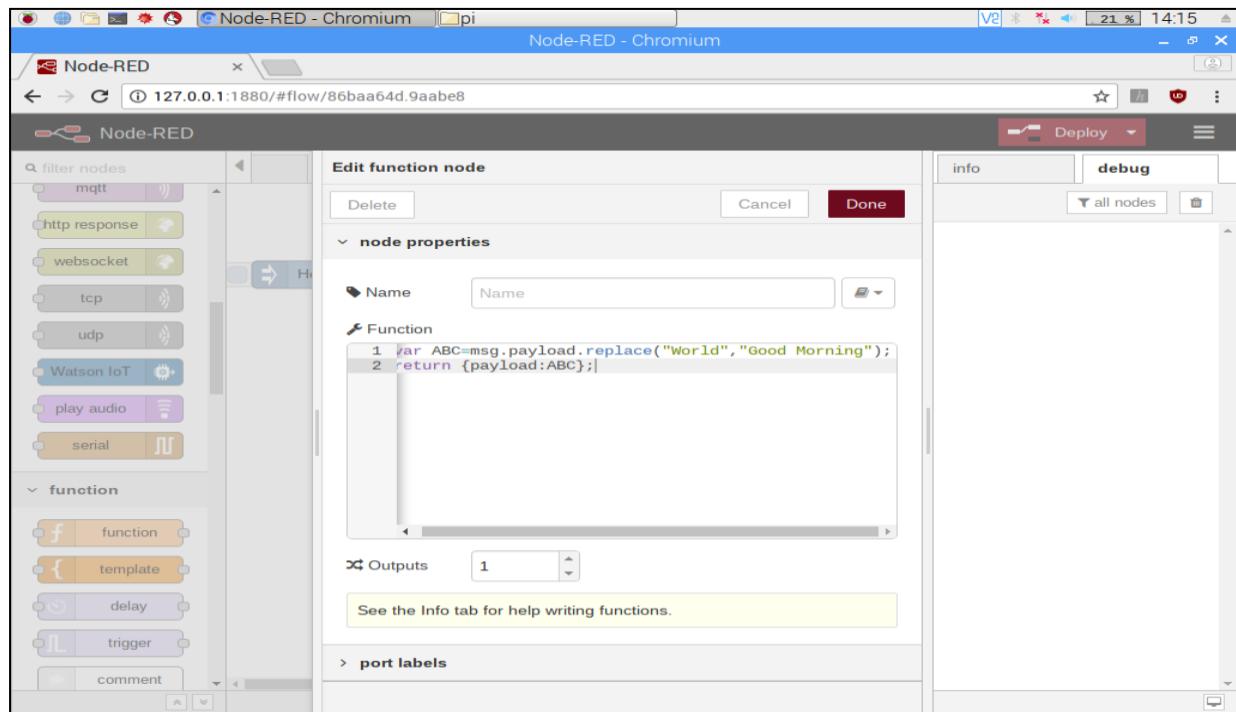
6. By using self declared functions(change function): Create a similar flow as above but instead of using readymade change function use a simple function



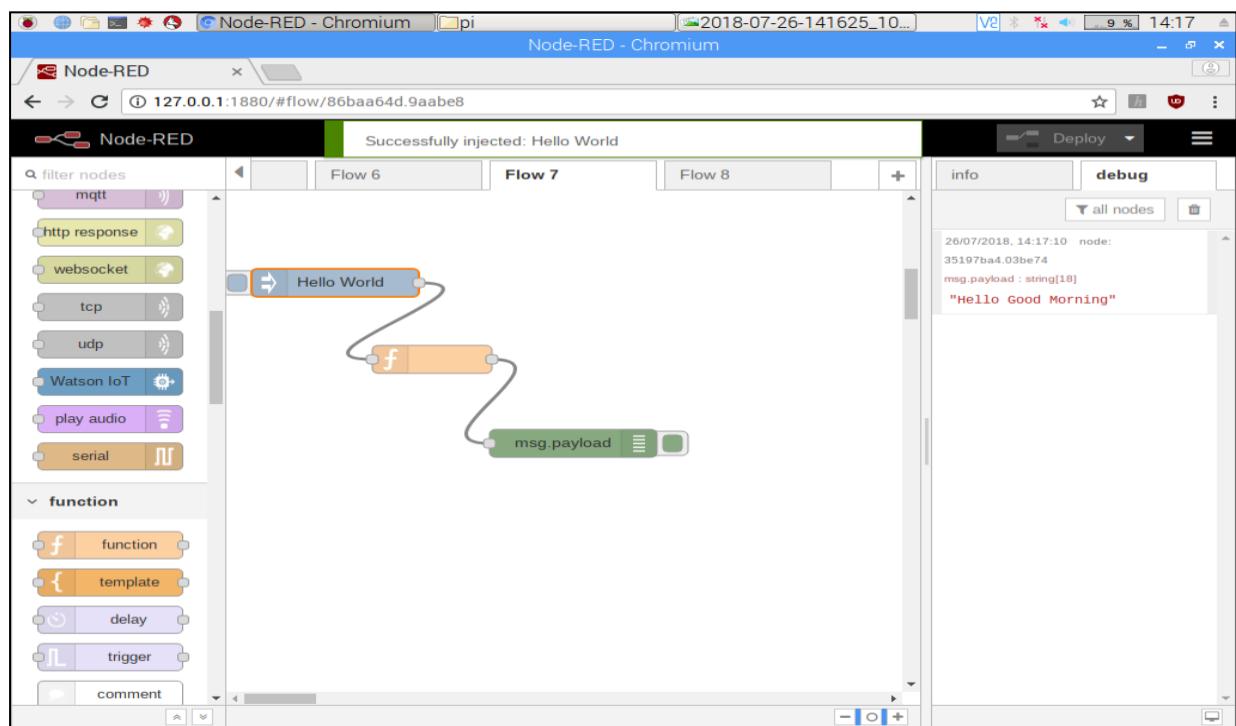
8. Double click on time stamp and write it as “Hello World” in payload



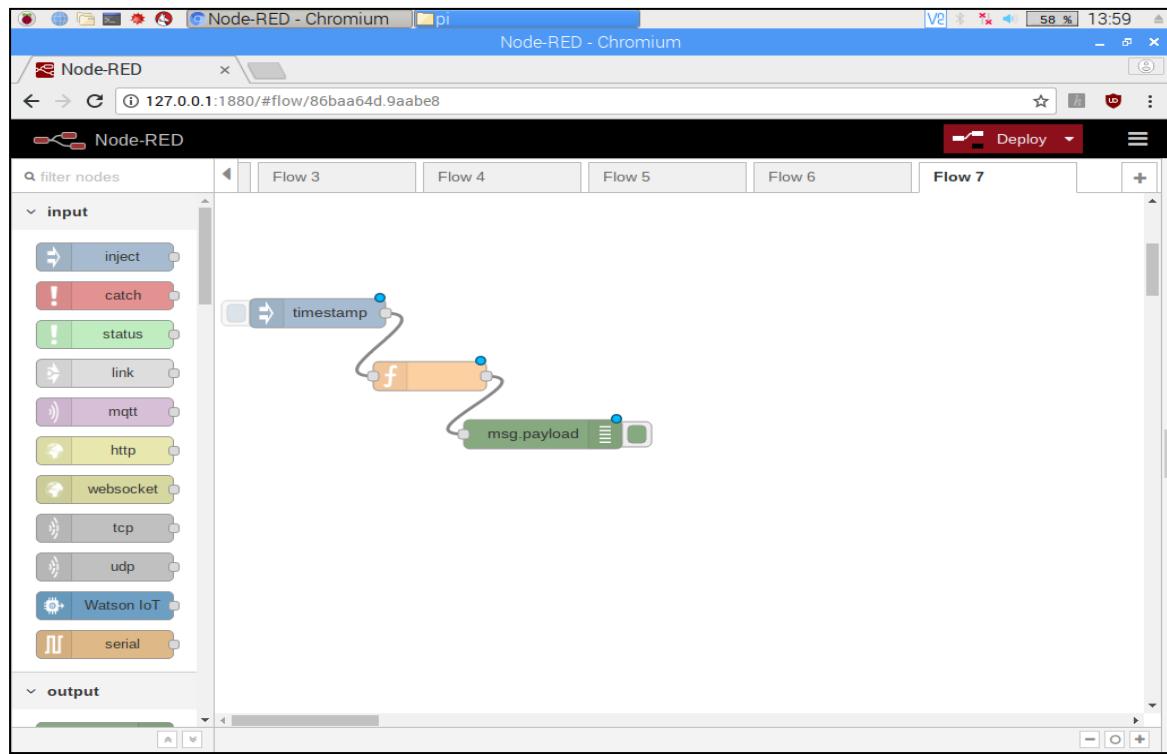
9. Double click on function and write the below code



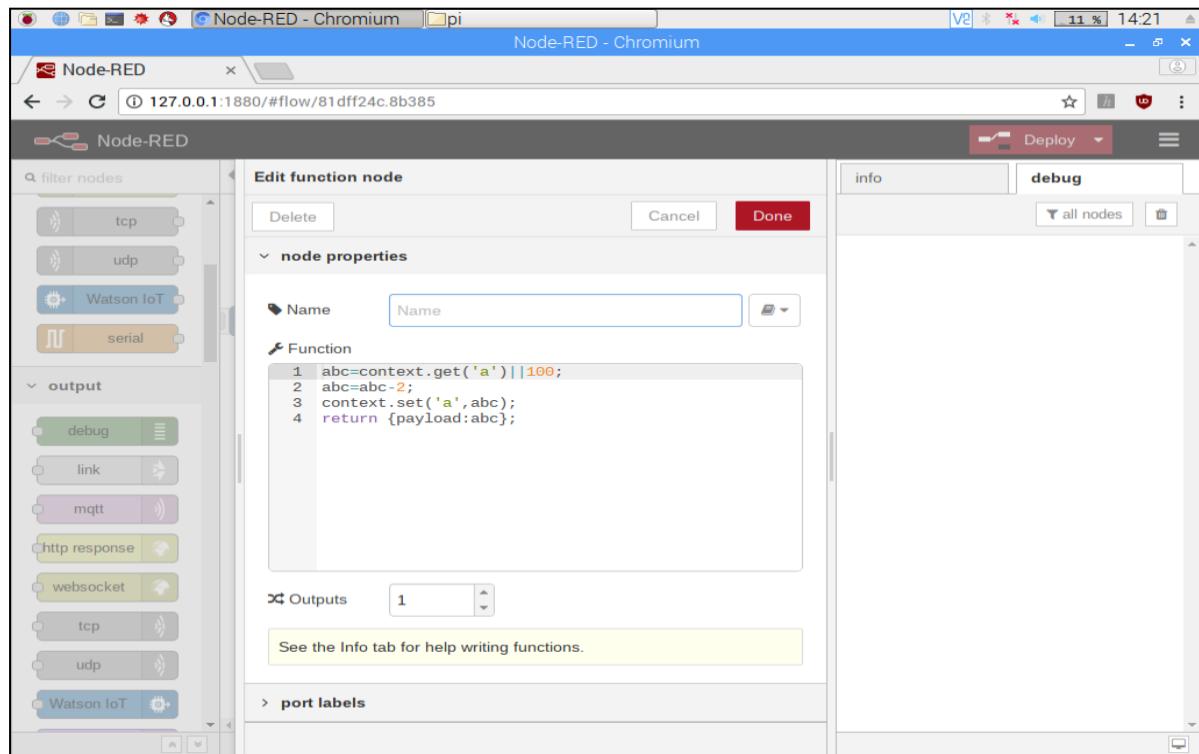
10. Deploy and check the output



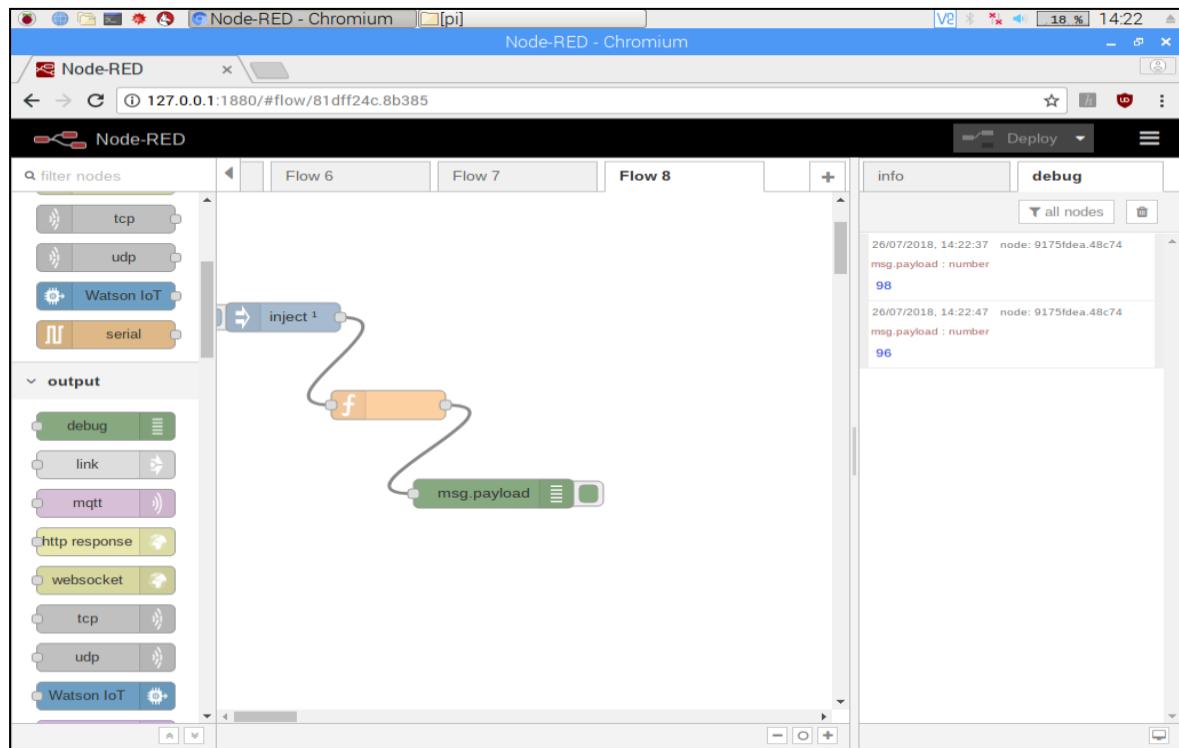
11. Function to reduce the number by 2. Create the flow as below



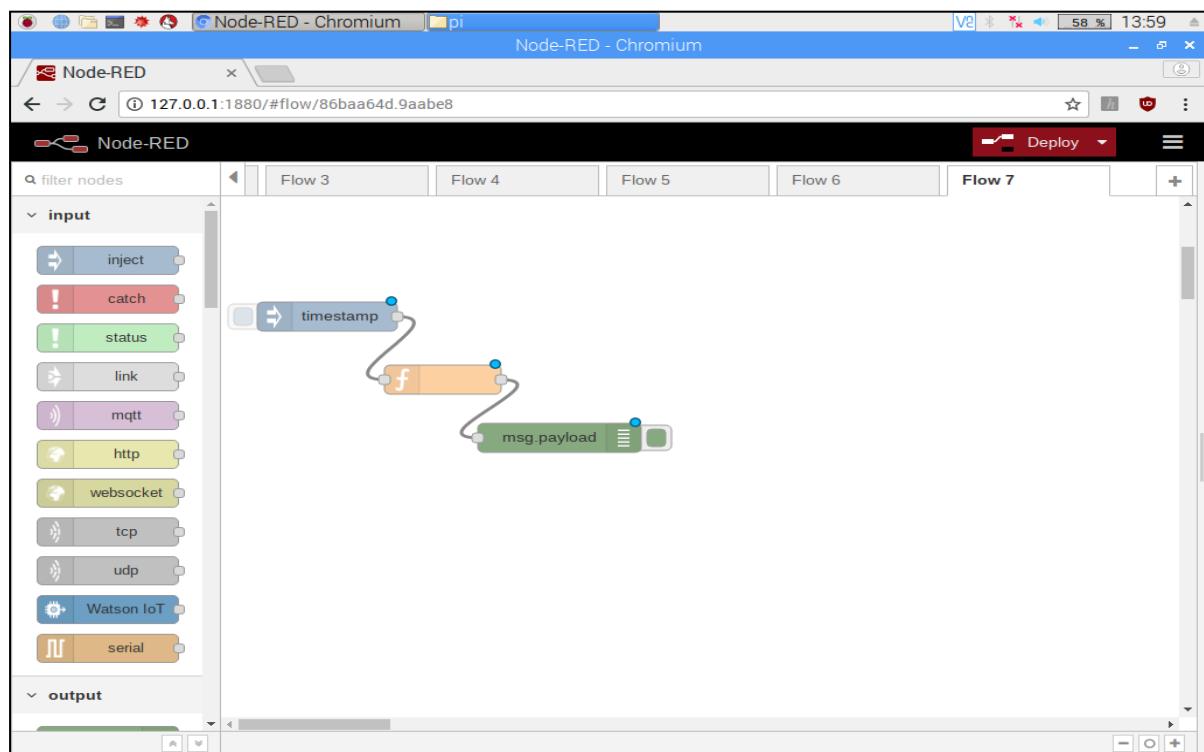
12. Double click the function and write the code given below:



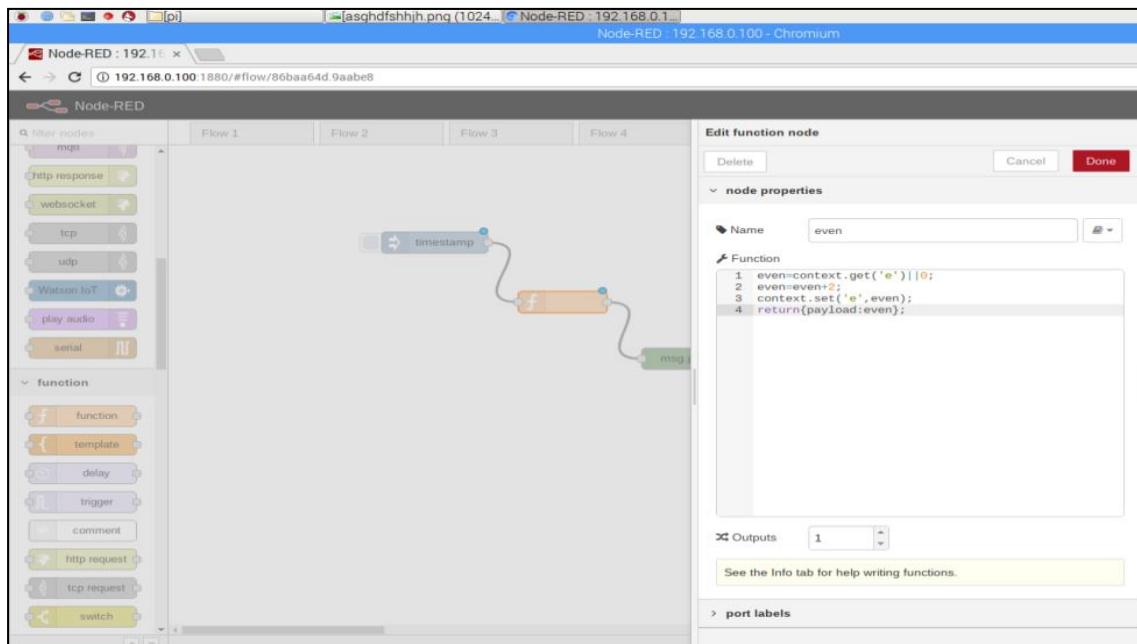
13. Then deploy and check the output



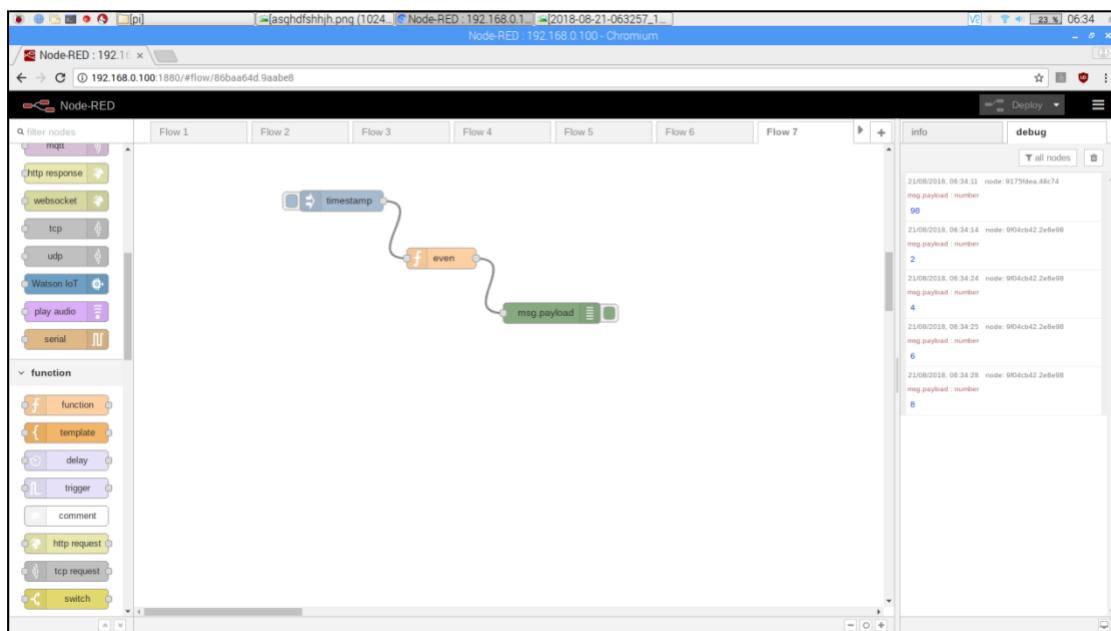
14. Function to display even numbers. Create the flow as below



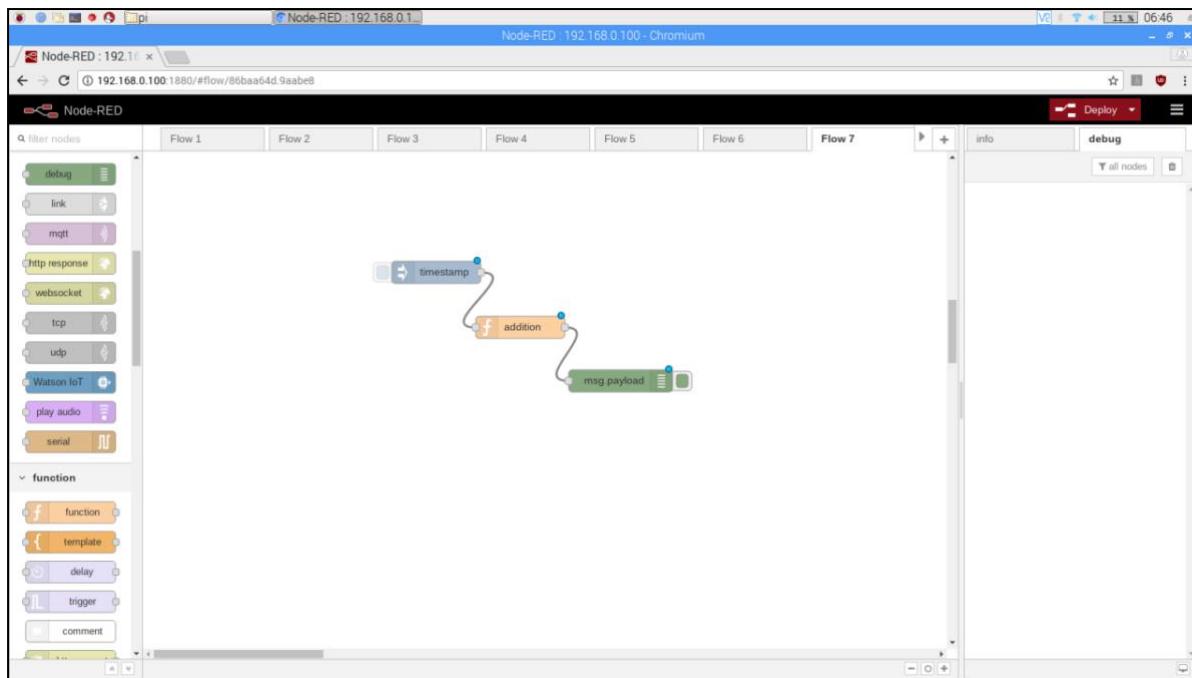
15. Double click the function and write the code given below:



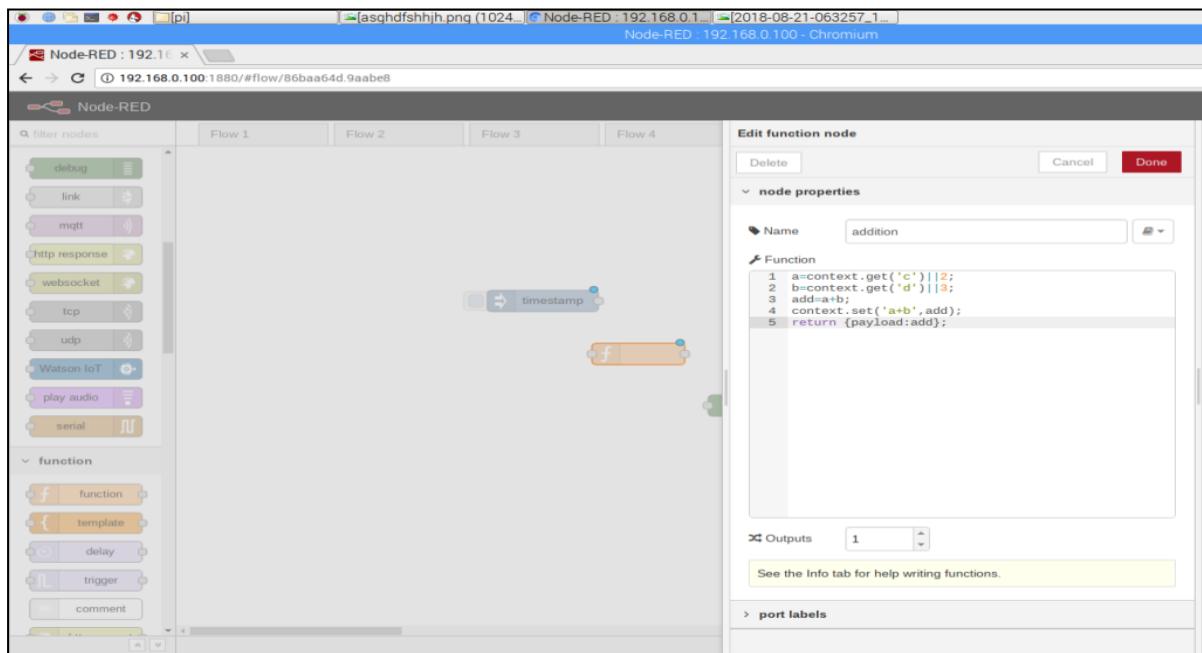
16. Deploy and check the output.



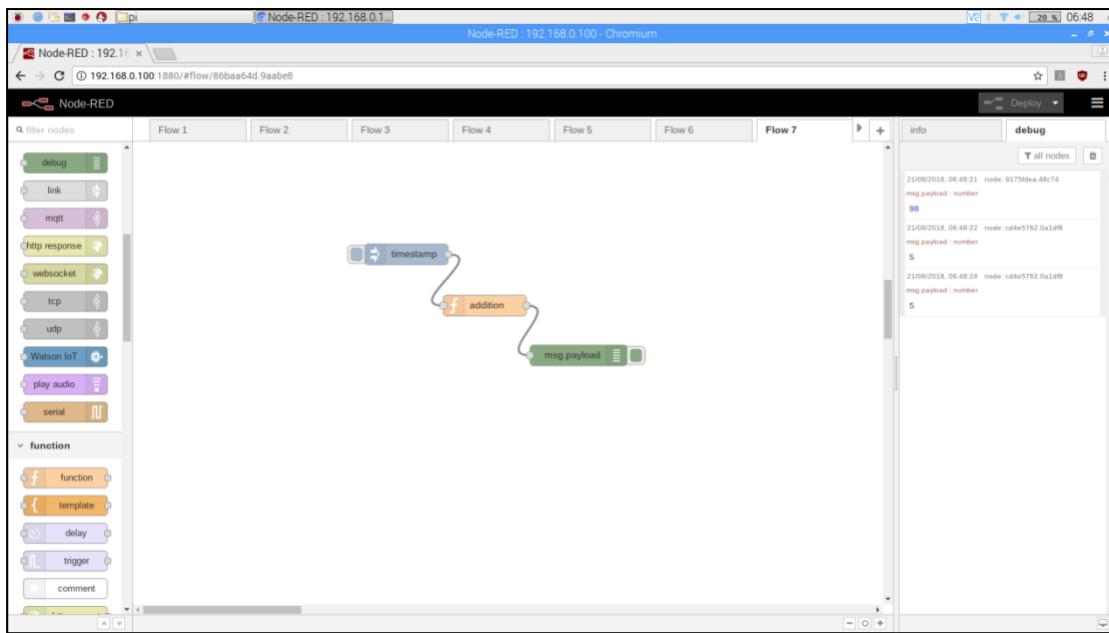
17. Function for adding two numbers. Create the flow as given below.



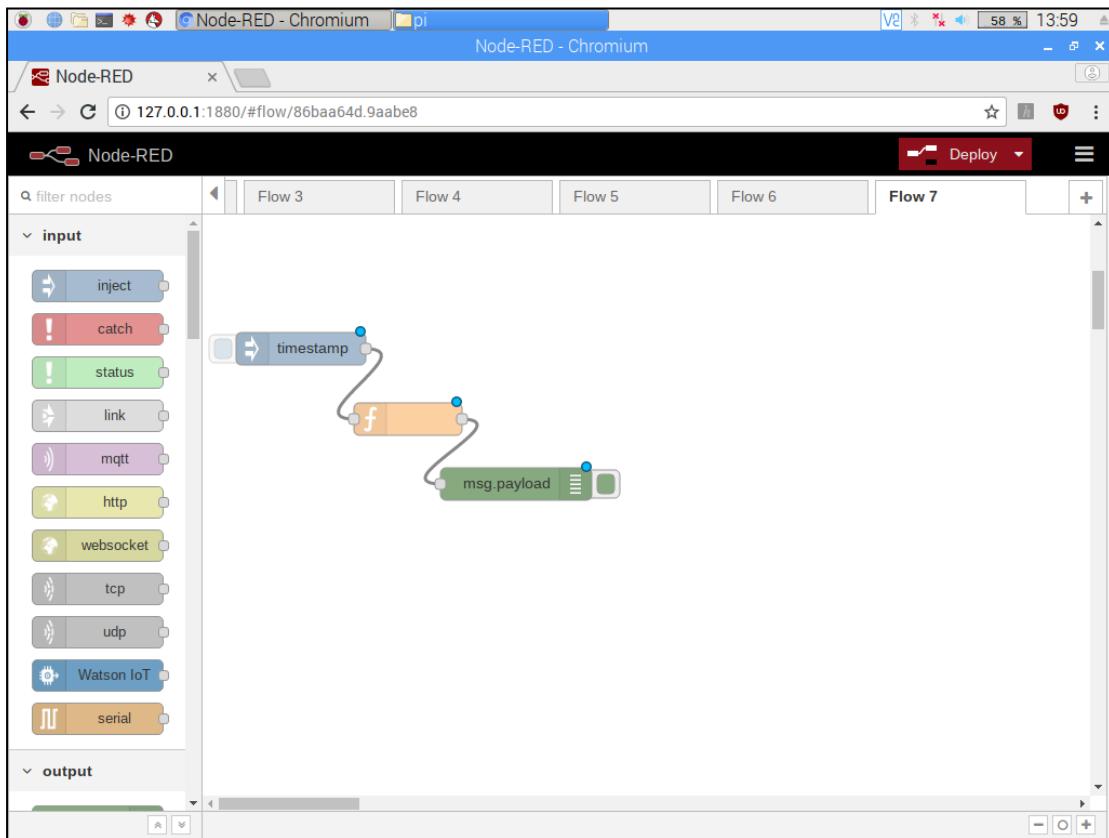
18. Double click the function and write the below given code



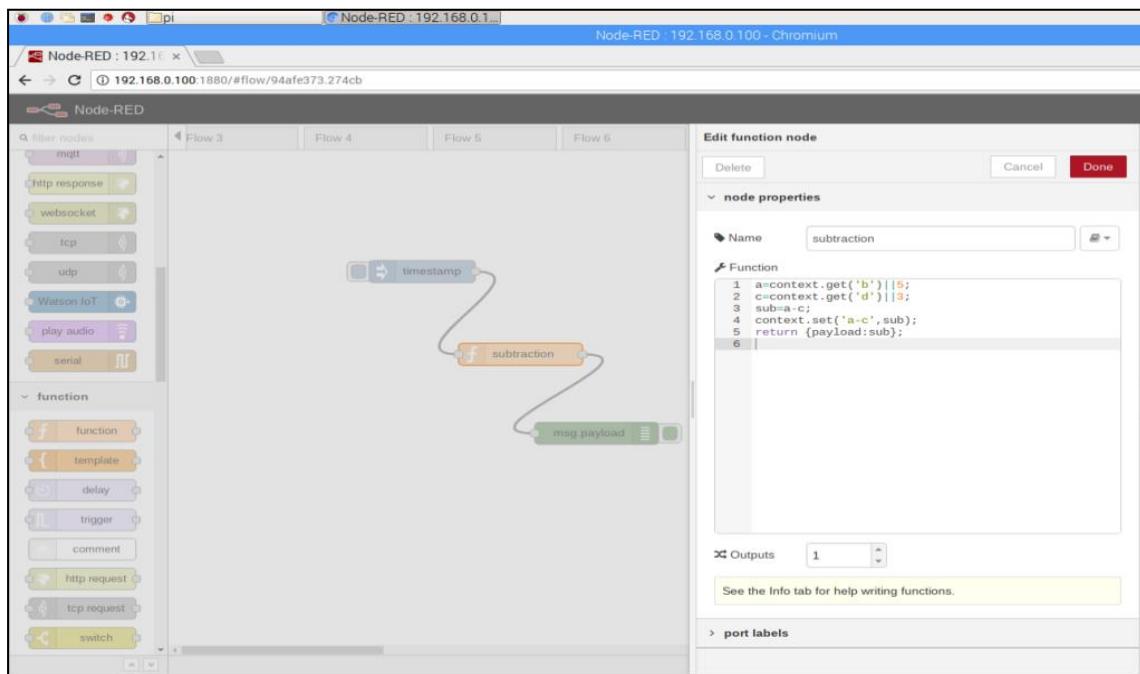
19. Deploy and check the output.



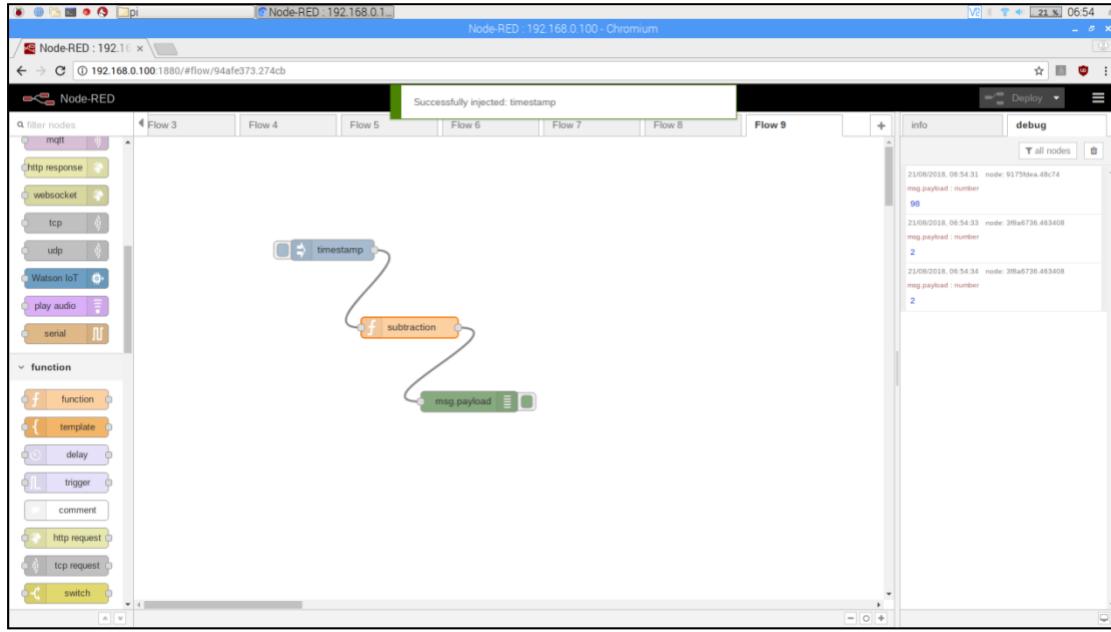
20. Function to subtract 2 numbers. Create the flow as given below.



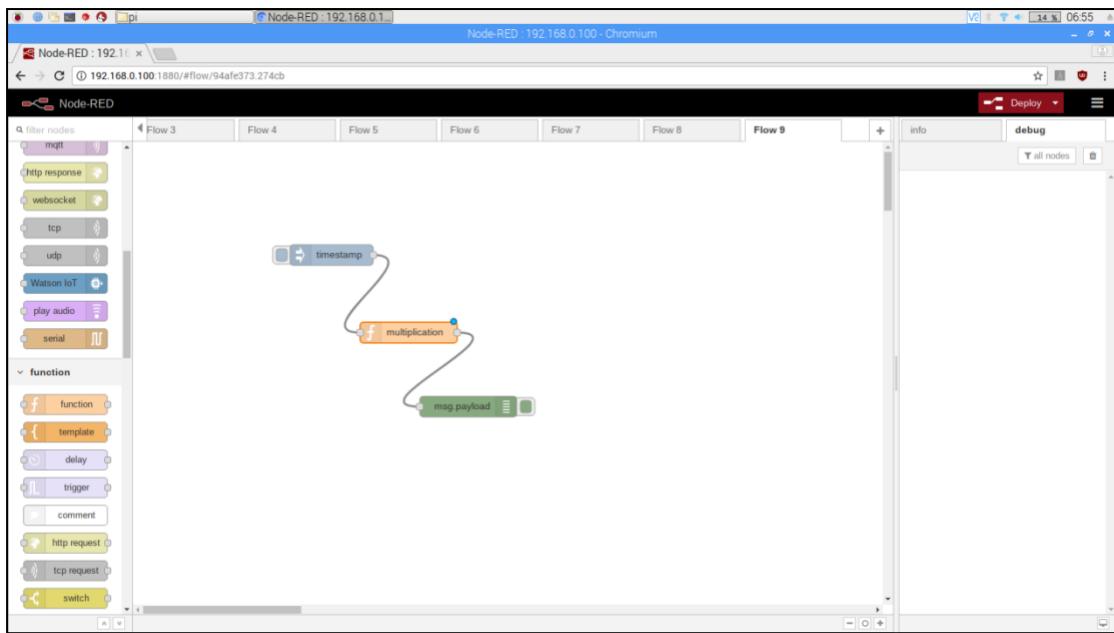
21. Double click the function and copy the code.



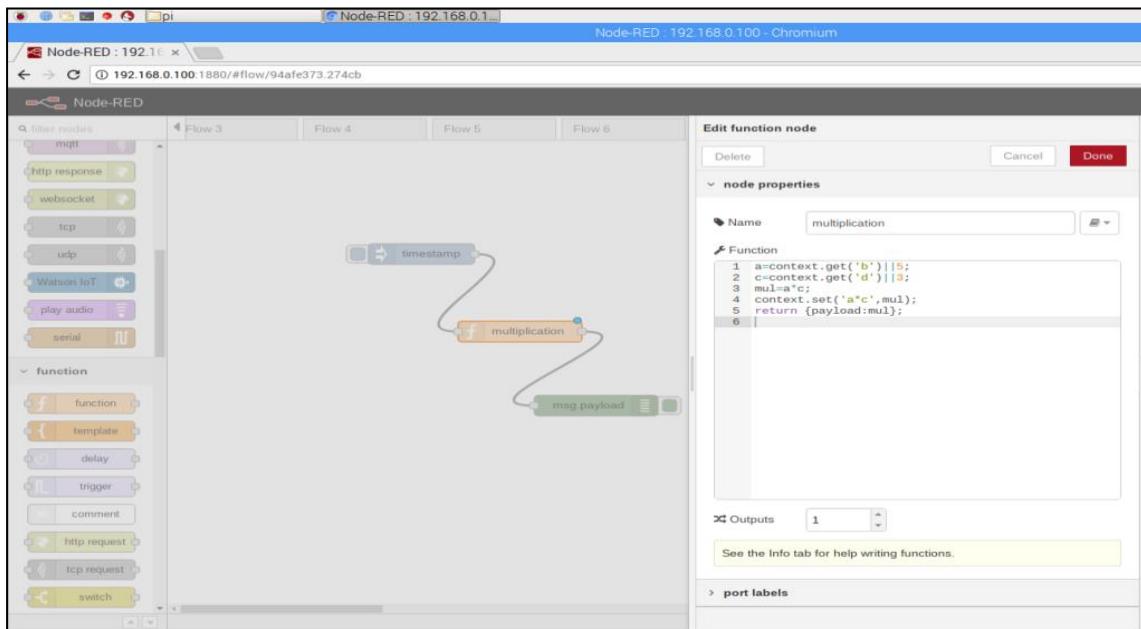
22. Deploy and check the output.



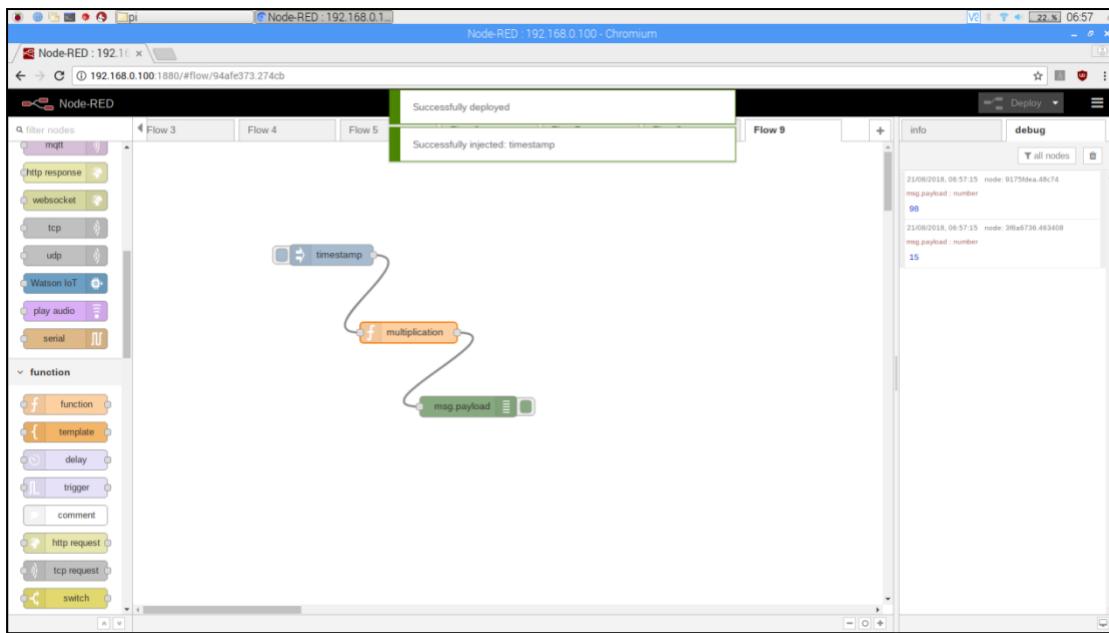
23. Function for multiplying 2 numbers. Create the flow as given below.



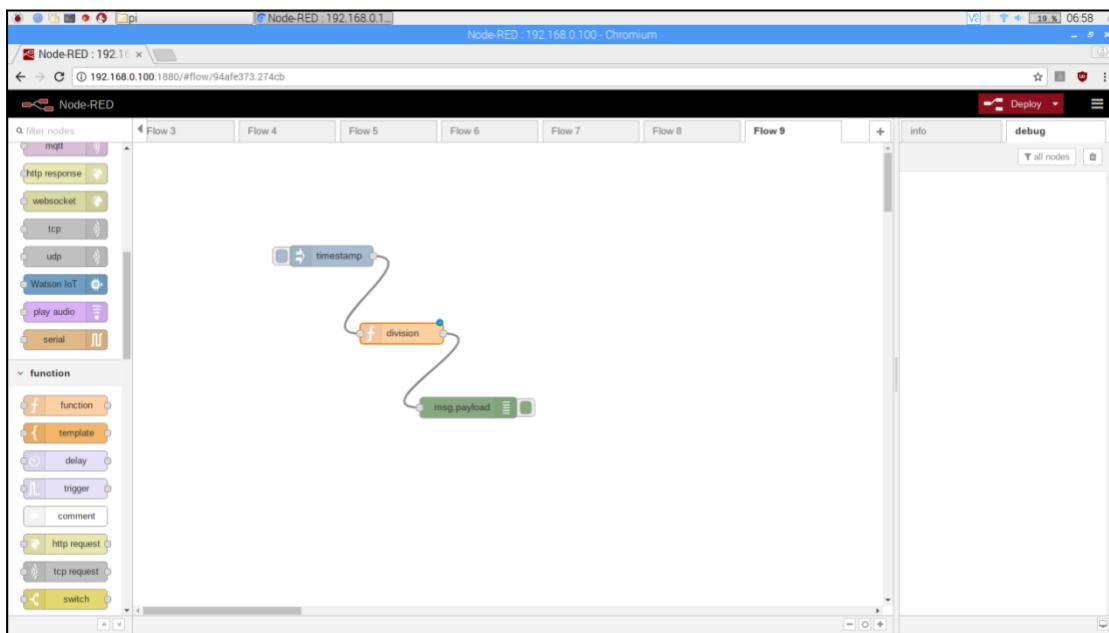
24. Double click the function and write the below code.



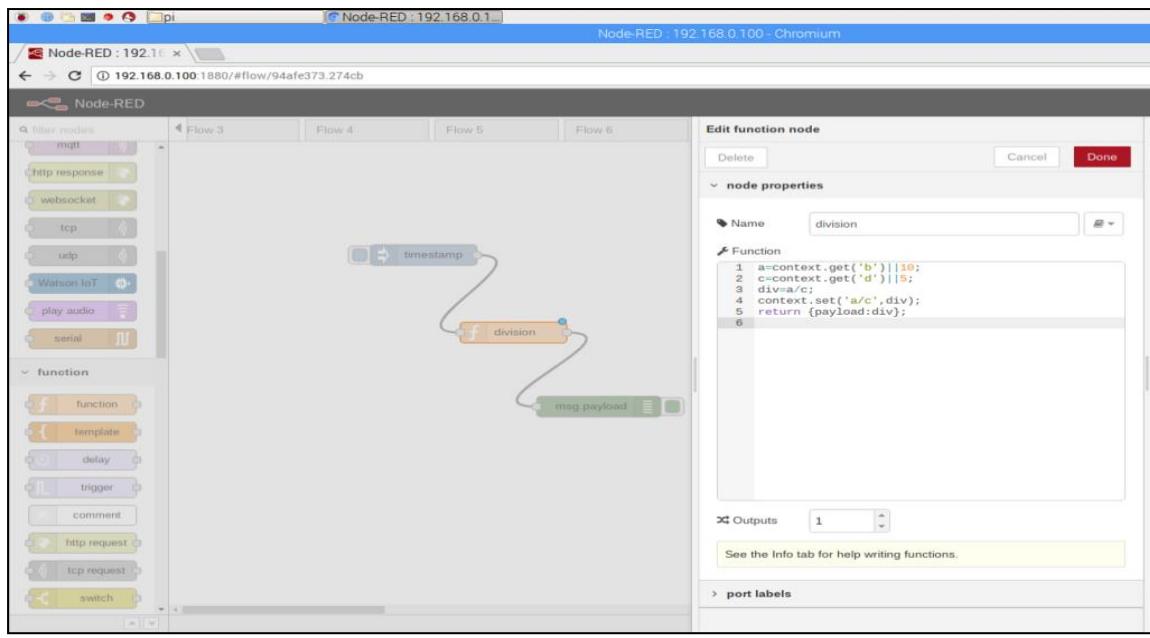
25. Deploy and check the result.



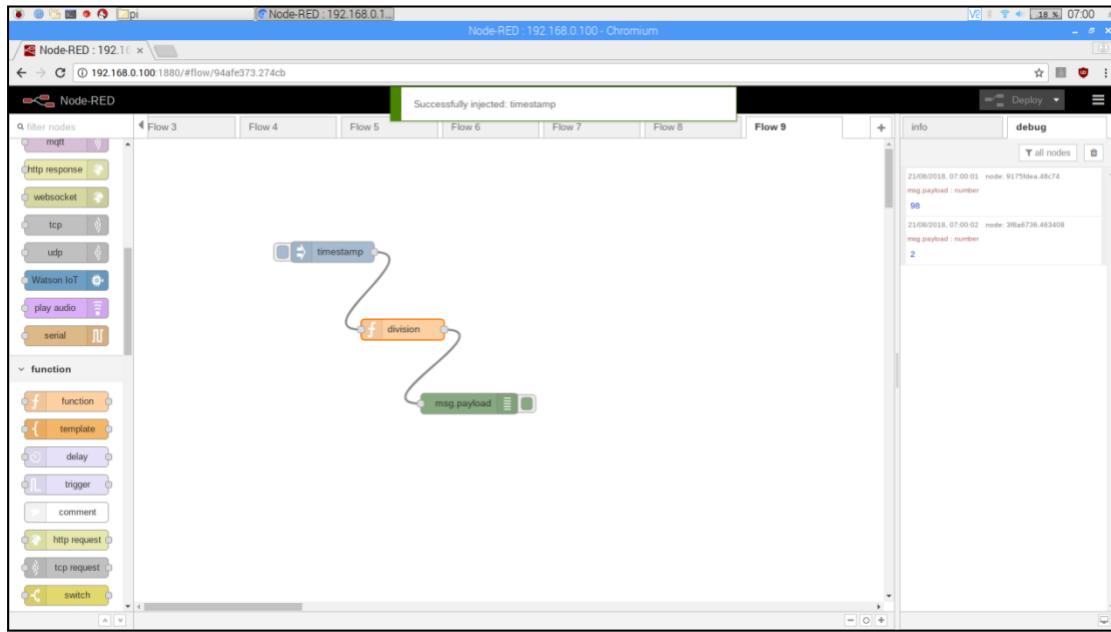
26. Function for dividing 2 numbers. Create the flow as given below.



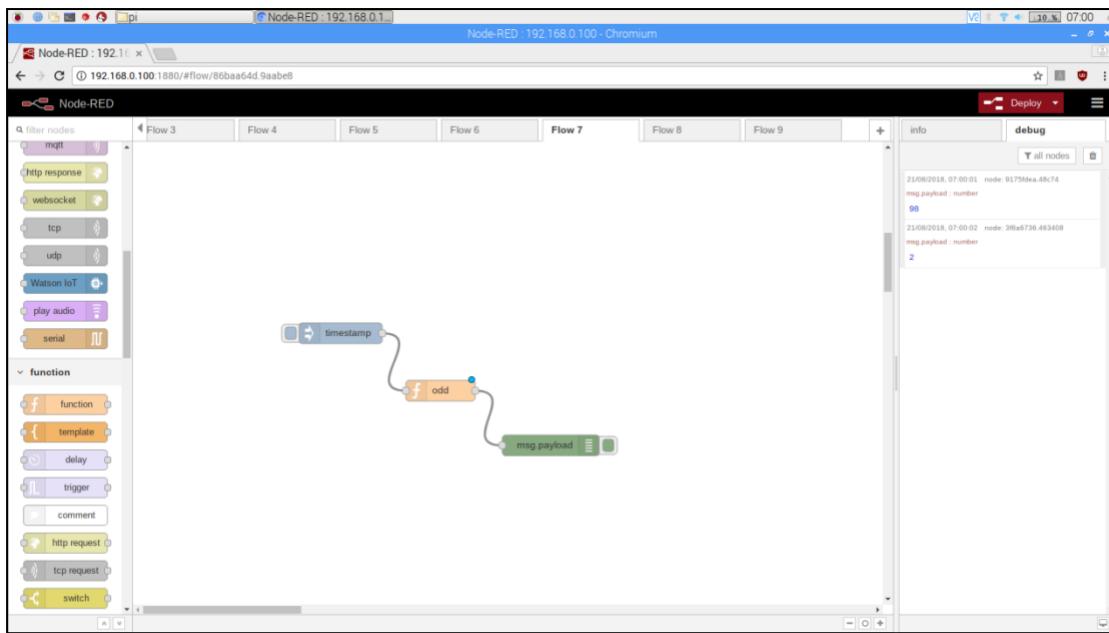
27. Double click the function and write the below code.



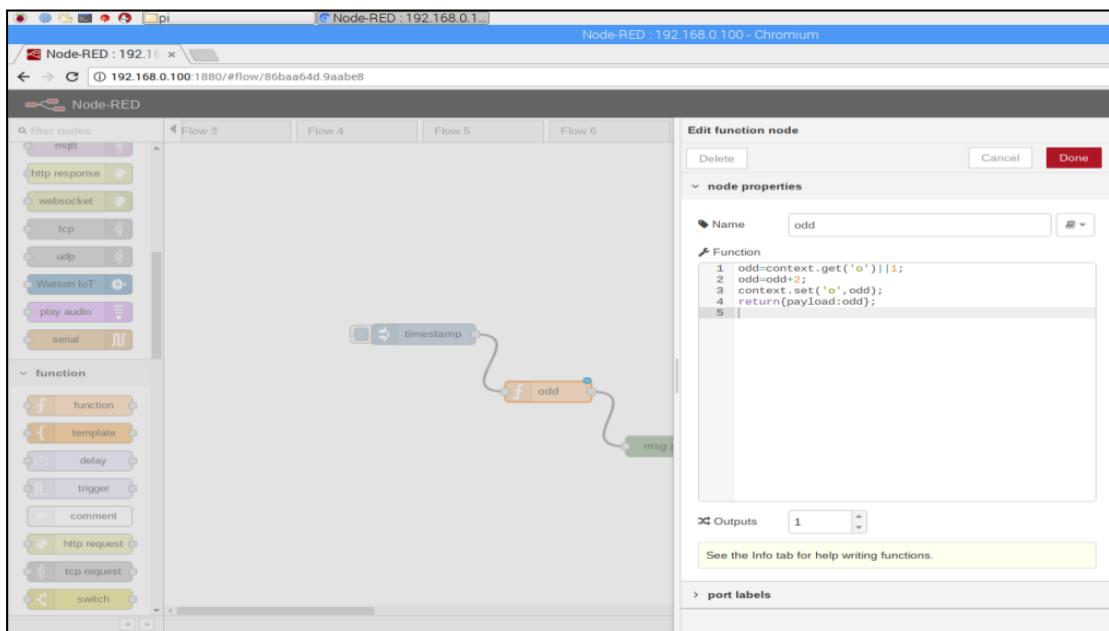
28. Deploy and check the output.



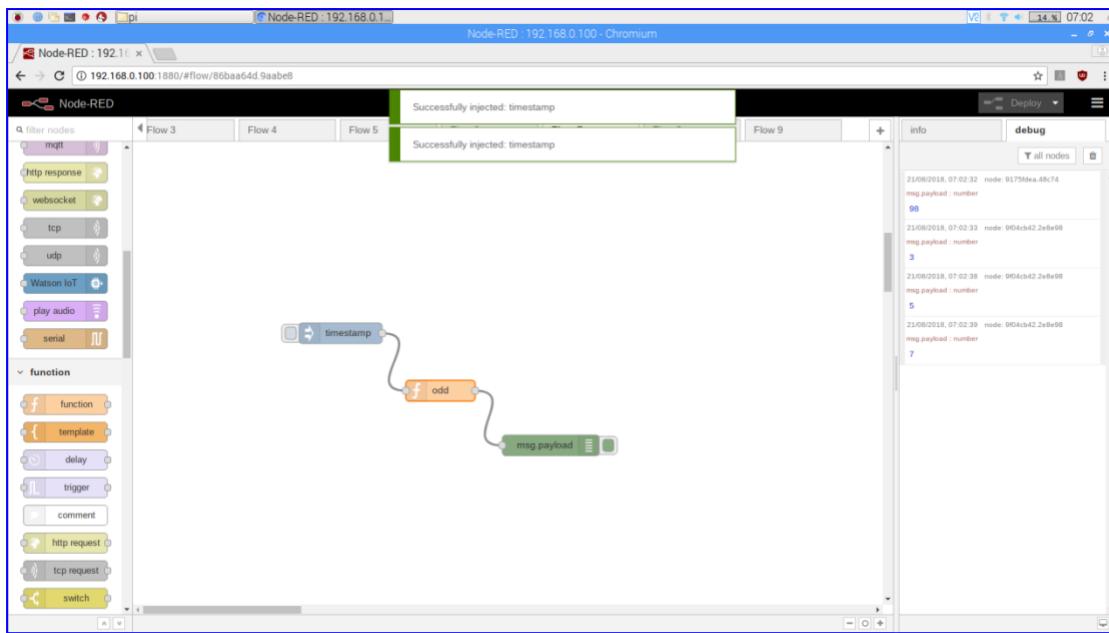
29. Function for generating odd numbers . Create the flow as given below.



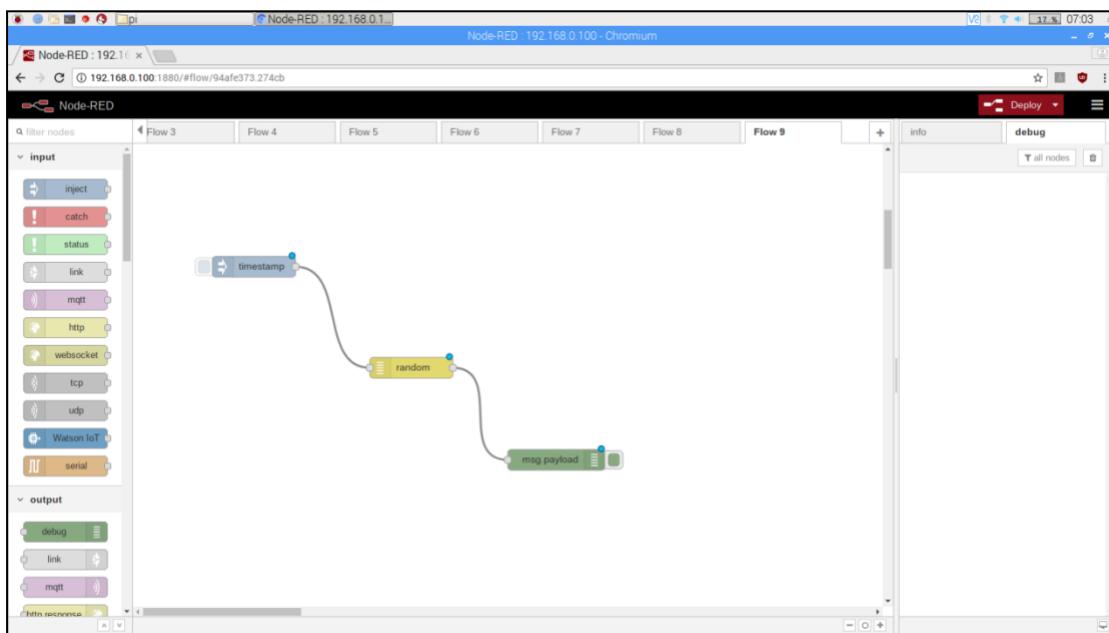
30. Double click the function and write the below code.



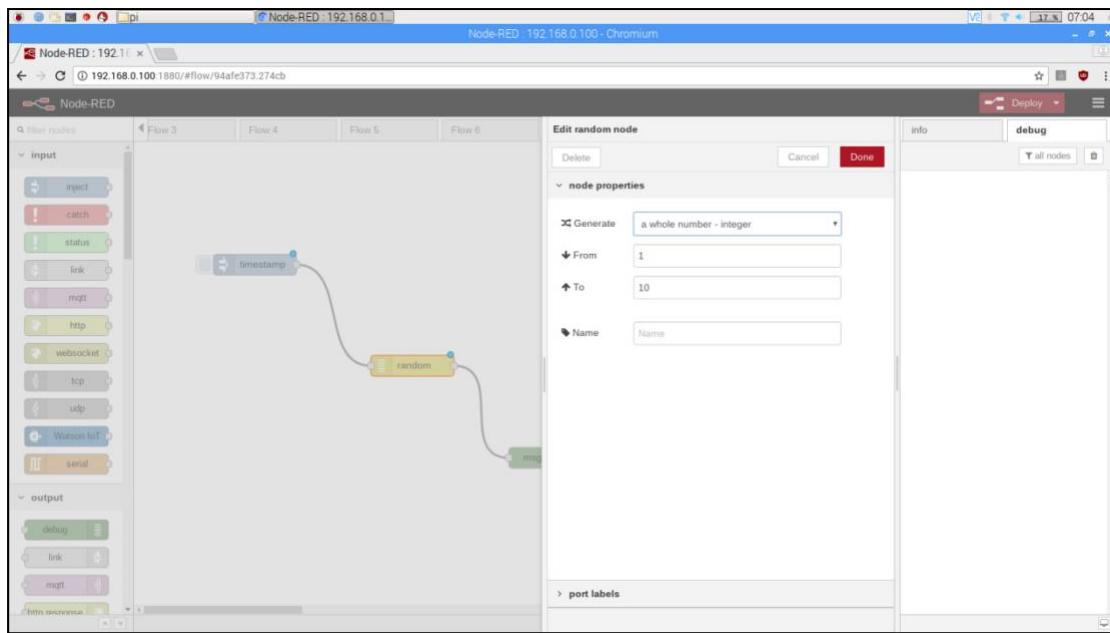
31. Deploy and check the result.



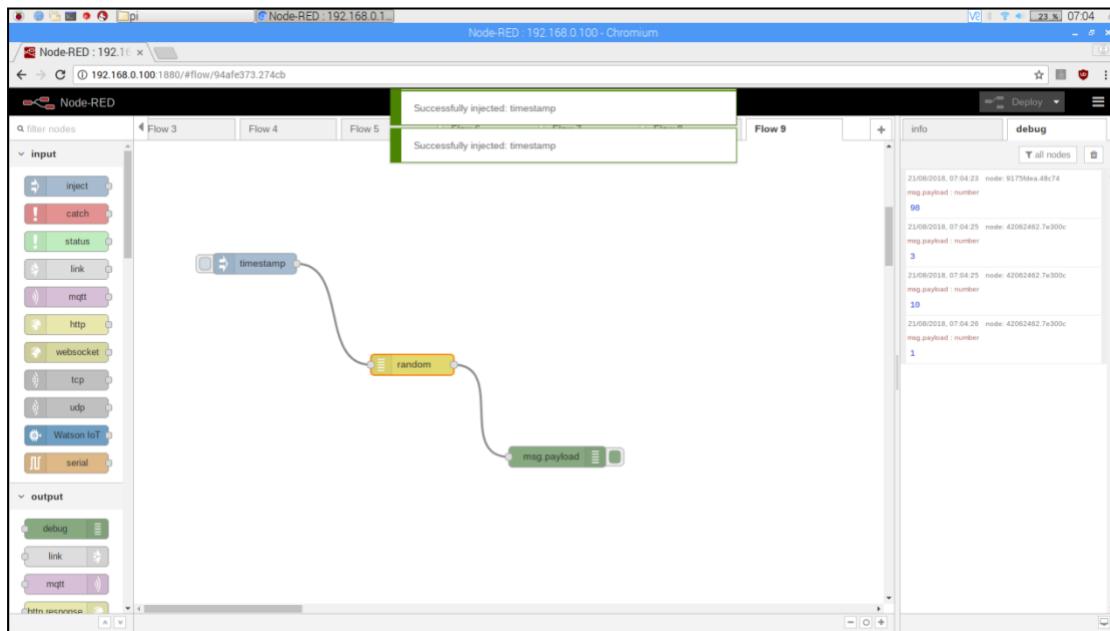
32. Function for random numbers. Create the flow given below by taking inbuilt random function.



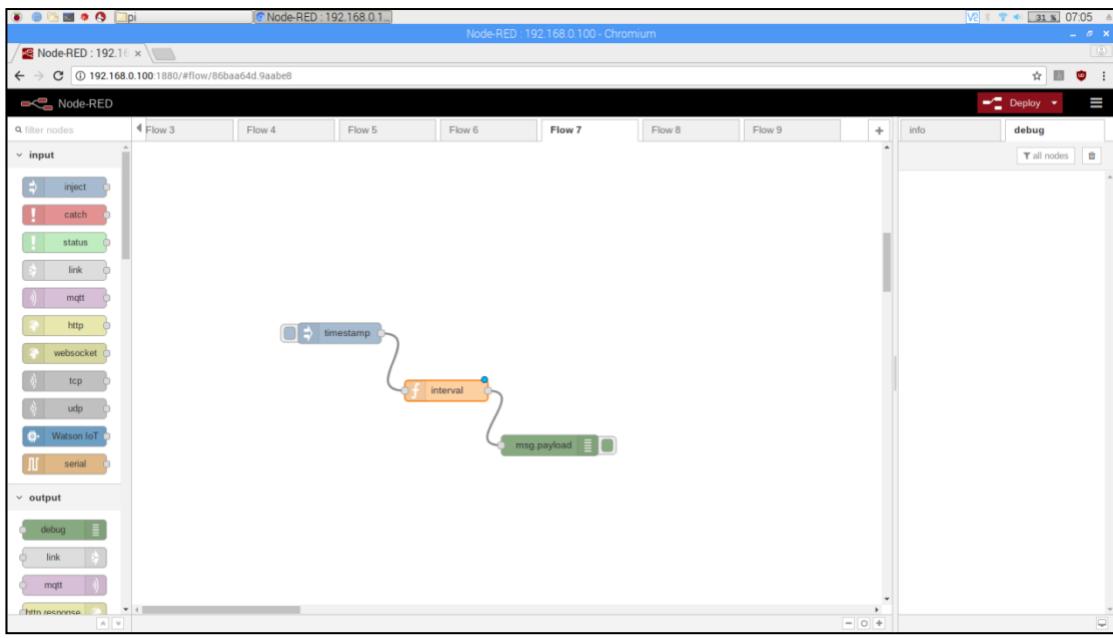
33. Double click the random function and make changes as given below.



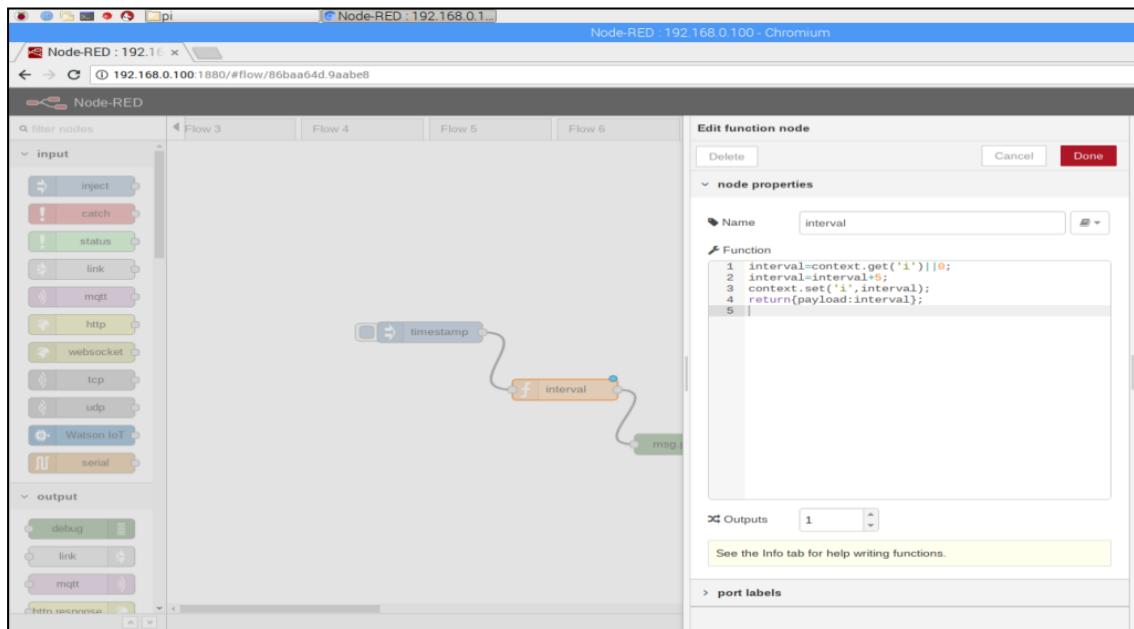
34. Deploy and check the output.



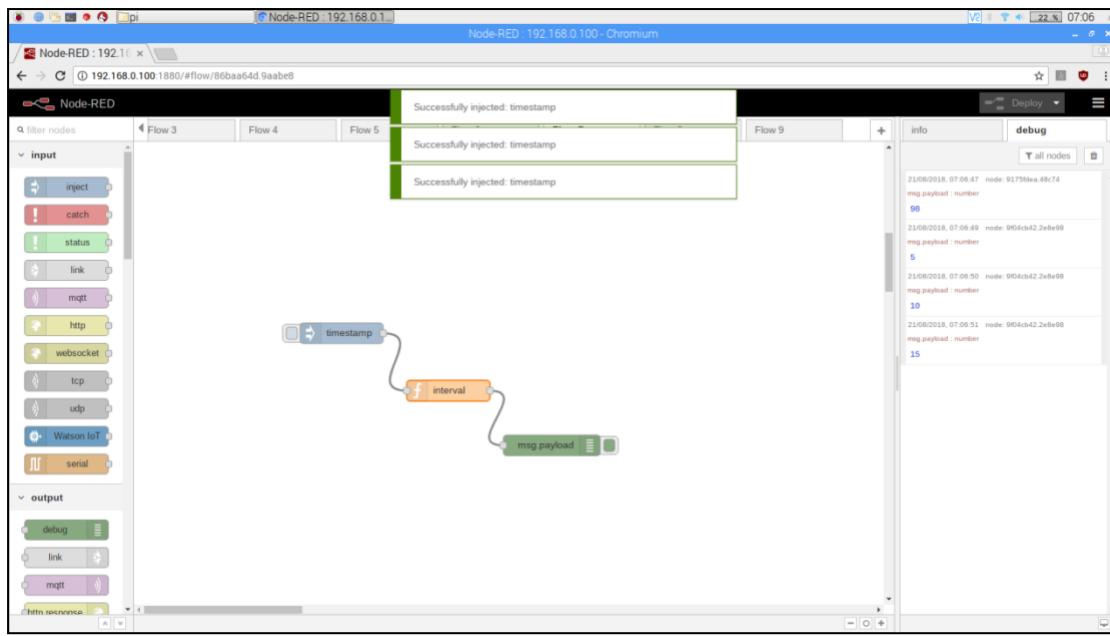
35. Function to get numbers with interval 5. create the flow as given below.



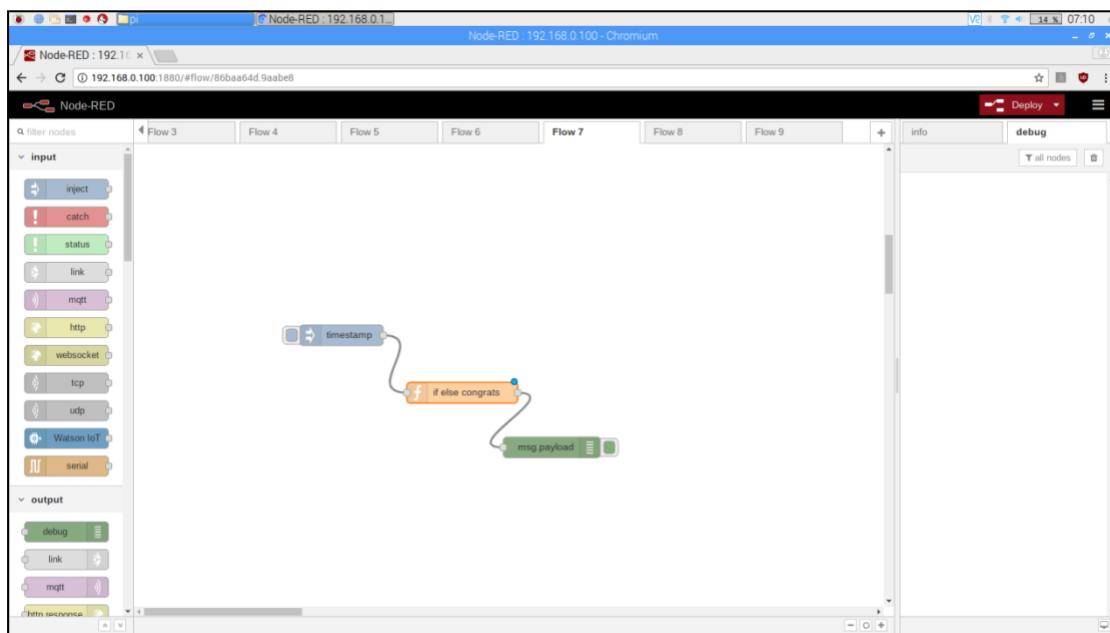
36. Double click the function and write the code as below.



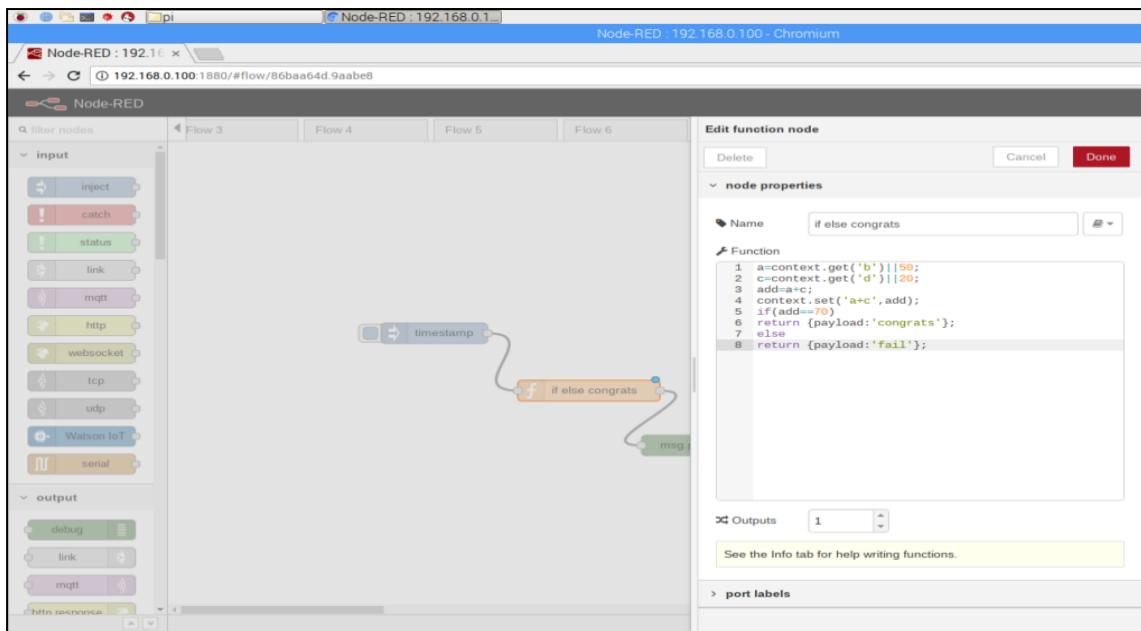
37. Deploy and check the output.



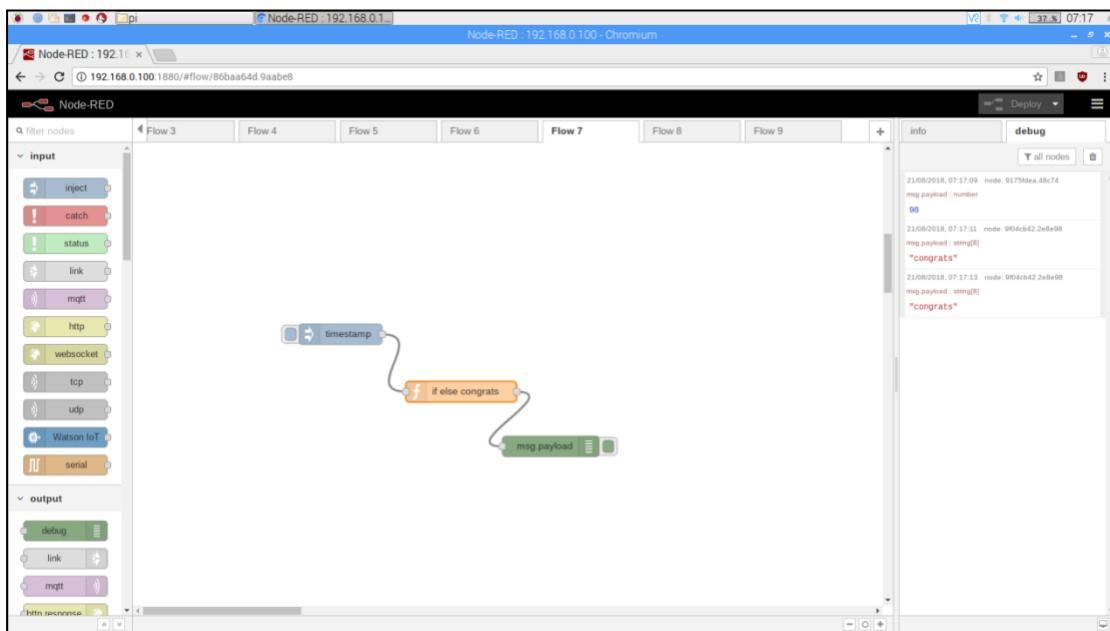
38. Function to print message “congrats” if marks are equal to 70. Create the flow as given below.



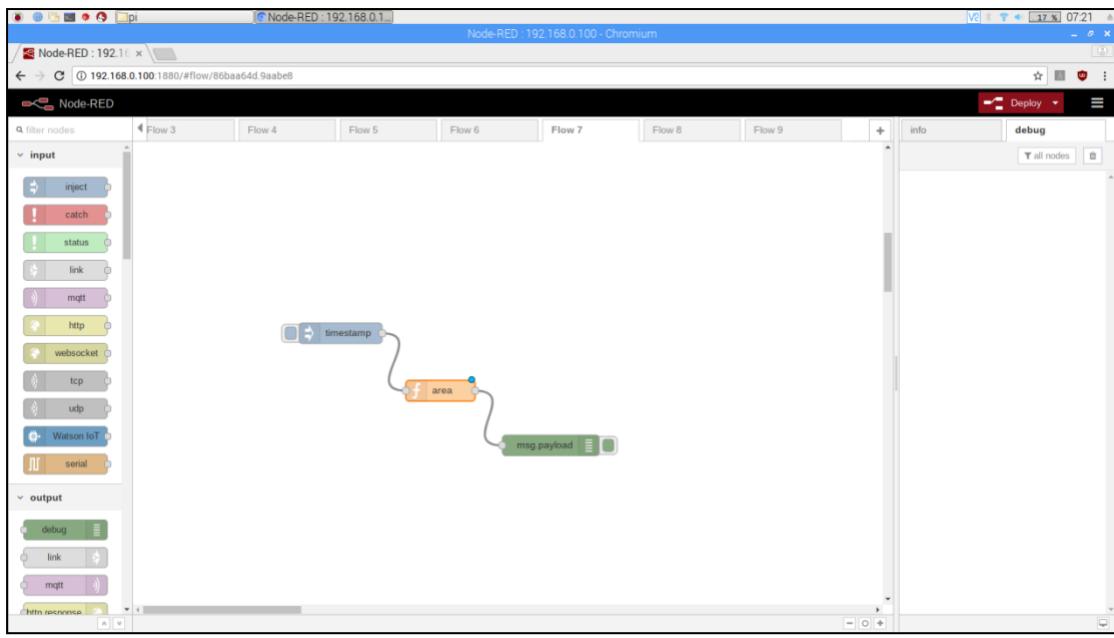
39. By double clicking the function Write the code as given below.



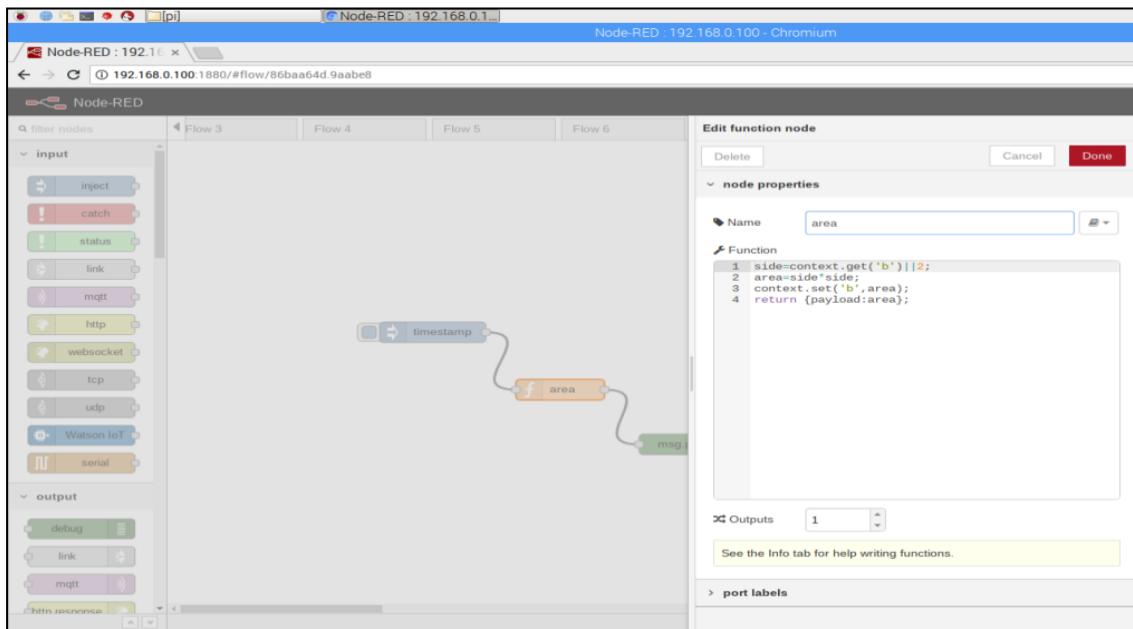
40. Deploy and check the output.



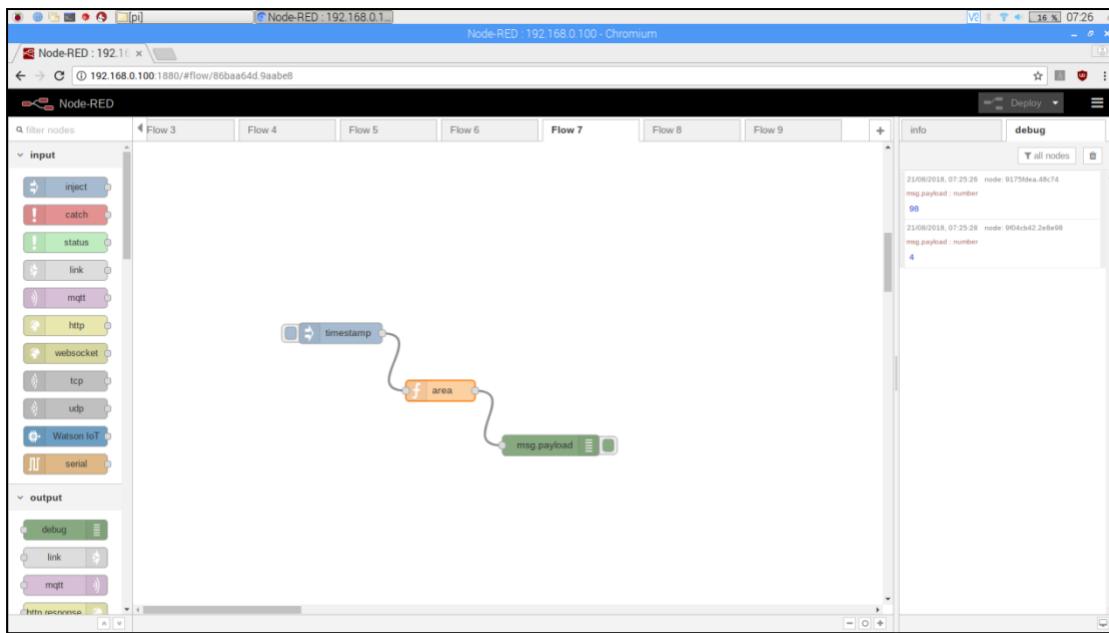
41. Function to calculate area of square. Create flow as given below



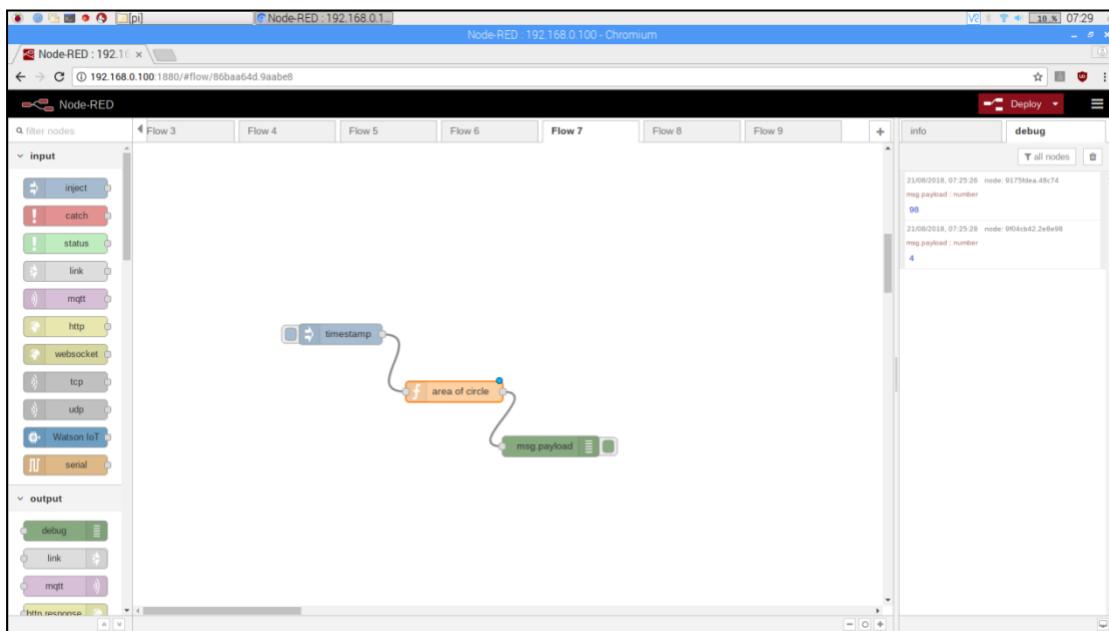
42. Double click the function and write the code.



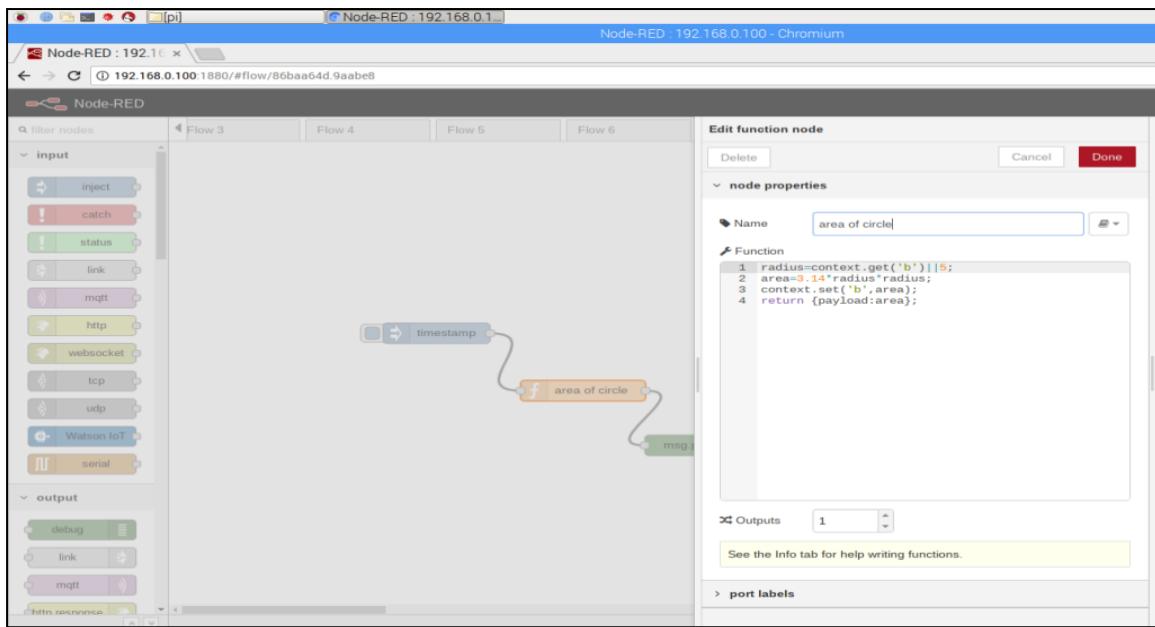
43. Deploy and check the result.



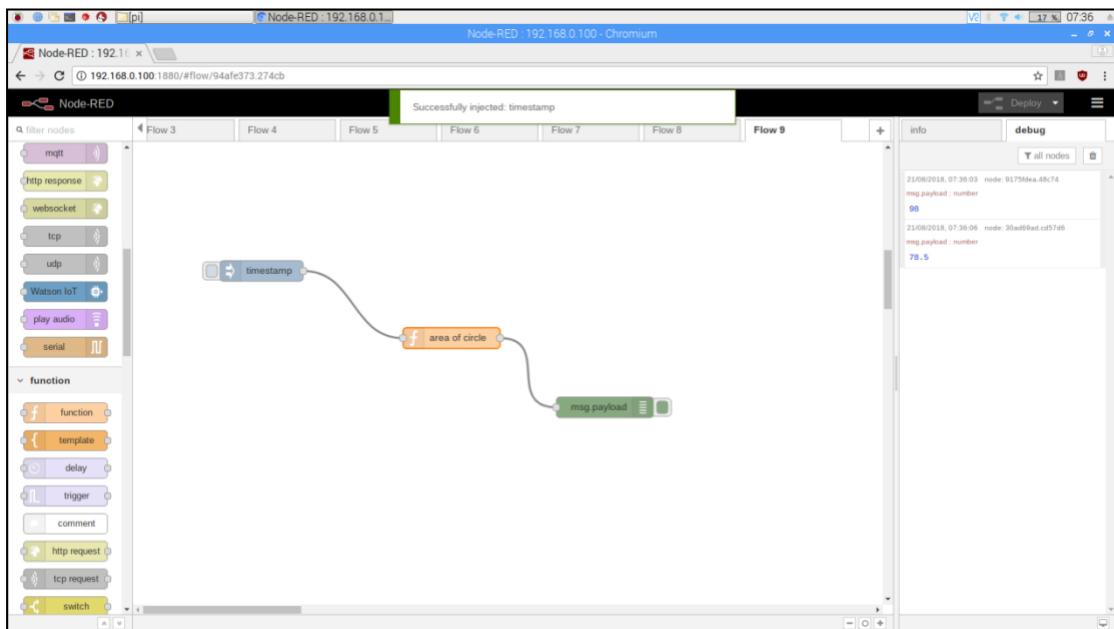
44. Function to calculate area of circle. Create the flow as given below.



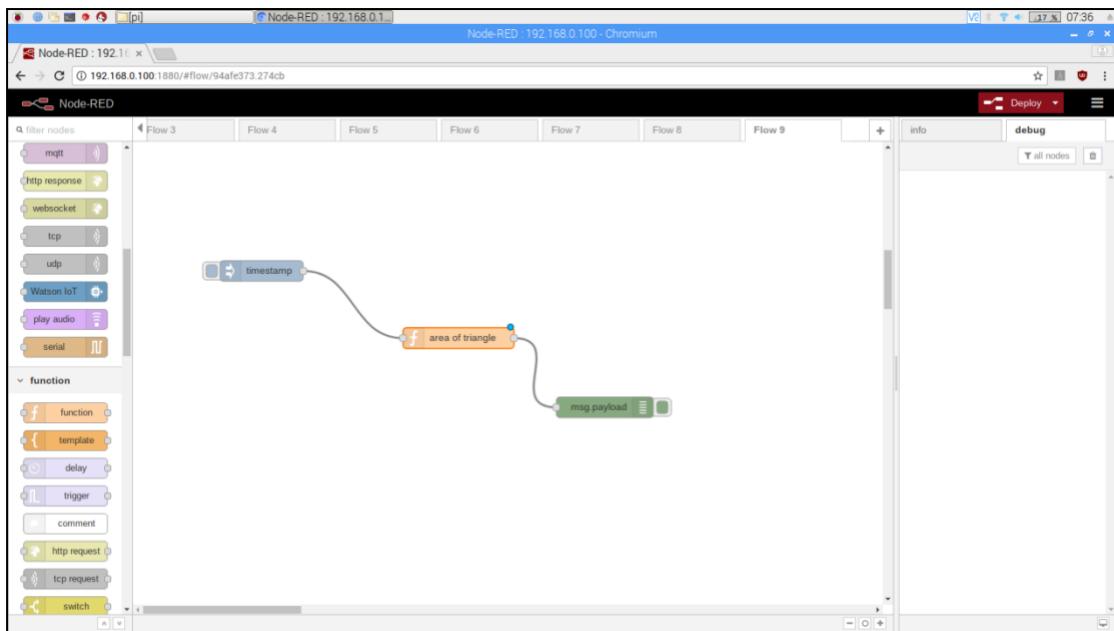
45. Double click the function and write the below code.



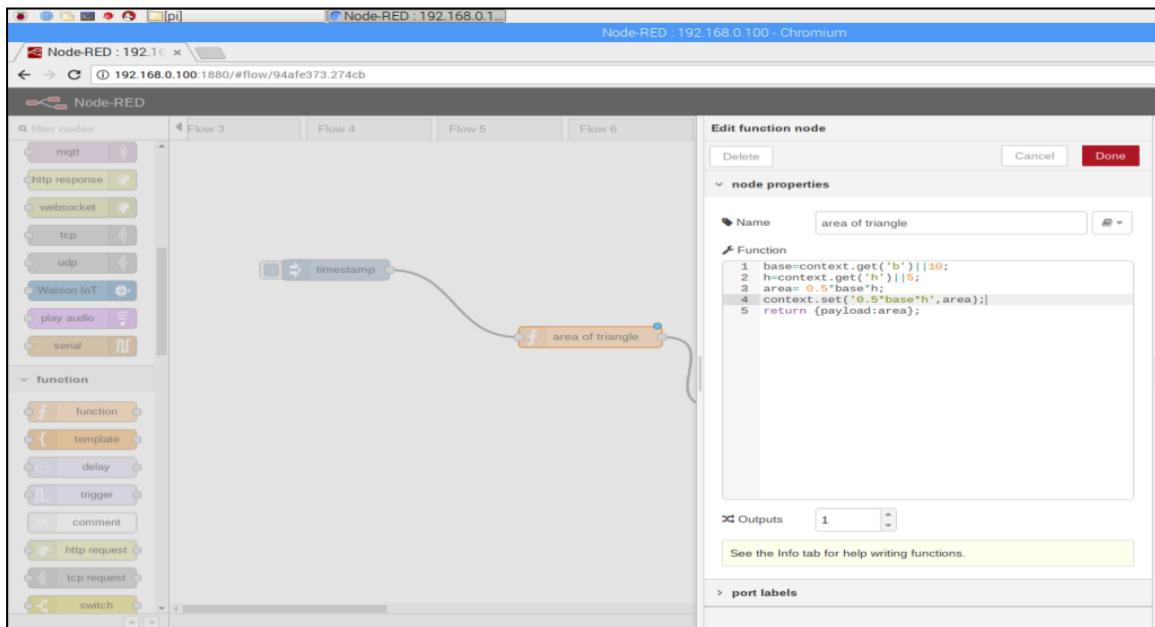
46. Deploy and check the output.



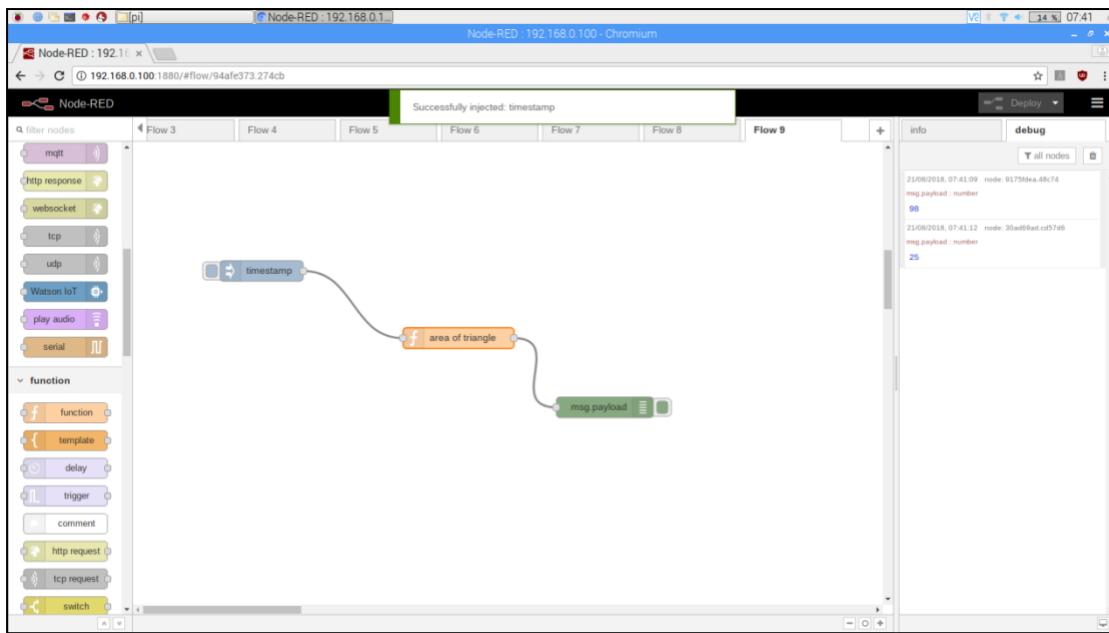
47. Function to calculate the area of triangle. Create the flow as given below.



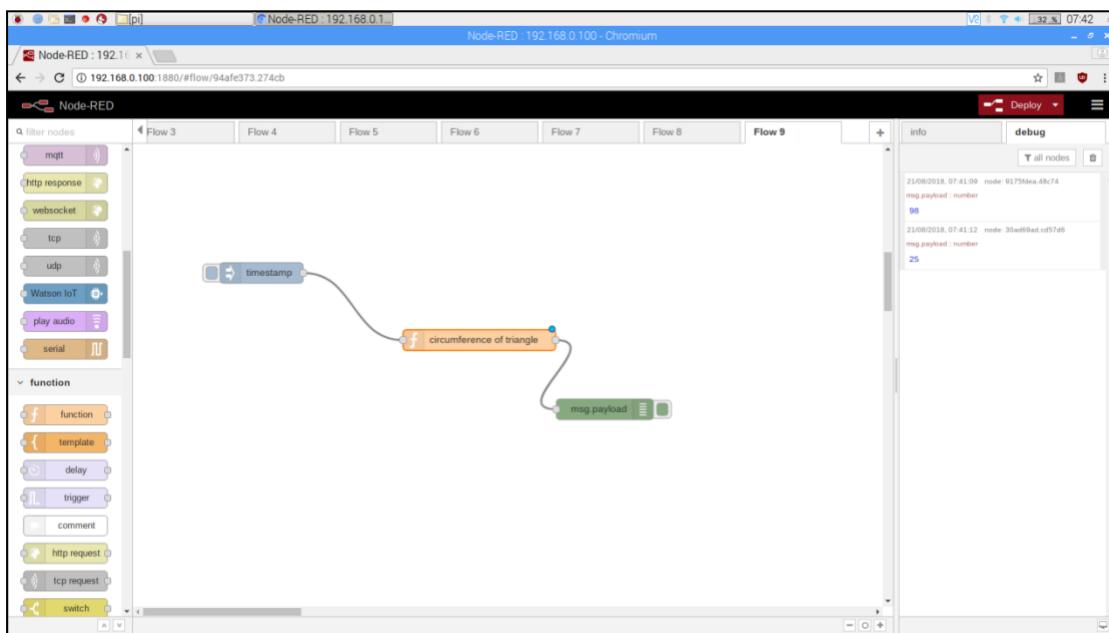
48. Double click the function and add the code .



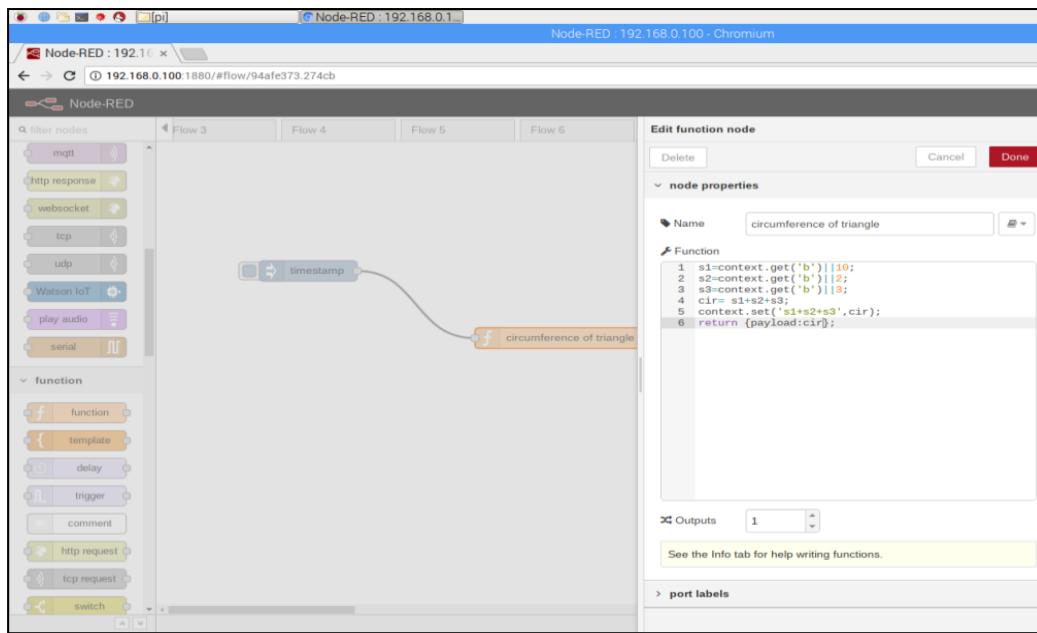
49. Deploy and check the result.



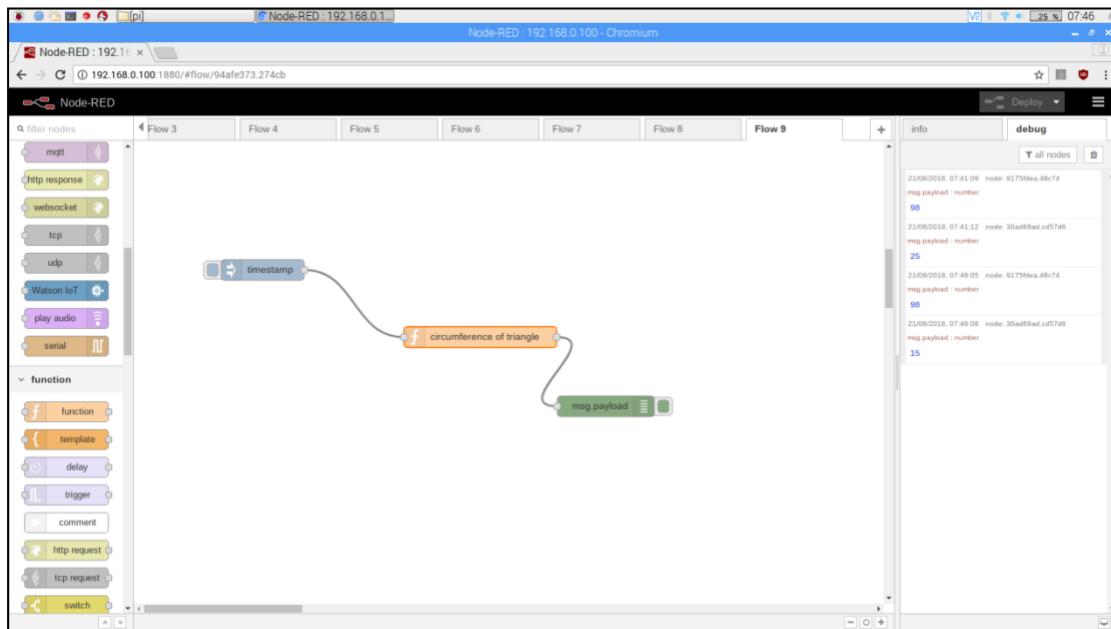
50. Function to calculate the perimeter of triangle. Create the flow as given below.



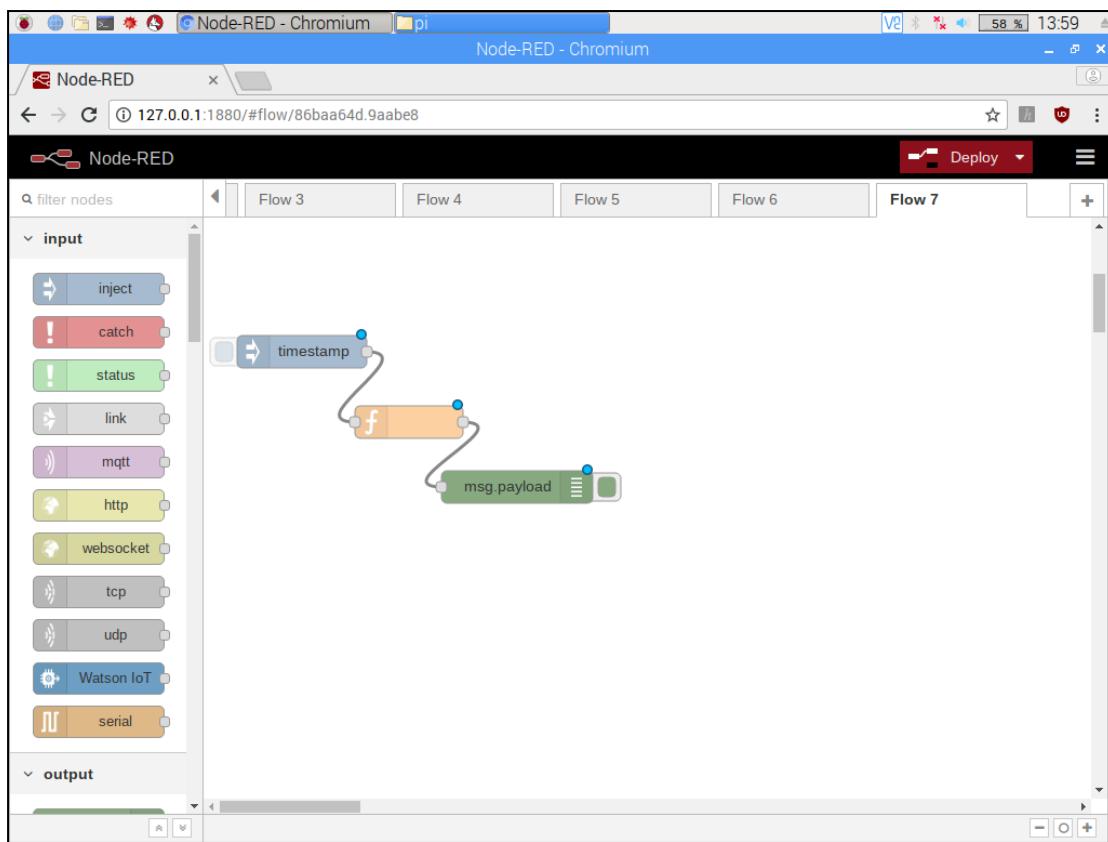
51. Double click the function and add the code given below.



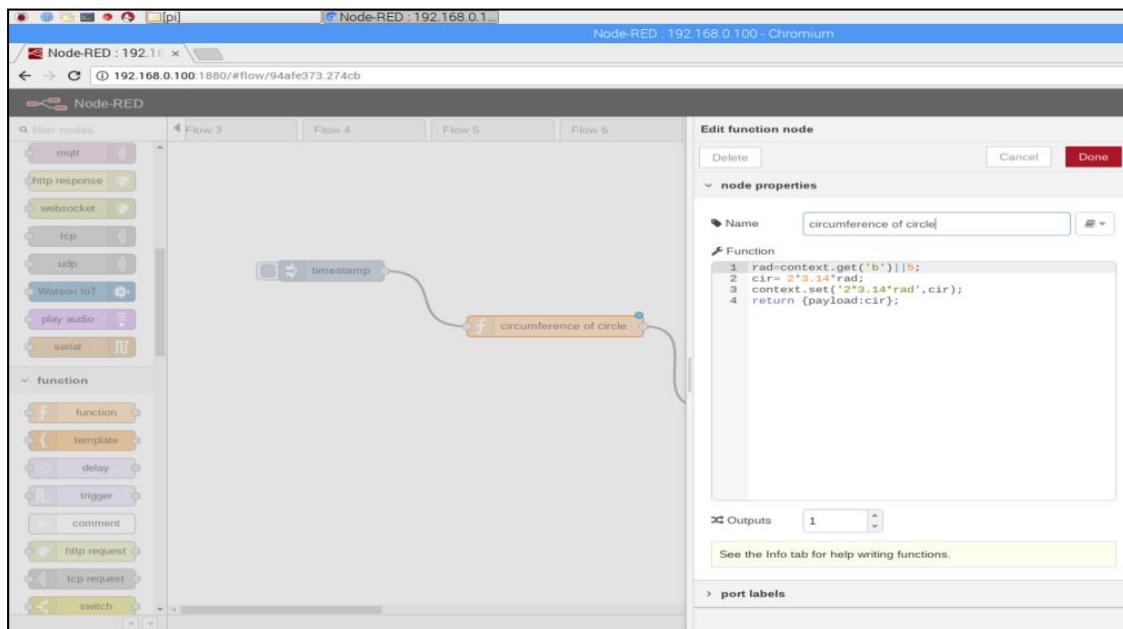
52. Deploy and check the output.



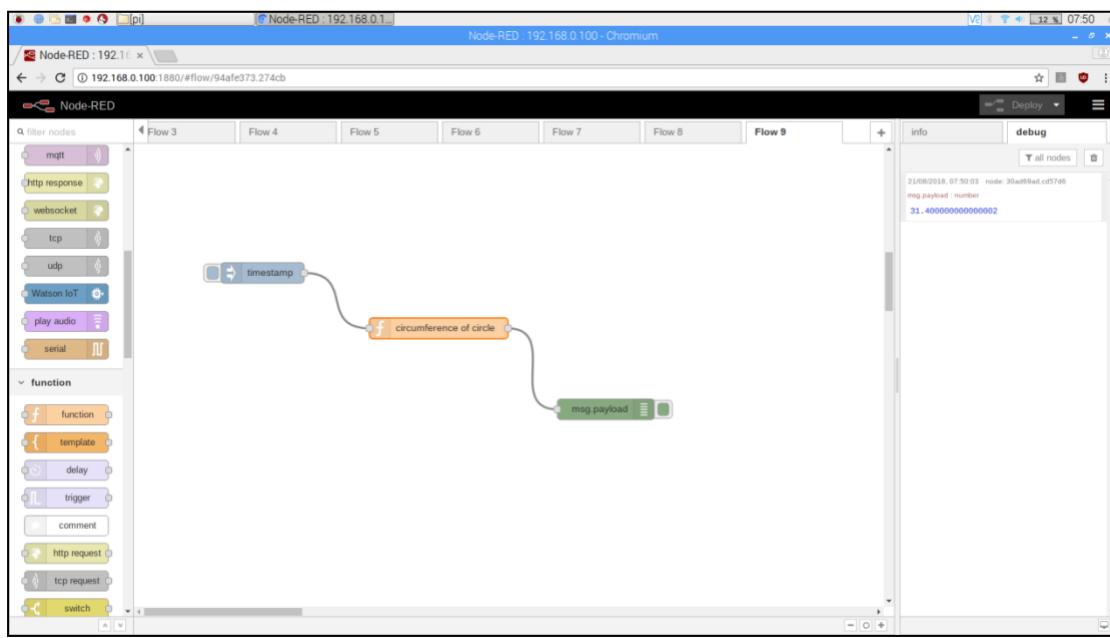
53. Function to calculate the circumference of circle. Create the flow as given below.



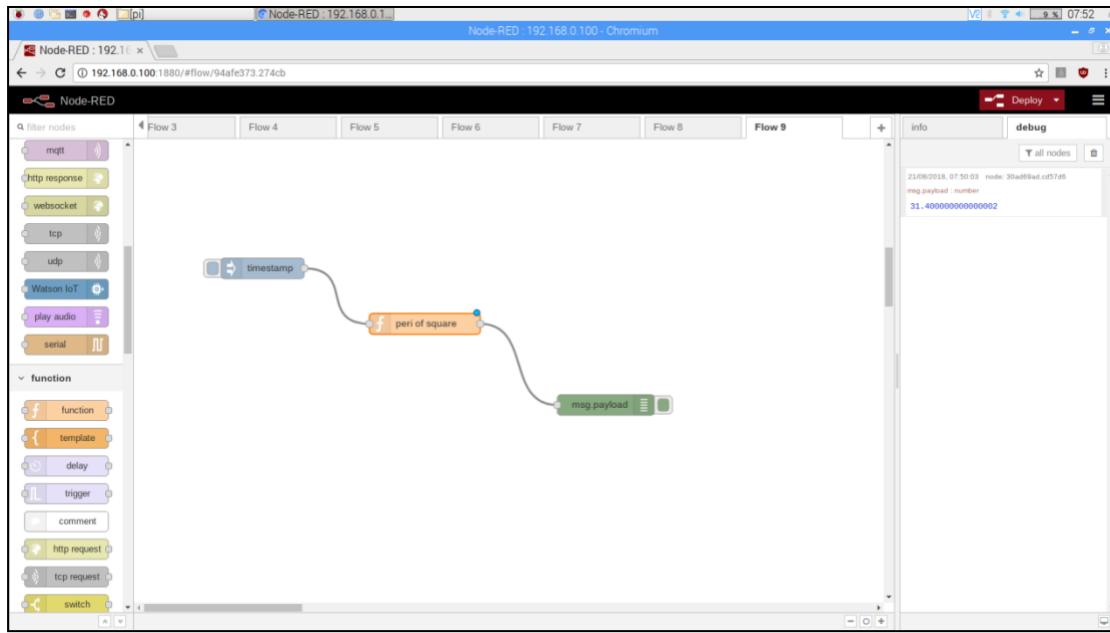
54. Double click the function and add the code.



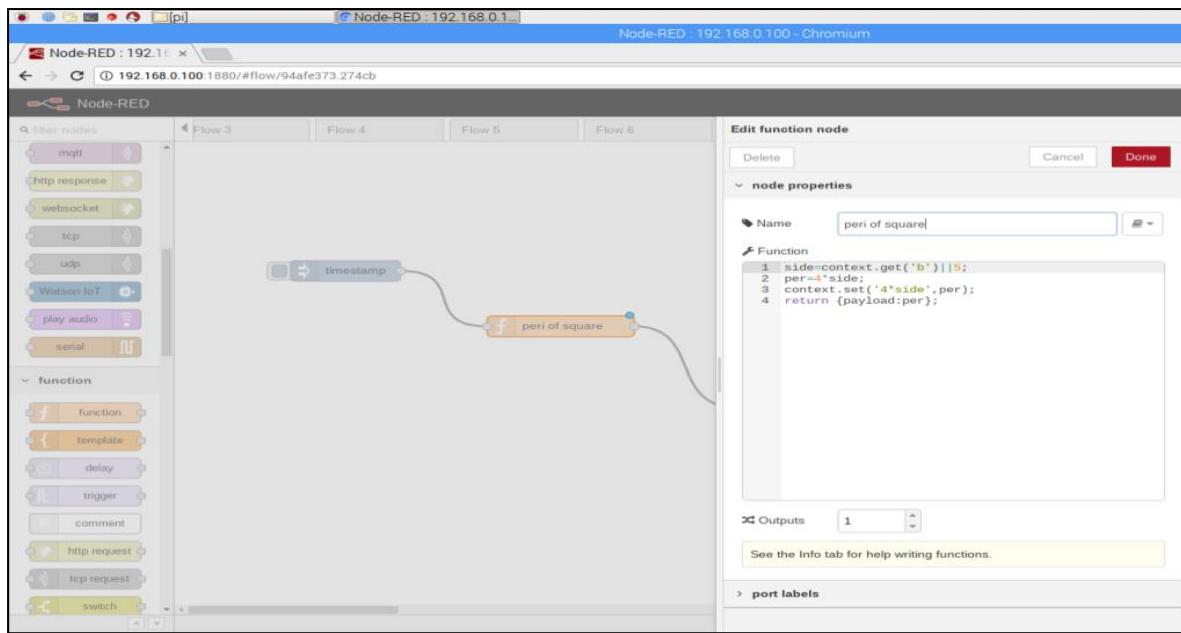
55. Deploy and check the output.



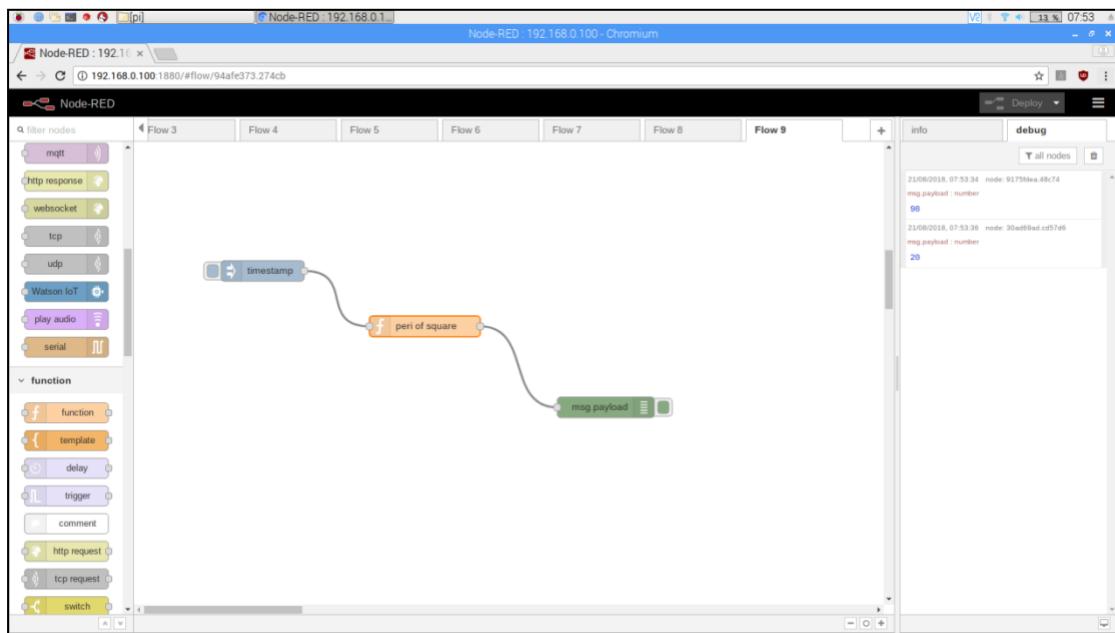
56. Function to calculate the perimeter of square. Create the flow as given below.



57. double click the function and add the code.



58. Deploy and check the output.



Practical 6:

Aim : How to install dashboard in NodeRED.

Step 1 : To Install NodeRED

Commands to Install NodeRED

Type the following commands in terminal

- sudo apt-get update
- sudo apt-get upgrade
- sudo reboot
- sudo apt-get install npm
- npm -h
- sudo npm l -g npm@2x

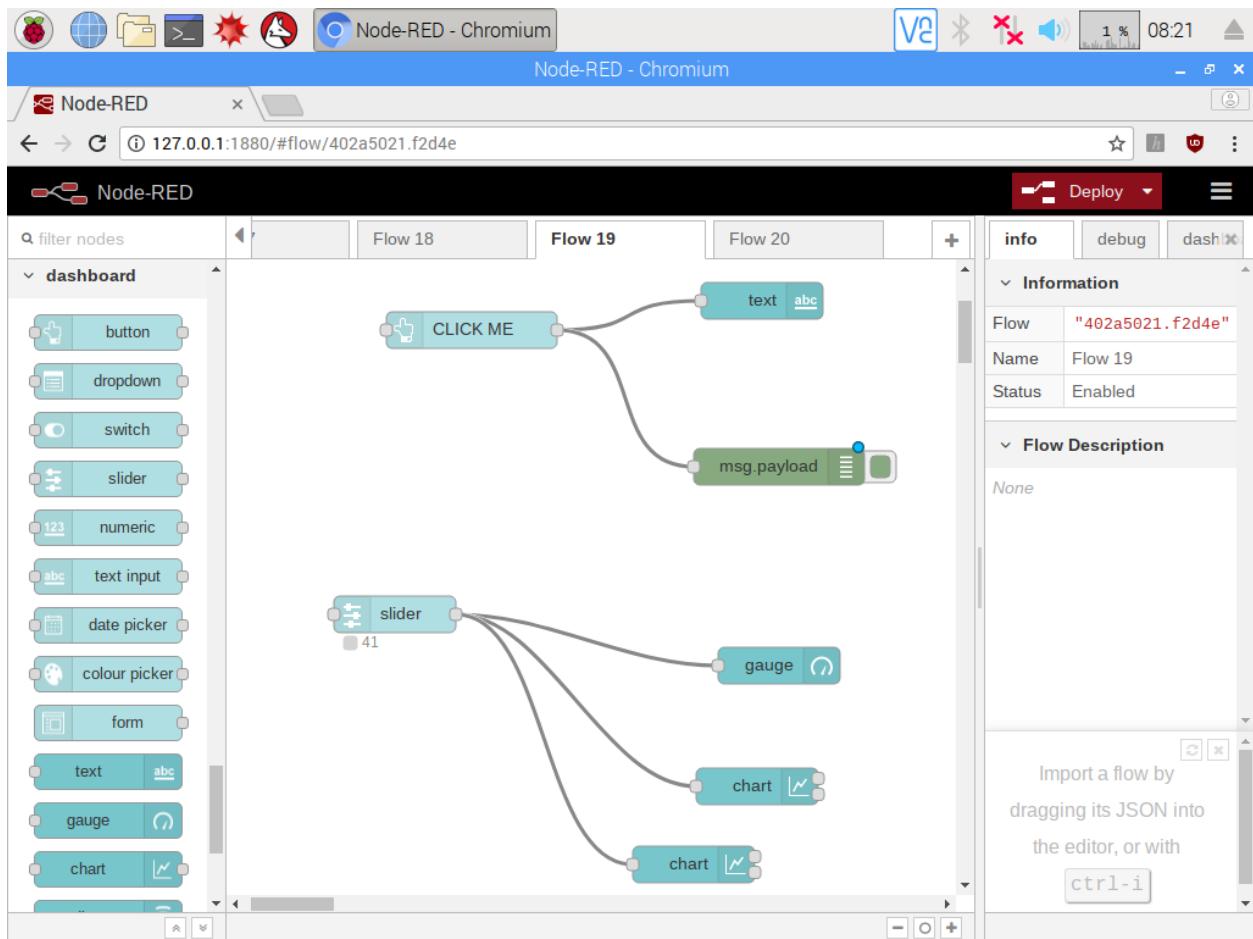
Step 2 : Go to node red and check “MANAGE PALLETE”

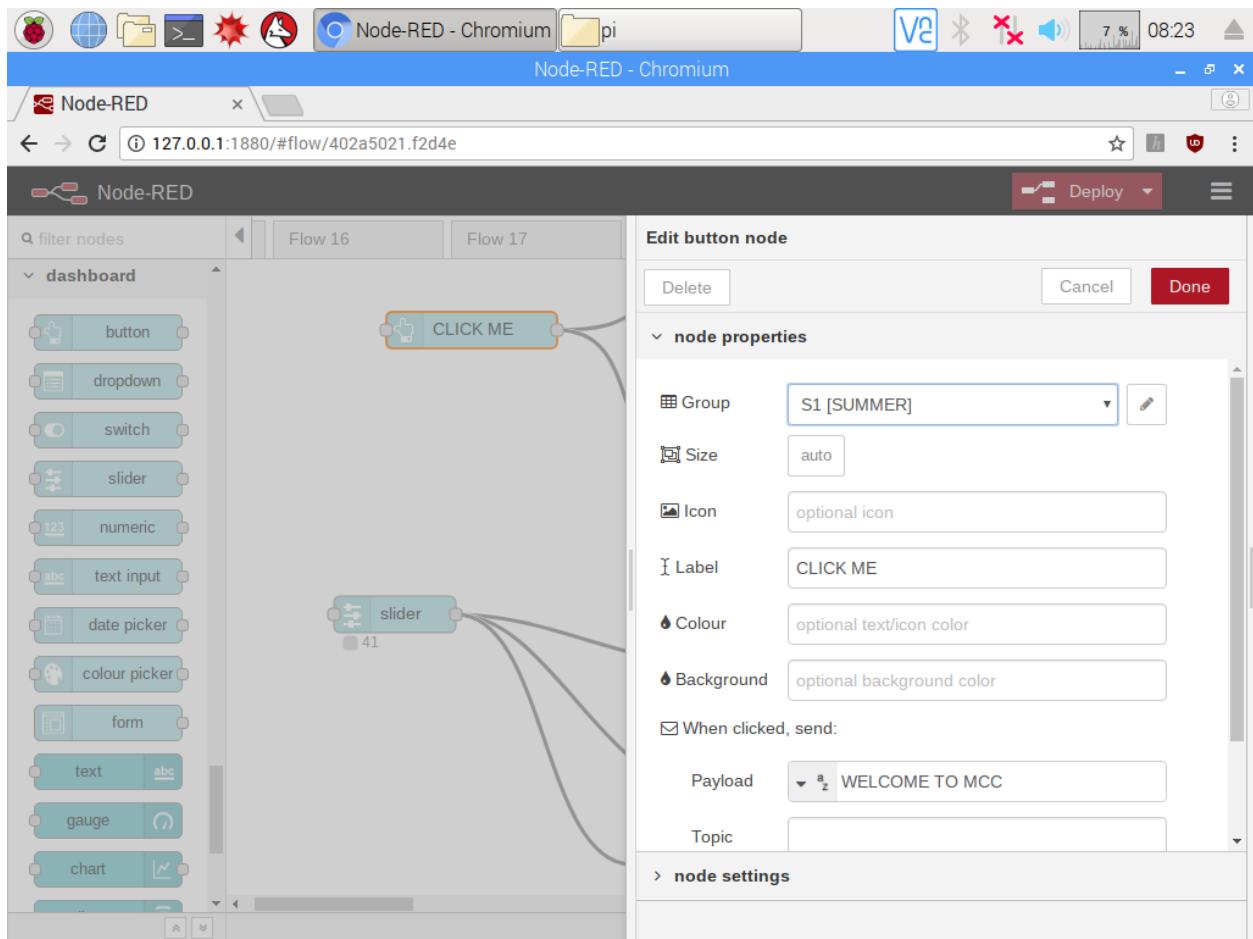
Step 3 : Go to install

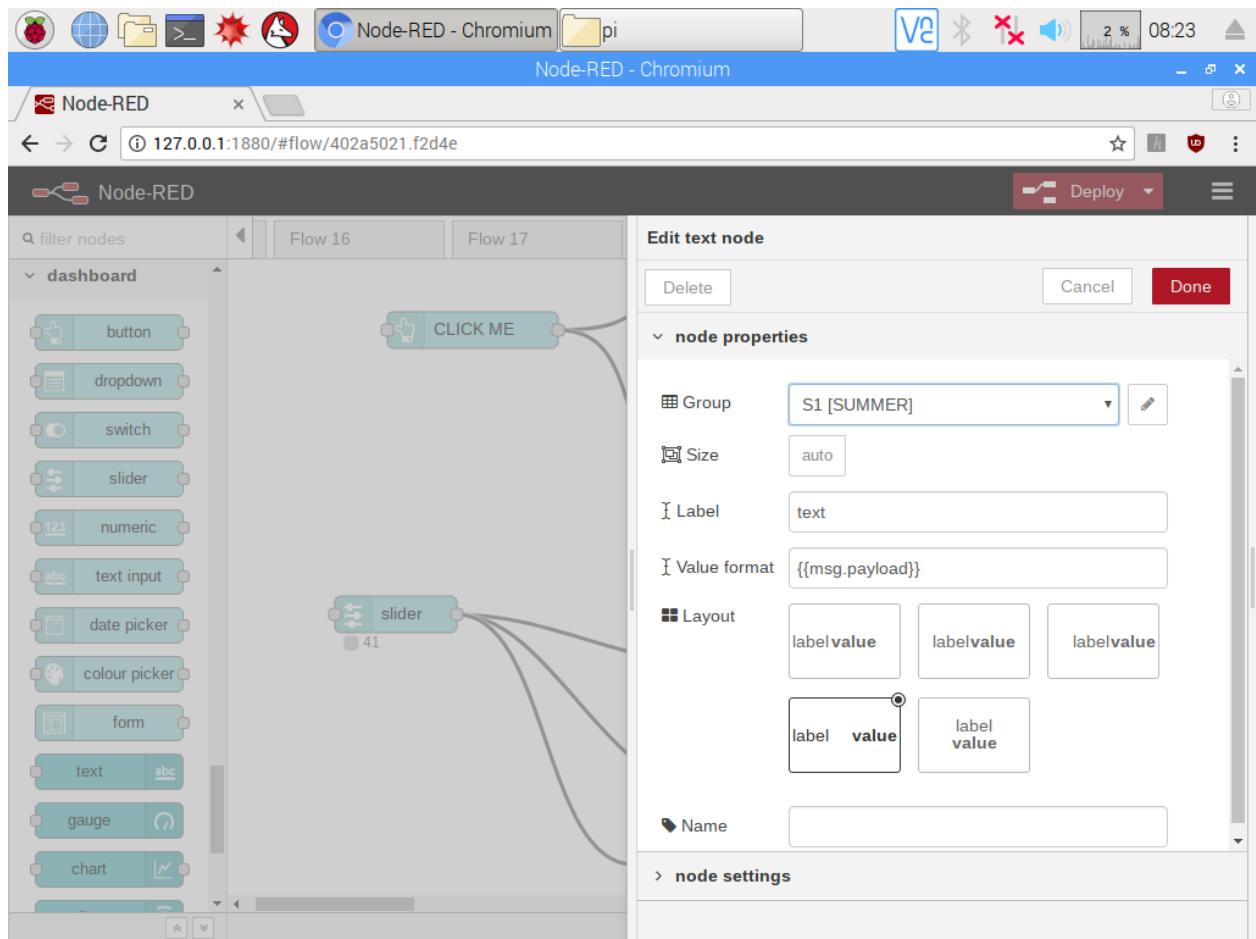
Step 4 : Search dashboard and install

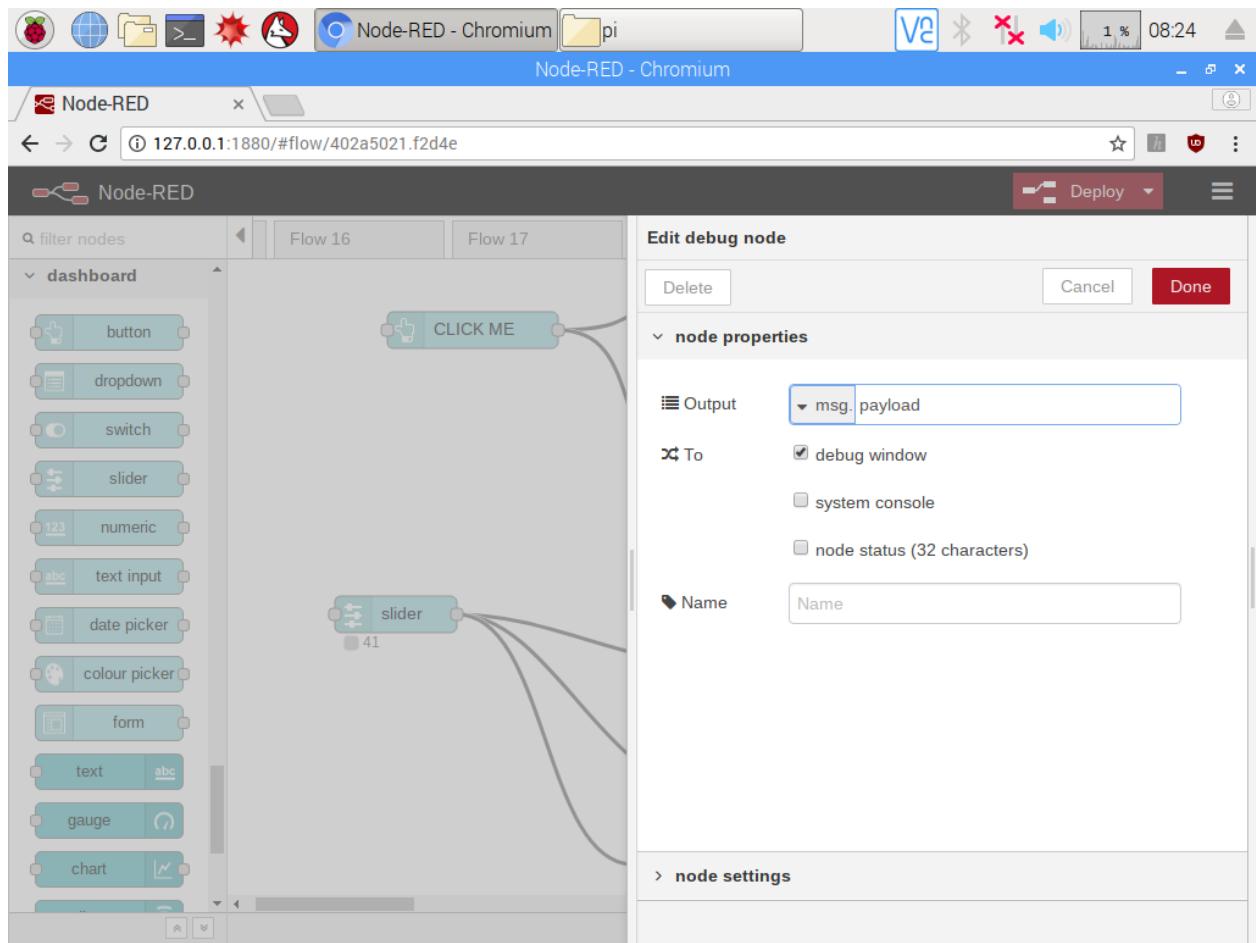
Step 5 : <http://192.168.0.110:1880/text> or page

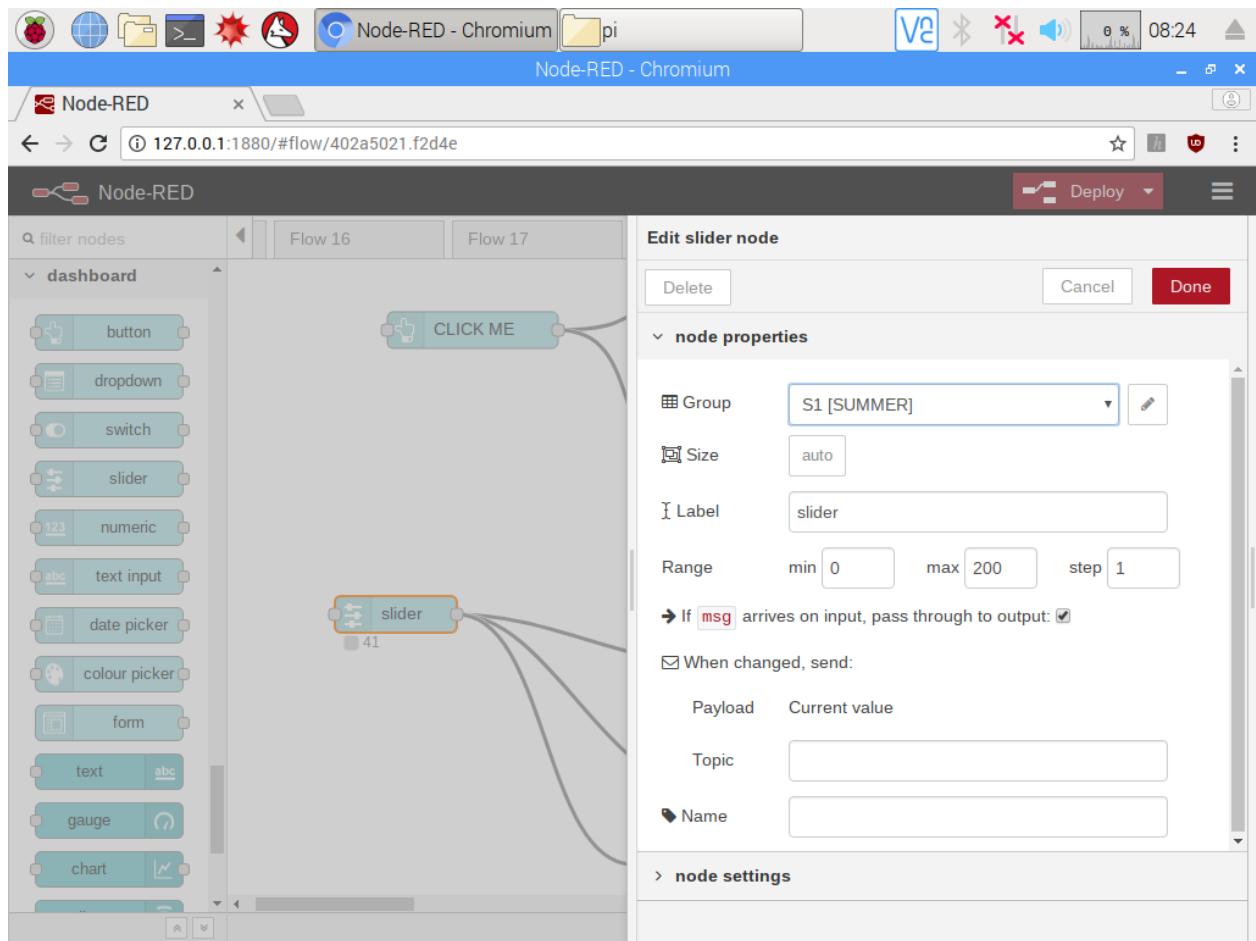
Step 6 : <http://192.168.0.110:1880/ui>

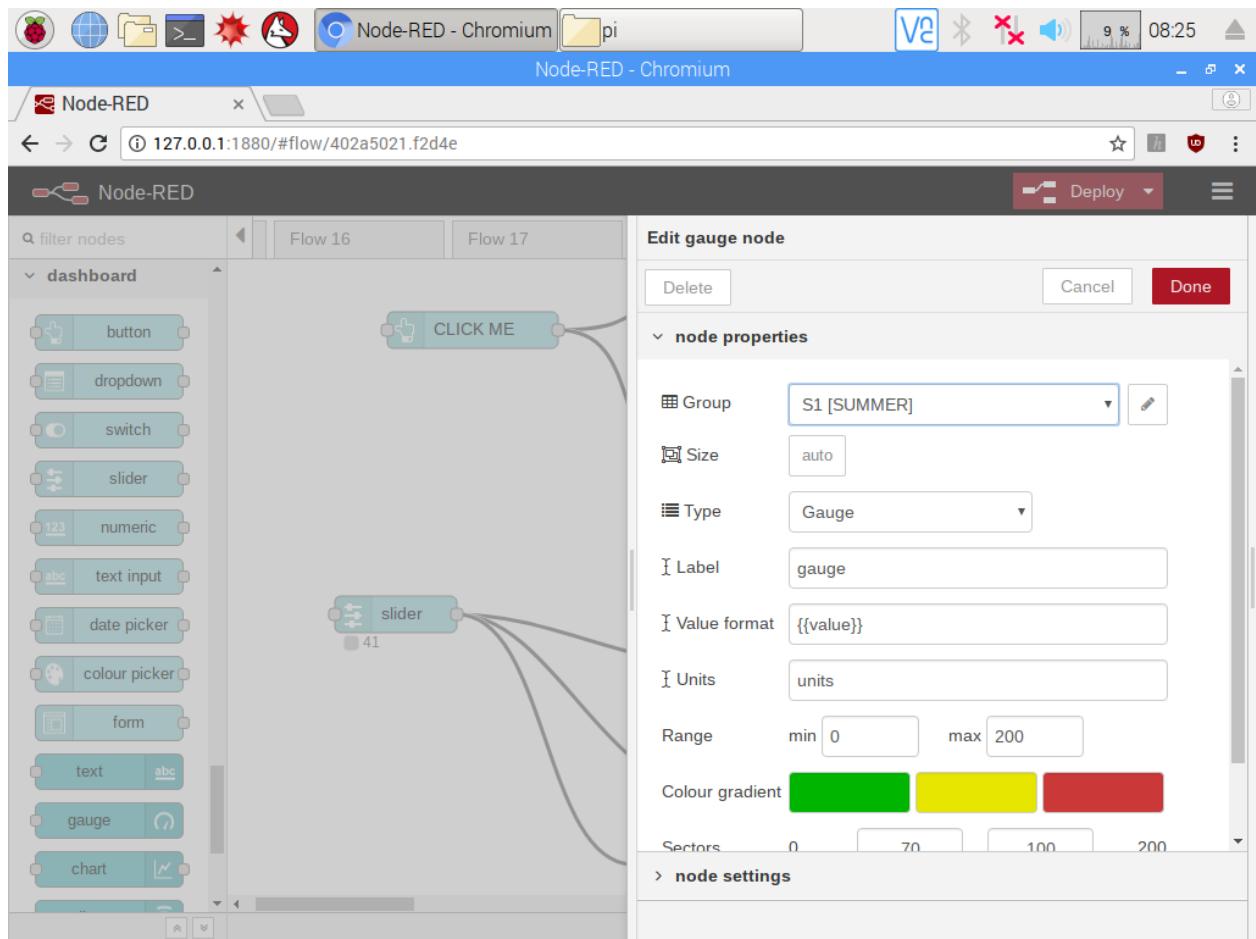


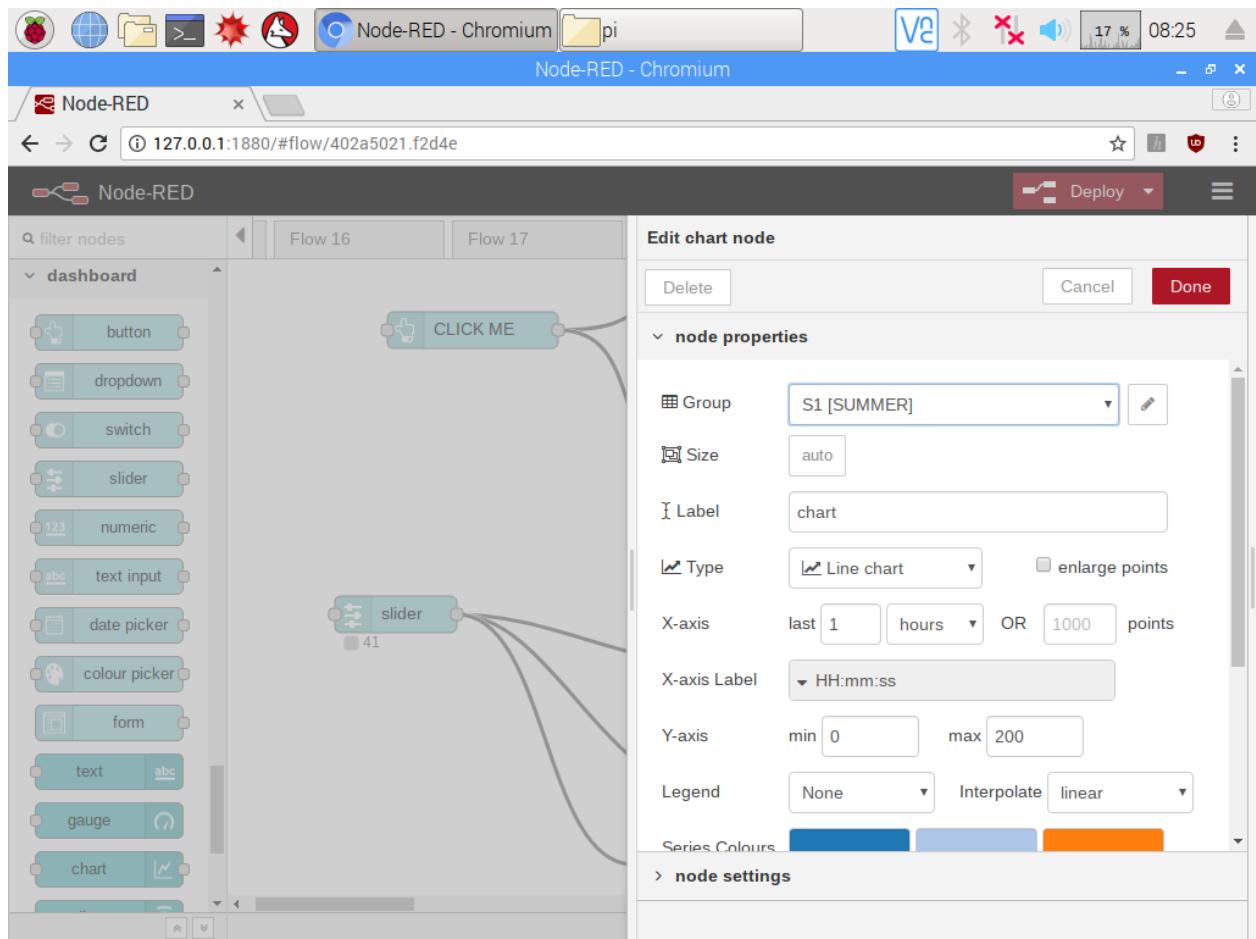


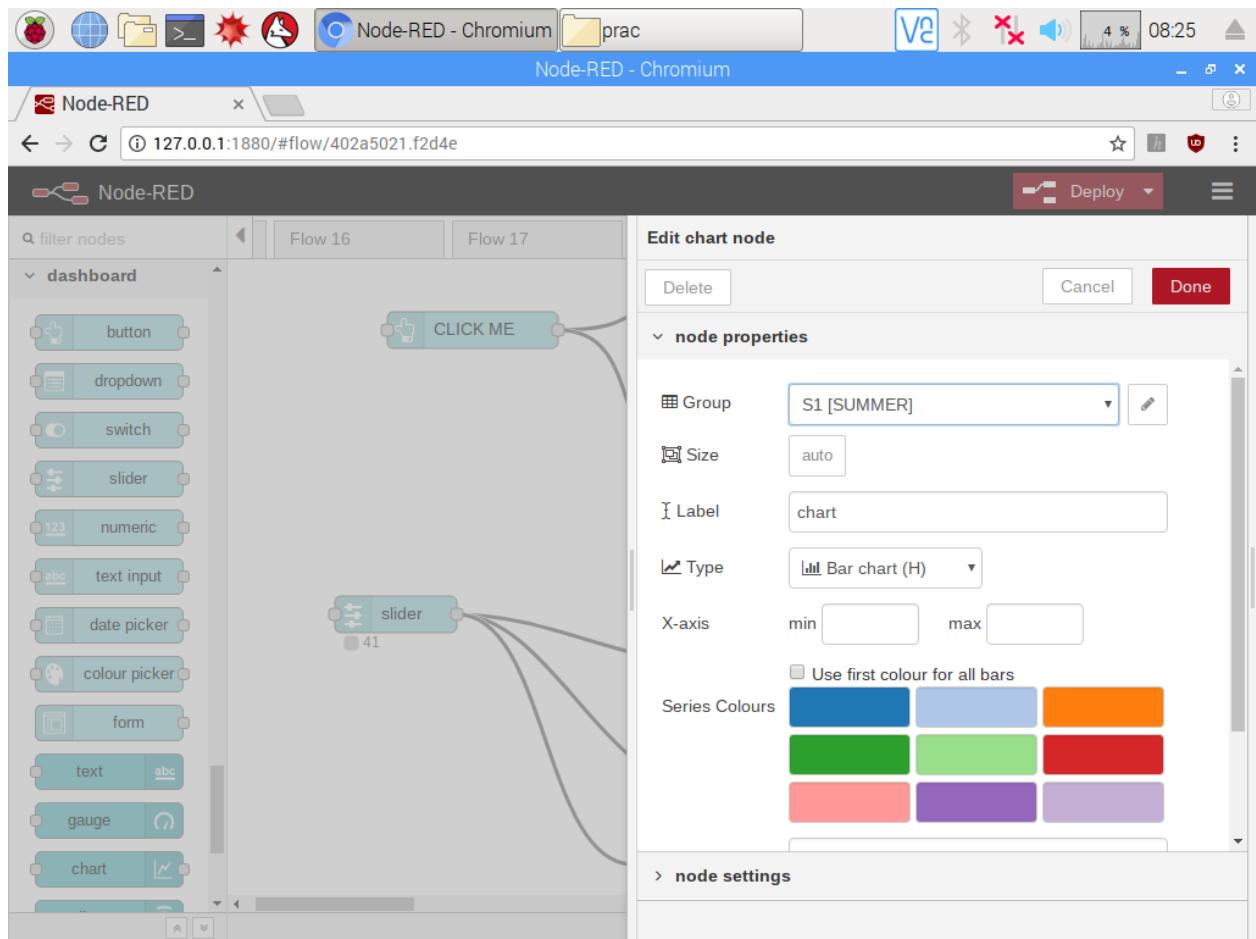


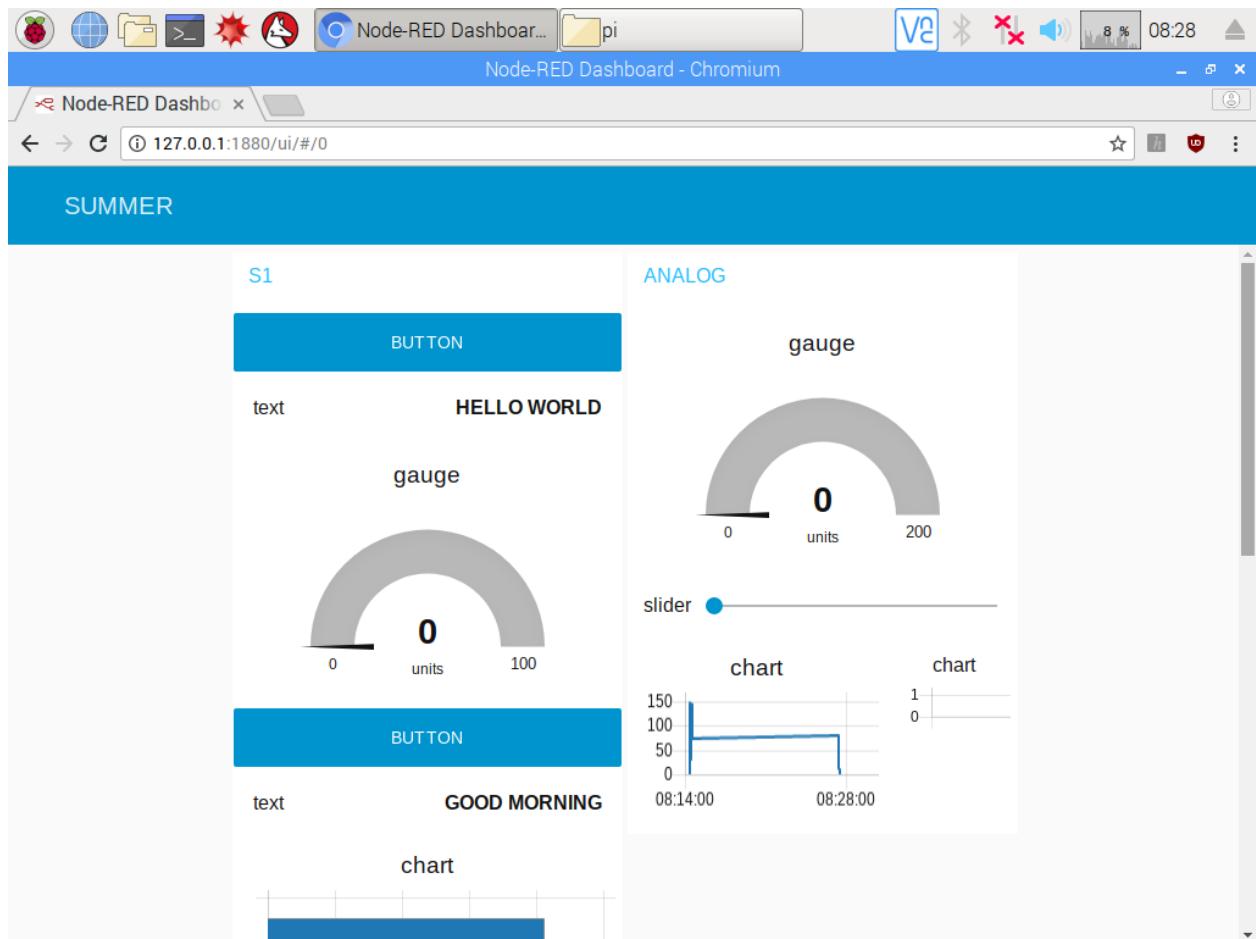


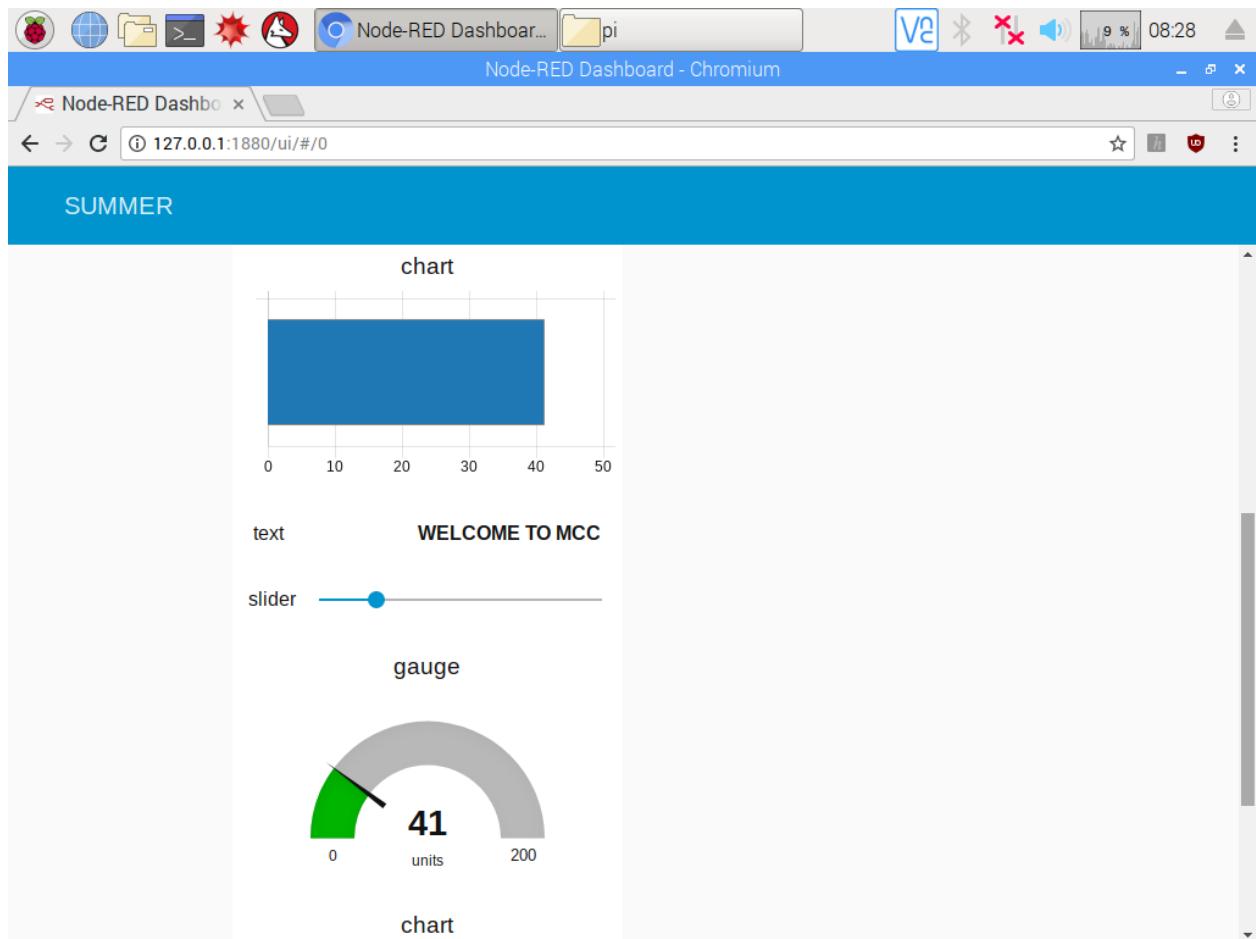


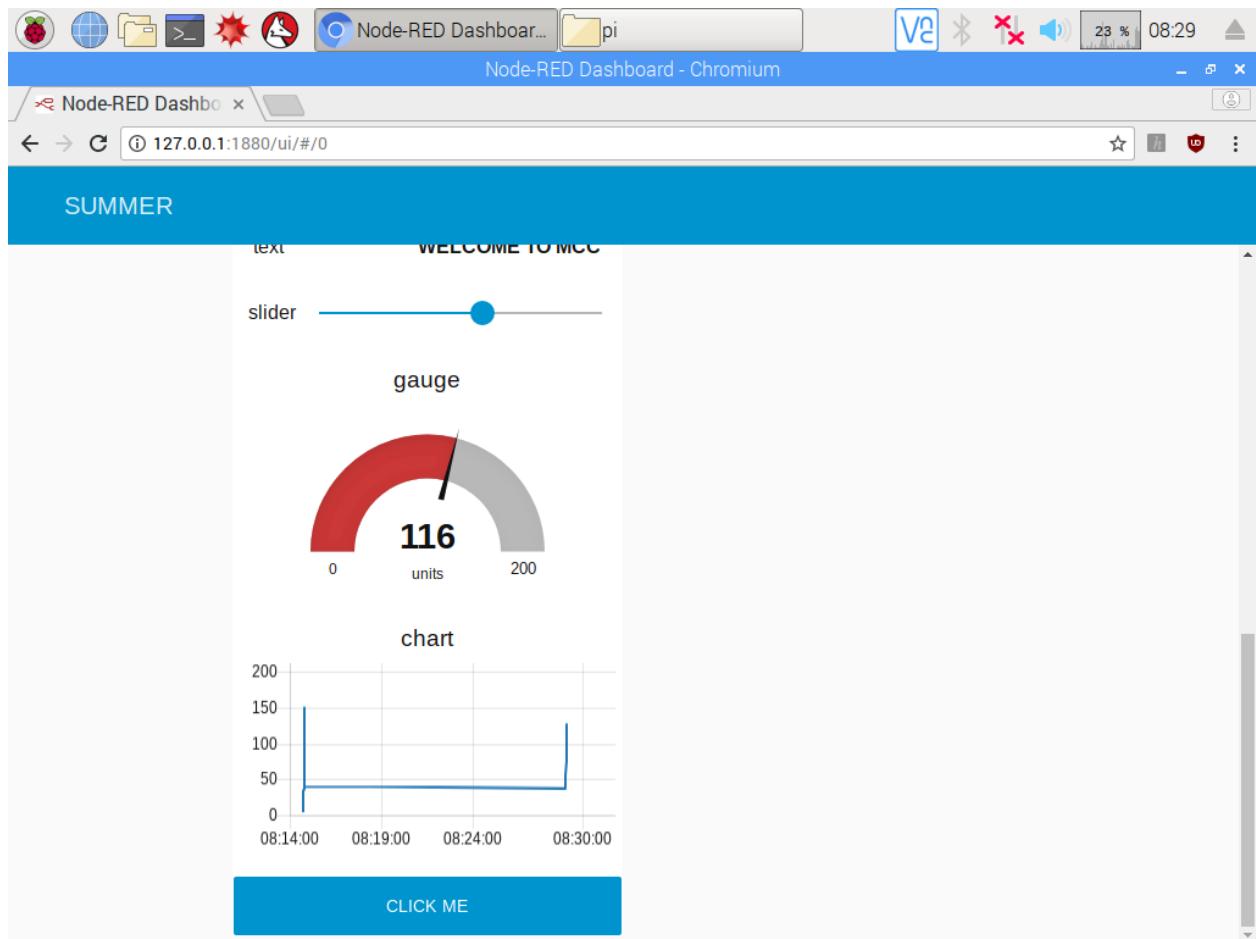


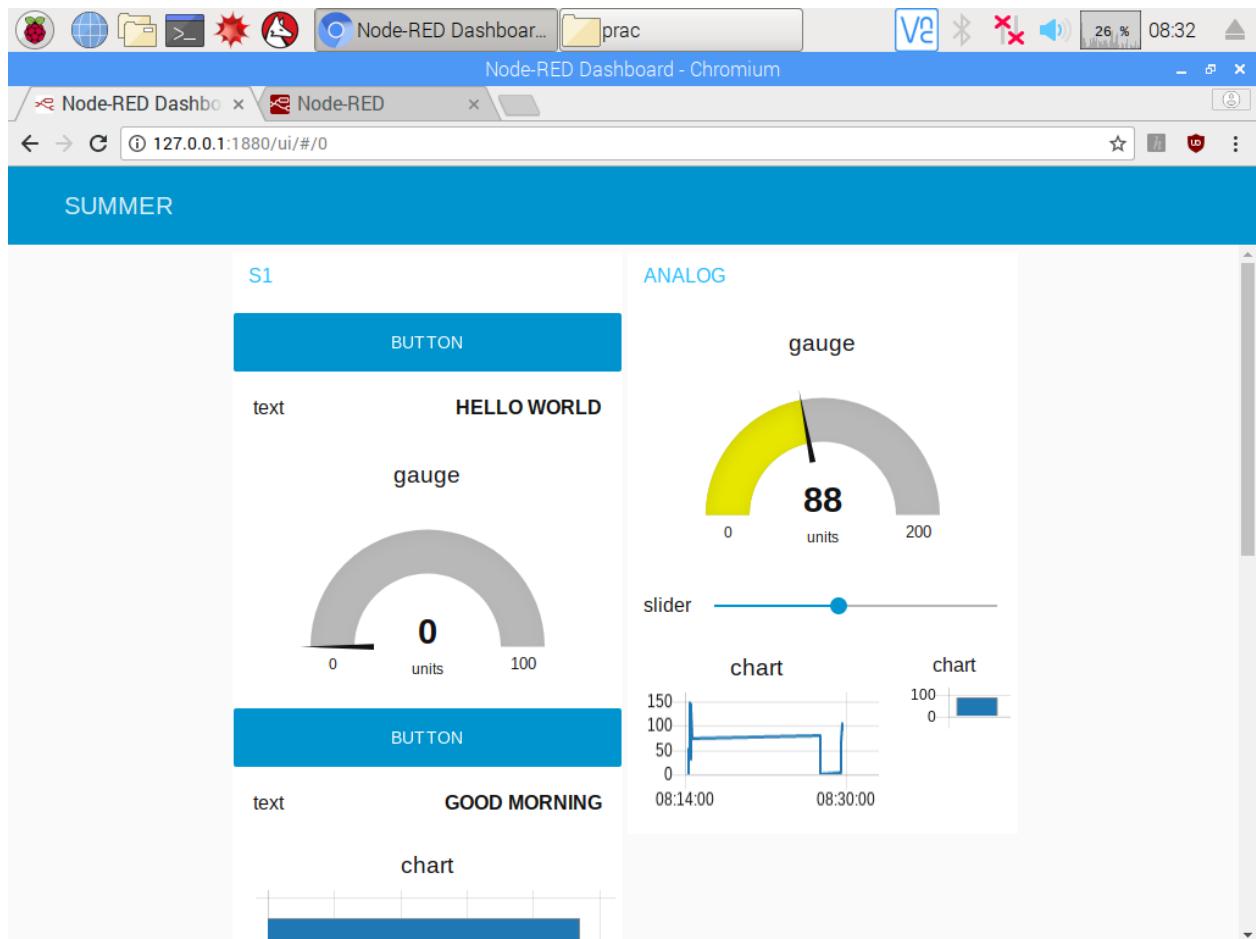












Practical 8:

MQTT

Aim : Set up the mosquitto MQTT server and client and write a python script to communicate data between Pi's.

Step 1 :

Update the system

Sudo apt-get update

Step 2 :

Update the system repositories

- a) Sudo wget <http://repo.mosquitto.org/debian/mosquitto-repo.gpg.key>
- b) Sudo apt-key add mosquito-repo.gpg.key
- c) cd/etc/apt/sources.list.d/
- d) sudo wget <http://repo.mosquitto.org/debian/mosquitto-wheezy.list>
- e) sudo apt-get update
- f) sudo apt-get install mosquitto

step 3:

install three parts of mosquitto proper

➔ sudo apt-get install mosquitto mosquitto-client python-mosquitto

step 4:

stop the server

➔ sudo /etc/init.d/mosquitto stop

step 5 :

Configuring and starting with the mosquitto

➔ sudo nano/etc/mosquitto/mosquitto conf
➔ log-dest topic
➔ log-type error
➔ log-type warning
➔ log-type notice
➔ log-type information

- ➔ connection_message true
- ➔ log-timestamp true
- ➔ include_dir/etc/mosquitto/conf.d

step 6 : Now start server

- ➔ sudo etc/init.d/mosquitto start

step 7 : Open two terminals using putty

a) At terminal 1 type the type following

- ➔ mosquitto_sub -d -t hello/world

b) At terminal 2 type following

- ➔ Mosquitto_pub -d -t hello/world -m "good morning"

The screenshot shows a terminal window titled "pi@raspberrypi: ~" with a blue header bar containing various icons. The main area of the terminal displays the following text:

```
File Edit Tabs Help
pi@raspberrypi:~ $ sudo /etc/init.d/mosquitto start
[ ok ] Starting mosquitto (via systemctl): mosquitto.service.
pi@raspberrypi:~ $ mosquitto_sub -d -t hello/world
Client mosqsub/15657-raspberry sending CONNECT
Client mosqsub/15657-raspberry received CONNACK
Client mosqsub/15657-raspberry sending SUBSCRIBE (Mid: 1, Topic: hello/world, QoS: 0)
Client mosqsub/15657-raspberry received SUBACK
Subscribed (mid: 1): 0
Client mosqsub/15657-raspberry received PUBLISH (d0, q0, r0, m0, 'hello/world',
... (12 bytes))
good morning
Client mosqsub/15657-raspberry sending PINGREQ
Client mosqsub/15657-raspberry received PINGRESP
^Z
[1]+  Stopped                  mosquitto_sub -d -t hello/world
pi@raspberrypi:~ $
```

A screenshot of a terminal window titled "pi@raspberrypi: ~". The window contains the following text:

```
pi@raspberrypi:~ $ mosquitto_pub -d -t hello/world -m "good morning"
Client mosqpub/15798-raspberry sending CONNECT
Client mosqpub/15798-raspberry received CONNACK
Client mosqpub/15798-raspberry sending PUBLISH (d0, q0, r0, m1, 'hello/world', .
.. (12 bytes))
Client mosqpub/15798-raspberry sending DISCONNECT
pi@raspberrypi:~ $
```

Practical 1-Substitution Cipher Technique-1

AIM: Write Programs to Implement the following Substitution Cipher Techniques:

- 1. Caesar Cipher**
- 2. Monoalphabetic Cipher**

1. CAESAR CIPHER

CODE:

```
def caesarCipher(pt,key):  
    pt=pt.replace(" ","")  
    list1='abcdefghijklmnopqrstuvwxyz'  
    en=""  
    for i in pt.lower():  
        k=(list1.index(i)+key)%26  
        en+=list1[k]  
    print("Encrypted Text:",en)  
    de=""  
    for i in en.lower():  
        k=(list1.index(i)-key)%26  
        de+=list1[k]  
    print("Decrypted Text:",de)  
    pt=input("Enter Plain Text:")  
    key=int(input("Enter Key:"))  
    caesarCipher(pt,key)
```

OUTPUT:

```
Enter Plain Text:welcome
Enter Key:2
Encrypted Text: ygneqog
Decrypted Text: welcome
>>> |
```

2. MONOALPHABETIC CIPHER

CODE:

```
def mono(pt):
    pt=pt.replace(" ","")
    al='abcdefghijklmnopqrstuvwxyz'
    key=input("Enter Key:")
    en=""
    de=""
    for j in pt.lower():
        for i in al:
            if i==j:
                en+=key[al.index(i)]
    for j in en.lower():
        for i in key:
            if i==j:
                de+=al[key.index(i)]
    print("Encrypted Text:",en)
    print("Decrypted Text:",de)
pt=input("Enter Plain Text:")
mono(pt)
```

OUTPUT:

```
Enter Plain Text:welcome
Enter Key:bcdefghijklmnopqrstuvwxyz
Encrypted Text: xfmdpmf
Decrypted Text: welcome
>>> |
```

Practical 2-Substitution Cipher Technique-2

AIM: Write Programs to Implement the following Substitution Cipher Techniques:

1. Vernam Cipher
2. Playfair Cipher
3. Vigenere Cipher

1. VERNAM CIPHER

CODE:

```
def ver(pt,key):  
    pt=pt.replace(" ","")  
    key=key.replace(" ","")  
    al='abcdefghijklmnopqrstuvwxyz'  
    en="  
    de="  
    i=0  
    j=0  
    n=0  
    a=[]  
    b=[]  
    for x in range(0,len(pt)):  
        a.append(0)  
        b.append(0)  
    for l1 in pt.lower():  
        a[i]=al.index(l1)  
        i+=1  
    for l2 in key.lower():  
        b[j]=al.index(l2)  
        j+=1  
    for k in range(0,len(pt)):
```

```

s1=(a[k]+b[k])%26
en+=al[s1]
print("Encrypted Text is:",en)
for k in range(0,len(pt)):
    n=n+1
    s2=al.index(en[k])-al.index(key[n-1])
    de=de+al[s2%26]
print("Decrypted Text is:",de)
pt=input("Enter Plain Text:")
key=input("Enter a key of same number of letters as that of plain text:")
ver(pt,key)

```

OUTPUT:

```

Enter Plain Text:deadpool
Enter a key of same number of letters as that of plain text:selenium
Encrypted Text is: vilhwix
Decrypted Text is: deadpool
>>>

```

2. PLAYFAIR CIPHER

CODE:

```

from itertools import product
from re import.findall
array=['a','b','c','d','e','f','g','h','i','j','k','l','m','n','o','p','q','r','s','t','u','v','w','x','y','z']
def datalist_normal(key):
    key=key.replace(" "," ")
    key=key.lower()
    list1=list()
    for i in range(len(key)):
        char=key[i]
        if char not in list1:
            if char=='i':
                list1.append('j')

```

```
    else:  
        list1.append(char)  
  
    for i in range(len(array)):  
        char=array[i]  
  
        if char not in list1:  
            if char=='i':  
                list1.append('j')  
  
            else:  
                list1.append(char)  
  
    return list1
```

```
def matrix(list1):  
    m=[]  
    index=0  
  
    for i in range(5):  
        a=[]  
        for j in range(5):  
            a.append(list1[index])  
            index=index+1  
  
        m.append(a)  
  
    print("Matrix:")  
  
    for i in range(5):  
        for j in range(5):  
            print(m[i][j],end=" ")  
  
    print()  
  
    return m  
  
def plain(text):  
    text=text.replace(" ","")  
    text=text.lower()  
    p=list()  
  
    for i in range(len(text)):
```

```

char=text[i]
if char=='i':
    p.append('j')
else:
    p.append(char)
for i in range(0,len(p),2):
    if i<len(p)-1:
        if p[i]==p[i+1]:
            p.insert(i+1,"x")
    if len(p)%2!=0:
        p.append("x")
return p

```

```

def enc(p,m):
    encr=""
    for i in range(0,len(p),2):
        print(p[i],":",p[i+1])
        for j in range(5):
            for k in range(5):
                if p[i]==m[j][k]:
                    a=j
                    b=k
        for j in range(5):
            for k in range(5):
                if p[i+1]==m[j][k]:
                    c=j
                    d=k
        if a==c and b!=d:
            encr+=(m[a][(b+1)%5])
            encr+=(m[c][(d+1)%5])
    elif b==d and a!=c:

```

```

encr+=(m[(a+1)%5][b])
encr+=(m[(c+1)%5][b])
else:
    encr+=(m[a][d])
    encr+=(m[c][b])
return encr

```

```

def dec(p,m):
    decr"""
    for i in range(0,len(p),2):
        print(p[i],":",p[i+1])
        for j in range(5):
            for k in range(5):
                if p[i]==m[j][k]:
                    a=j
                    b=k
                for j in range(5):
                    for k in range(5):
                        if p[i+1]==m[j][k]:
                            c=j
                            d=k
                if a==c and b!=d:
                    decr+=(m[a][(b-1)%5])
                    decr+=(m[c][(d-1)%5])
                elif b==d and a!=c:
                    decr+=(m[(a-1)%5][b])
                    decr+=(m[(c-1)%5][b])
                else:
                    decr+=(m[a][d])
                    decr+=(m[c][b])
    return decr

```

```

key=input("Enter Key:")
text=input("Enter Text:")

#creating dataList
list1=datalist_normal(key)
print("Datalist:",list1)

#creating matrix
matrix1=matrix(list1)

#creating plaintext list and adding dummy letters
plaintext=plain(text)
print("Plaintext:",plaintext)

#creating pairs
#pair(plaintext)

```

```

#encryption
encrypt=enc(plaintext,matrix1)
print("Encrypted:",encrypt)

```

```

#decryption
decrypt=dec(encrypt,matrix1)
print("Decrypyted:",decrypt)

```

OUTPUT:

```

Enter Key:4
Enter Text:Hello
Datalist: ['!', 'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r', 't', 'v', 'w', 'x', 'y', 'z']
Matrix:
4 a b c d
e f g h j
k l m n o
p q r s t
u v w x y
Plaintext: ['h', 'e', 'l', 'l', 'o', ' ', 'w', 'o', 'r', 'l', 'd']
h : e
l : l
o : o
w : v
r : o
l : e
d : t
Encrypted: jfnvwmkxx
j : f
n : v
m : k
w : o
r : x
l : e
d : t
Decrypyted: helxloxx
>>> |

```

3. VIGENERE CIPHER

CODE:

```
import math

l="abcdefghijklmnopqrstuvwxyz"
pt=input("Enter Plain Text:")
pt=pt.replace(" ","")
pt=pt.lower()
lenk=int(input("Enter Length of Key:"))

k=[]

for i in range(0,lenk):
    v=int(input("Enter Key"+str(i+1)+":"))
    k.append(v)

en=""
de=""
n=0

for i in range(0,len(pt)):
    if(n<len(k)):
        n=n+1
    if(n>=len(k)):
        n=0
    ind=l.index(pt[i])+k[n-1]
    en=en+l[ind%26]

n=0

print("\nEncrypted Text:",en)

for i in range(0,len(pt)):
    if(n<len(k)):
        n=n+1
    if(n>=len(k)):
        n=0
    ind=l.index(en[i])-k[n-1]
```

```
de=de+l[ind%26]
print("\nDecrypted Text:",de)
```

OUTPUT:

```
Enter Plain Text:welcome
Enter Length of Key:2
Enter Key1:1
Enter Key2:3
Encrypted Text: xhmfffff
Decrypted Text: welcome
>>> |
```

Practical 3-Transposition Cipher Technique

AIM: Write Programs to Implement the following Transposition Cipher Techniques:

1. Simple Columnar Technique
2. Railfence Technique

1. SIMPLE COLUMNAR TECHNIQUE:

CODE:

```
def plainText(text,key):  
    text=text.lower()  
    text=text.replace(" ","")  
    for i in range(len(text)):  
        if len(text)%len(key)!=0:  
            text+="x"  
    return text  
  
def keyList(key):  
    list1=list()  
    for i in range(len(key)):  
        list1.append(key[i])  
    return list1  
  
#Encryption Starts here!!!!  
  
def matrix_encrypt(text,list1):  
    m=[]  
    index=0  
    for i in range(len(text)//len(list1)):  
        a=[]  
        for j in range(len(list1)):  
            if index<len(text):  
                a.append(text[index])  
            index=index+1  
        m.append(a)
```

```

m.append(a)

print("Matrix:")

for i in range(len(text)//len(list1)):

    for j in range(len(list1)):

        print(m[i][j],end=" ")

    print()

return m

def encrypt(m, list1,list2, text):

    en=""

    row=(len(text)//len(list1))

    for k in range(len(list1)):

        num=list1.index(min(list2))

        list2.remove(min(list2))

        for i in range(row):

            for j in range(len(list1)):

                #print(m[i])

                #print(num)

                en+=m[i][num]

            break

        print(" ")

    print("Cipher Text: ",en)

    return en

def encryptionAlgo(text, key):

    plain_text=plainText(text,key)

    key_list1=keyList(key)

    key_list2=keyList(key)

    print(plain_text)

    print(key_list1)

    m_plain=matrix_encrypt(plain_text,key_list1)

    cipher=encrypt(m_plain,key_list1, key_list2, plain_text)

    return cipher

```

```

#Decryption Starts here!!!!
def keyList(key):
    list1=list()
    for i in range(len(key)):
        list1.append(key[i])
    return list1

def matrix_list(cipher,list1):
    a=[]
    matrix_list=list()
    var=len(cipher)//len(list1)
    index=0
    for i in range(len(list1)):
        if index<len(list1):
            letter=list1[i]
            letter=int(letter)
            num=(letter*var)-var
            for j in range(num,num+var):
                a.append(cipher[j])
        else:
            break
    print("List of Matrix Characters: ",a)
    return a

def matrix_decrypt(mat_list,list1):
    m=[]
    index=0
    for i in range(len(list1)):
        a=[]
        for j in range(len(mat_list)//len(list1)):
            if index<len(mat_list):
                a.append(mat_list[index])
            index=index+1
        m.append(a)
    return m

```

```
m.append(a)

print("Columnwise Groups of Matrix Characters: ",m)

print("Matrix:")

for i in range(len(mat_list)//len(list1)):

    for j in range(len(list1)):

        print(m[j][i],end=" ")

    print()

return m

def decryption(m,mat_list,list1):

    de=""

    for i in range(len(mat_list)//len(list1)):

        for j in range(len(list1)):

            de+=m[j][i]

        print(" ")

    print("Plain text: ",de)

    return de

def decryptionAlgo(cipher, key):

    list1=keyList(key)

    mat_list=matrix_list(cipher,list1)

    m=matrix_decrypt(mat_list,list1)

    plain=decryption(m,mat_list,list1)

    return plain

text=input("Enter plain text:")

key=input("Enter key:")

print("Encryption Goes Here!!!")

print(" ")

cipher=encryptionAlgo(text, key)

print(" ")

print("Decryption Goes Here!!!")

print(" ")

plain=decryptionAlgo(cipher, key)
```

OUTPUT:

```
Enter plain text:hello world
Enter key:1432
Encryption Goes Here!!!
helloworldxx
['1', '4', '3', '2']
Matrix:
h e l l
o w o r
l d x x
Cipher Text: holirxioxewd
Decryption Goes Here!!!
List of Matrix Characters: ['h', 'o', 'l', 'e', 'w', 'd', 'l', 'o', 'x', 'r', 'x']
Columnwise Groups of Matrix Characters: [['h', 'o', 'l'], ['e', 'w', 'd'], ['l', 'o', 'x'], ['r', 'x']]
Matrix:
h e l l
o w o r
l d x x
Plain text: helloworldxx
>>> |
```

2. RAILFENCE TECHNIQUE:

CODE:

```
def rf(pt):
```

```
    pt=pt.replace(" ","")
```

```
    u=""
```

```
    l=""
```

```
    en=""
```

```
    de=""
```

```
    j=len(pt)//2
```

```
    for i in range(0,len(pt)):
```

```
        if(i%2==0):
```

```
            u+=pt[i]
```

```
        else:
```

```
            l+=pt[i]
```

```
    en=u+l
```

```
    print("Encrypted Text is: ",en)
```

```
    if(len(pt)%2==0):
```

```
        for i in range(0,j):
```

```
            de+=u[i]
```

```
            de+=l[i]
```

```
    print("Decrypted Text is: ",de)
```

```
else:  
    for i in range(0,j):  
        de+=u[i]  
        de+=l[i]  
        de+=u[-l]  
    print("Decryption text is:",de)  
pt=input("Enter Some Plain Text:")  
rf(pt)
```

OUTPUT:

```
-----  
Enter Some Plain Text:helloworld  
Encrypted Text is: hloolelwrd  
Decrypted Text is: helloworld  
>>> |
```

Practical 4

AIM: Write program to encrypt and decrypt strings using

1. DES Algorithm
2. AES Algorithm

1. DES Algorithm:

CODE:

```
from pyDes import*
data=input("Enter Data:")
k=des("DESCRYPT",CBC,"0\0\0\0\0\0\0\0",pad=None,padmode=PAD_PKCS5)
d=k.encrypt(data)
print("Encrypted:%r"%d)
print("Decrypted:%r"%(k.decrypt(d)))
```

OUTPUT:



The screenshot shows the Python 3.7.0 Shell window. The title bar reads "Python 3.7.0 Shell". The menu bar includes File, Edit, Shell, Debug, Options, Window, and Help. The main window displays the following text:

```
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:59:51) [MSC v.1914 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
=====
RESTART: E:\des.py =====
Enter Data:HelloWorld
Encrypted:b'\xc5\xfa%\x\uff\x9b\x\x4\xb0\xab\x8am\x94g\x9c\t'
Decrypted:b'HelloWorld'
>>> |
```

2. AES Algorithm:

CODE:

```
import pyaes

aes=pyaes.AESModeOfOperationCTR(b'DCRYPTDESCRYPT')

plaintext=input("Enter text:")

ct=aes.encrypt(plaintext)

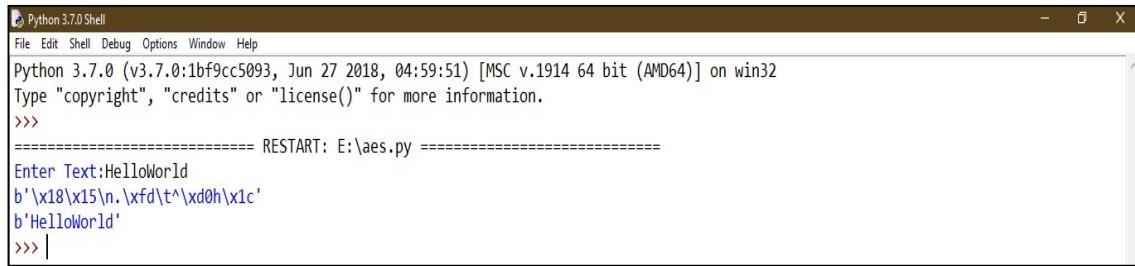
print(ct)

aes=pyaes.AESModeOfOperationCTR(b'DCRYPTDESCRYPT')

plaintext=aes.decrypt(ct)

print(plaintext)
```

OUTPUT:



The screenshot shows the Python 3.7.0 Shell window. The title bar reads "Python 3.7.0 Shell". The menu bar includes File, Edit, Shell, Debug, Options, Window, and Help. The main window displays the following text:

```
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:59:51) [MSC v.1914 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.

>>>
=====
RESTART: E:\aes.py =====
Enter Text:HelloWorld
b'\x18\x15\n.\xfd\t^\\xd0h\x1c'
b'HelloWorld'
>>> |
```

Practical 5

AIM: Write a Program to Demonstrate RSA Algorithm.

CODE:

```
def gcd(m,n):
    if m<n:
        (m,n)=(n,m)
    if(m%n)==0:
        return n
    else:
        return(gcd(n,m%n))#Recursion taking place

def rsaAlgo(p,q):
    print("p=",p,"q=",q)
    n=p*q
    fin=(p-1)*(q-1)
    for i in range(1,fin):
        if gcd(i,fin)==1:
            e=i
            d=i
    print("d=",d)
    #Encryption
    print("Enter Message such that Message<",n)
    message=int(input(""))
    enc=message**e%n
    print("enc=",enc)

    #Decryption
    print("Enter c such that c<",n)
    cipher=int(input(""))
    dec=cipher**d%n
    print("dec=",dec)
```

```

def primeNum(n):
    if n>1:
        for i in range(2,n):
            if n%i==0:
                print("Invalid")
            n=int(input("Enter a Prime Number"))

    else:
        print("Valid")

    return n

p=int(input("Enter A Prime Number:"))

p1=primeNum(p)

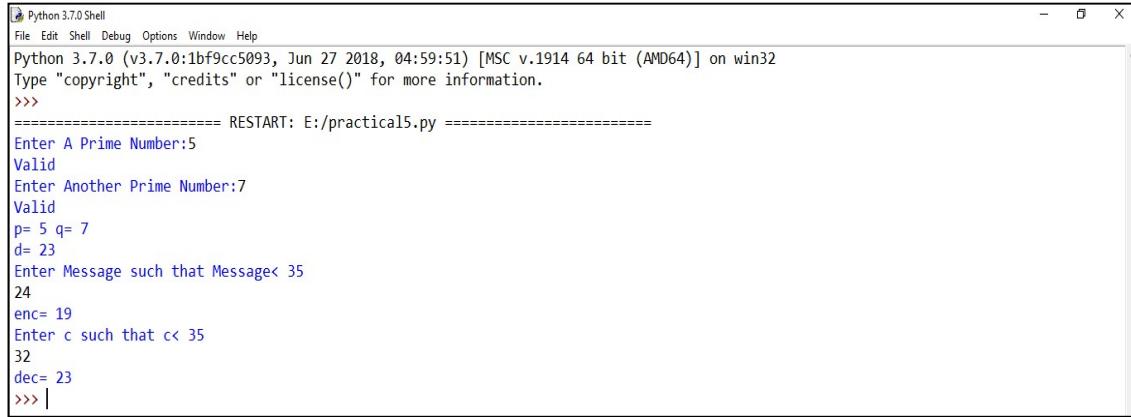
q=int(input("Enter Another Prime Number:"))

q1=primeNum(q)

rsaAlgo(p1,q1)

```

OUTPUT:



The screenshot shows the Python 3.7.0 Shell window. The title bar reads "Python 3.7.0 Shell". The menu bar includes File, Edit, Shell, Debug, Options, Window, and Help. The main window displays the following session:

```

Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:59:51) [MSC v.1914 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> ===== RESTART: E:/practical5.py =====
Enter A Prime Number:5
Valid
Enter Another Prime Number:7
Valid
p= 5 q= 7
d= 23
Enter Message such that Message< 35
24
enc= 19
Enter c such that c< 35
32
dec= 23
>>> |

```

Practical 6

AIM: Write a Program to Perform Diffie-Hellman Key Exchange.

CODE:

```
#Diffie Hellman Key Exchange
from math import sqrt

# Returns True if n is prime
def isPrime( n):

    # Corner cases
    if (n <= 1):
        return False
    if (n<= 3):
        return True
    if (n % 2 == 0 or n % 3 == 0):
        return False
    i = 5
    while(i * i <= n):
        if (n % i== 0 or n % (i + 2) == 0) :
            return False
        i= i + 6
    return True

def power(x,y,p):
    res = 1 # Initialize result
    x = x % p # Update x if it is more
    # than or equal to p
    while (y > 0):
        # If y is odd, multiply x with result
        if (y & 1):
            res = (res * x) % p
```

```

y = y>>1

x = (x * x) % p

return res

# Utility function to store prime # factors of a number

def findPrimefactors(s, n):

    while(n%2==0):

        # Print the number of 2s that divide n while (n % 2 == 0) :

        s.add(2)

        n = n // 2

        # n must be odd at this po. So we can # skip one element (Note i = i +2)

    for i in range(3, int(sqrt(n)), 2):

        # While i divides n, print i and divide n

        while (n % i == 0) :

            s.add(i)

            n = n // i

        if (n > 2) :

            s.add(n)

# Function to find smallest primitive # root of n

def findPrimitive( n) :

    s= set()

    # Check if n is prime or not

    if (isPrime(n) == False):

        return -1

    # Find value of Euler Totient function

    # of n. Since n is a prime number, the # value of Euler Totient function is n-1 # as there are
    n-1 relatively prime numbers.

    phi = n - 1

    # Find prime factors of phi and store in a set

    findPrimefactors(s, phi)

```

```

# Check for every number from 2 to phi
for r in range(2, phi + 1):
    # Iterate through all prime factors of phi. # and check if we found a power with value 1
    flag = False
    for it in s:
        if (pow(r, phi // it,n) == 1):
            flag = True
            break
    # If there was no power with value 1.
    if (flag == False):
        return r
    # If no primitive root found
return r

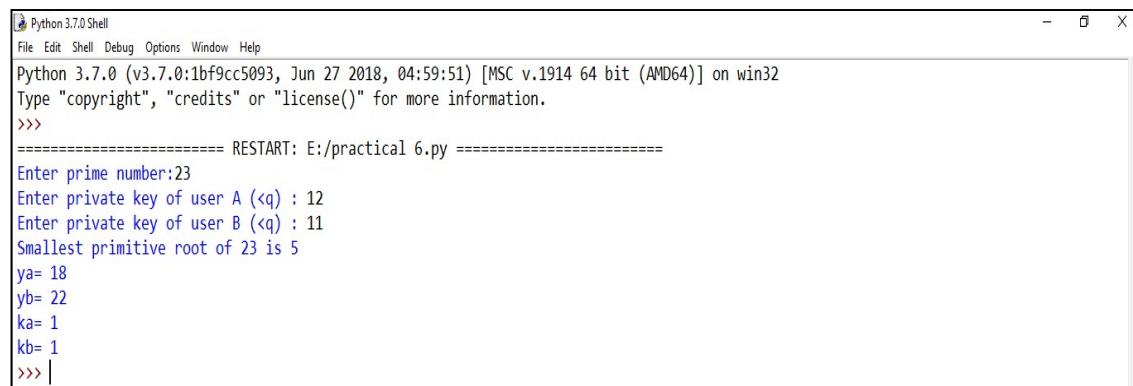
#generating public key of user A,B
def pua(xa,a,q):
    if xa<q:
        ya=(a**xa)%q
        return ya
    else:
        print("xa should be < ", q)
        #key generation
def keyGen(x,y,q):
    k=(y**x)%q
    return k

q = int(input("Enter prime number:"))
a=findPrimitive(q)
xa=int(input("Enter private key of user A (<q) : "))
xb=int(input("Enter private key of user B (<q) : "))
ya=pua(xa,a,q)
yb=pua(xb,a,q)

```

```
ka=keyGen(xa,yb,q)
kb=keyGen(xb,ya,q)
print("Smallest primitive root of", q, "is", a)
print("ya=",ya)
print("yb=",yb)
print("ka=",ka)
print("kb=",kb)
```

OUTPUT:



The screenshot shows a Python 3.7.0 Shell window. The menu bar includes File, Edit, Shell, Debug, Options, Window, and Help. The title bar says "Python 3.7.0 Shell". The main area displays the following text:

```
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:59:51) [MSC v.1914 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
=====
RESTART: E:/practical 6.py =====
Enter prime number:23
Enter private key of user A (<q>) : 12
Enter private key of user B (<q>) : 11
Smallest primitive root of 23 is 5
ya= 18
yb= 22
ka= 1
kb= 1
>>> |
```

Practical 7

AIM: Write a Program to Perform MD5.

CODE:

```
#MD5
import hashlib

def file_check(filename):
    hash1=hashlib.md5()
    with open(filename,'rb')as open_file:
        content=open_file.read()
        hash1.update(content)
    print(hash1.hexdigest())

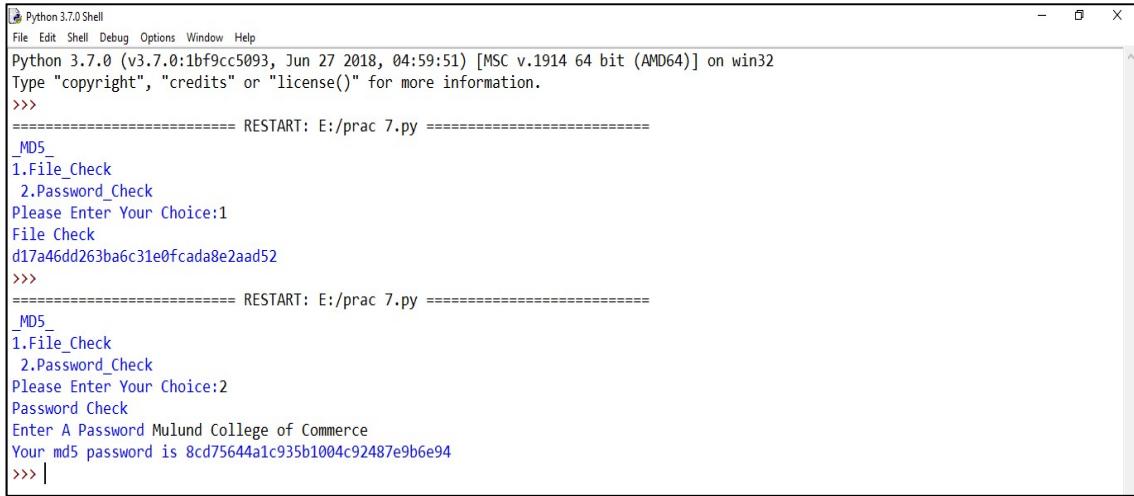
def pass_check(pw):
    hash1=hashlib.md5(pw.encode('utf-8'))
    print("Your md5 password is",hash1.hexdigest())
    print("_MD5_")
    print("1.File_Check \n 2.Password_Check")
    choice=int(input("Please Enter Your Choice:"))

    if(choice==1):
        print("File Check")
        fn='hello.txt'
        file_check(fn)

    elif(choice==2):
        print("Password Check")
        pw=input("Enter A Password")
        pass_check(pw)

    else:
        print("Wrong Choice")
```

OUTPUT:



The screenshot shows a window titled "Python 3.7.0 Shell". The menu bar includes File, Edit, Shell, Debug, Options, Window, and Help. The main area displays the following text:

```
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:59:51) [MSC v.1914 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.

>>> ===== RESTART: E:/prac 7.py =====
MD5_
1.File_Check
2.Password_Check
Please Enter Your Choice:1
File Check
d17a46dd263ba6c31e0fcada8e2aad52
>>> ===== RESTART: E:/prac 7.py =====
MD5_
1.File_Check
2.Password_Check
Please Enter Your Choice:2
Password Check
Enter A Password Mulund College of Commerce
Your md5 password is 8cd75644a1c935b1004c92487e9b6e94
>>> |
```

Practical 8

AIM: Write a Program to Calculate HMAC SHA-1 & SHA-512 Signature.

1) SHA-1

CODE:

```
import hashlib

def sha(m):
    m=m.encode("utf8")
    hash1=hashlib.sha1(m)
    print("SHA-1 signature of Your Message is:",hash1.hexdigest())
pt=input("Enter a Message:")
sha(pt)
```

2) SHA-512

CODE:

```
import hashlib

def sha(m):
    m=m.encode("utf8")
    hash512=hashlib.sha1(m)
    print("SHA-512 signature of Your Message is:",hash512.hexdigest())
pt=input("Enter a Message:")
sha(pt)
```

OUTPUT:



The screenshot shows a Python 3.7.0 Shell window. The title bar reads "Python 3.7.0 Shell". The menu bar includes File, Edit, Shell, Debug, Options, Window, and Help. The main window displays the following text:

```
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:59:51) [MSC v.1914 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.

>>>
=====
RESTART: E:\prac 8.py =====
Enter a Message:Mulund College of Commerce
SHA-1 signature of Your Message is: 6d1d2e40af03ca1a389ac70ca8a3204bc7cb1b44
>>>
=====
RESTART: E:\prac 8.2.py =====
Enter a Message:Mulund College of Commerce
SHA-512 signature of Your Message is: 6d1d2e40af03ca1a389ac70ca8a3204bc7cb1b44
>>> |
```

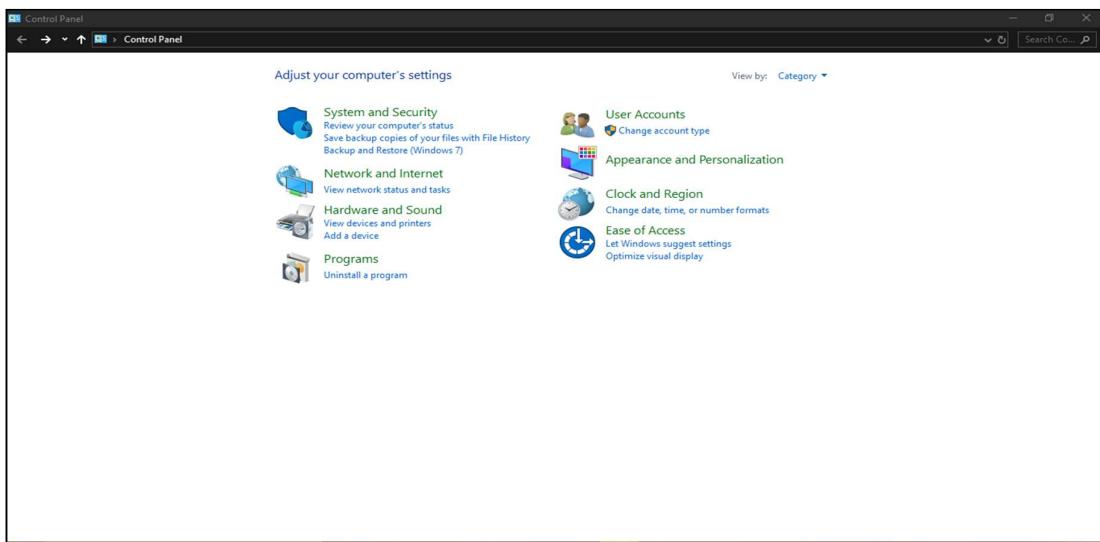
Practical 9

AIM: Configure Windows Firewall to block:

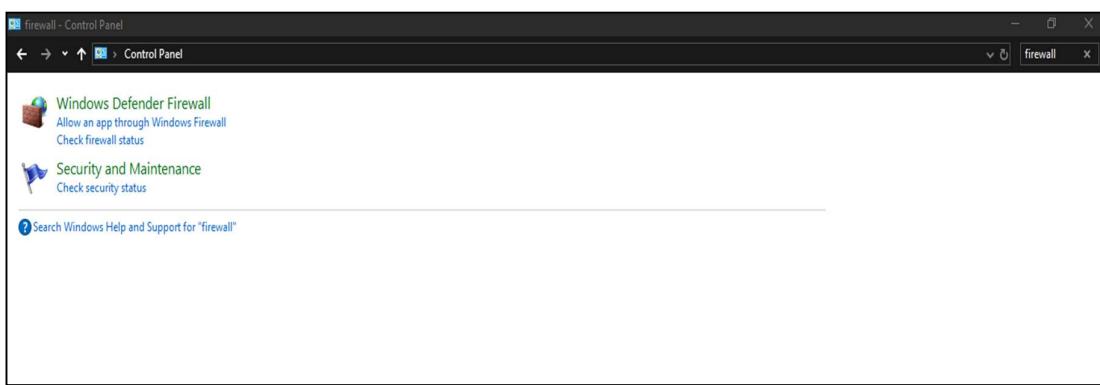
- 1) A Port
- 2) A Program
- 3) A Website

1) A Port

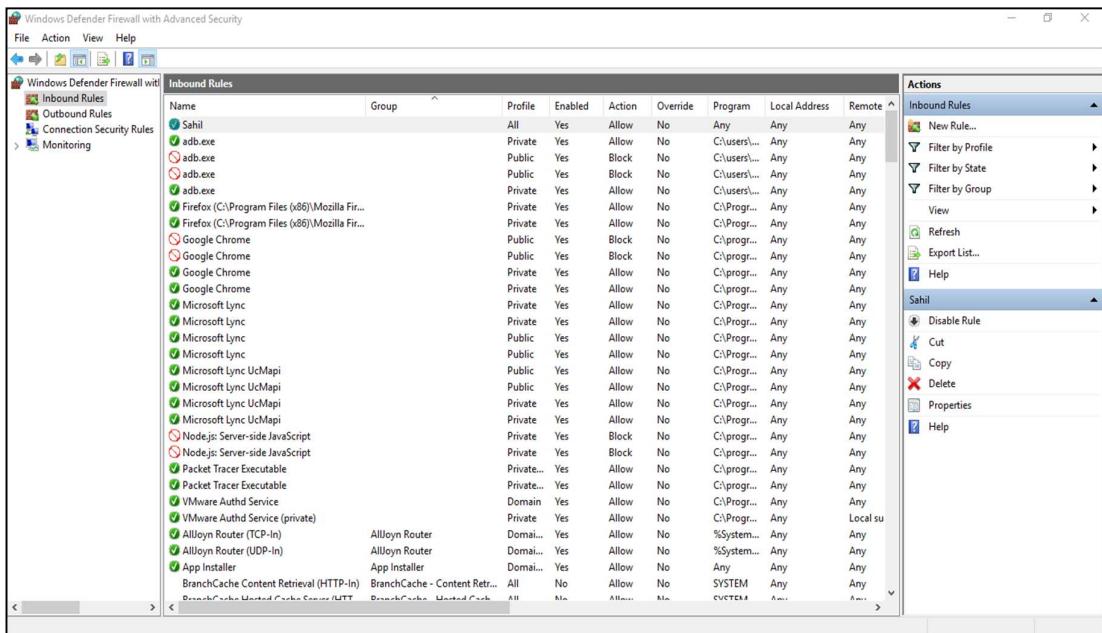
Step 1: Open Control Panel.



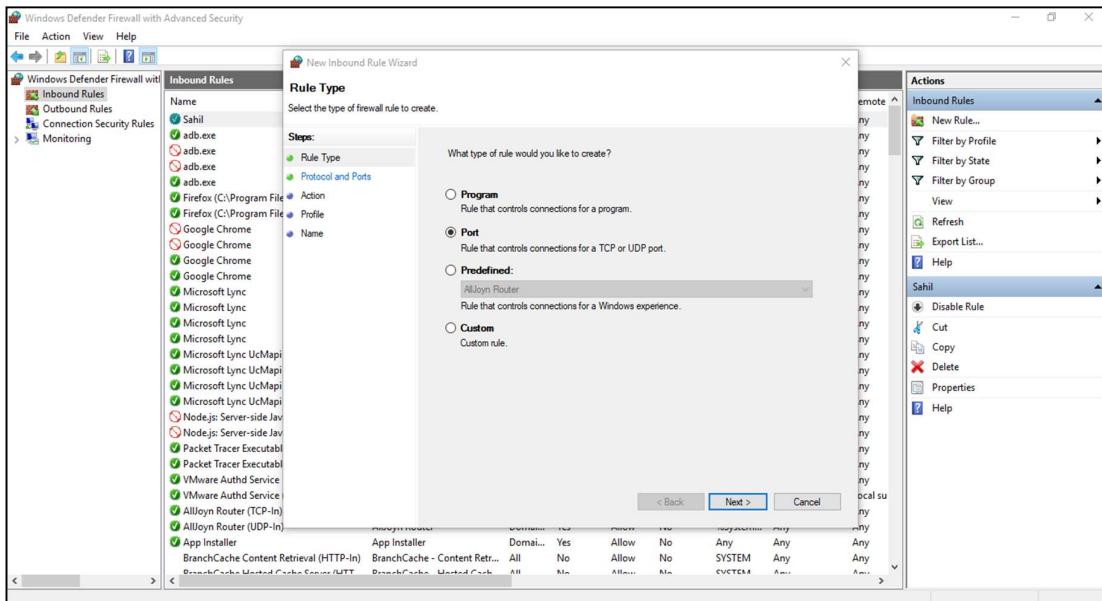
Step 2: Go To Firewall in Search Box.



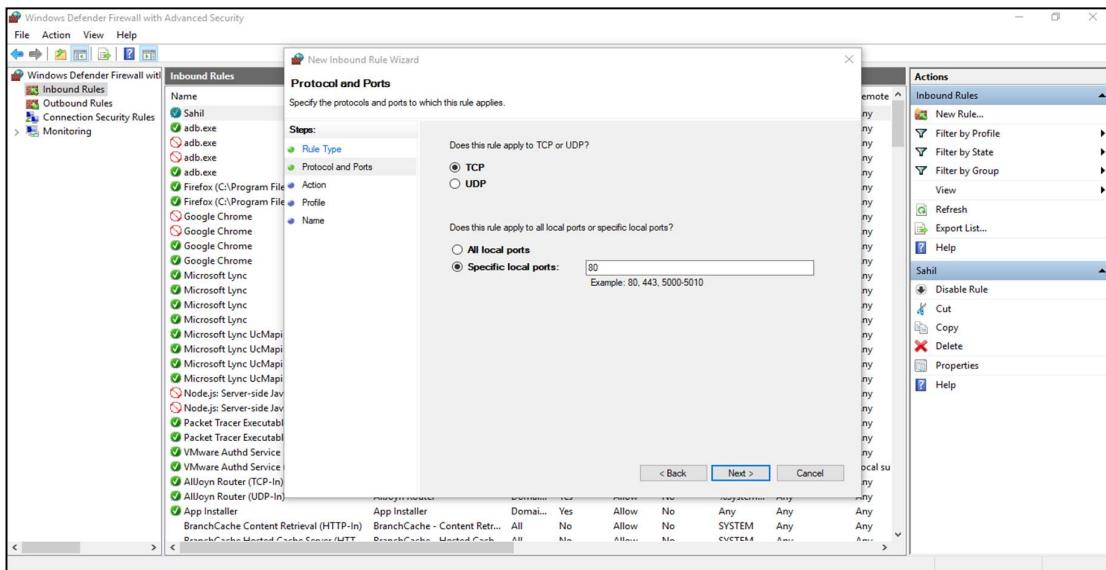
Step 3: In the Windows Firewall, Click on advanced setting and then click on Inbound Rule.



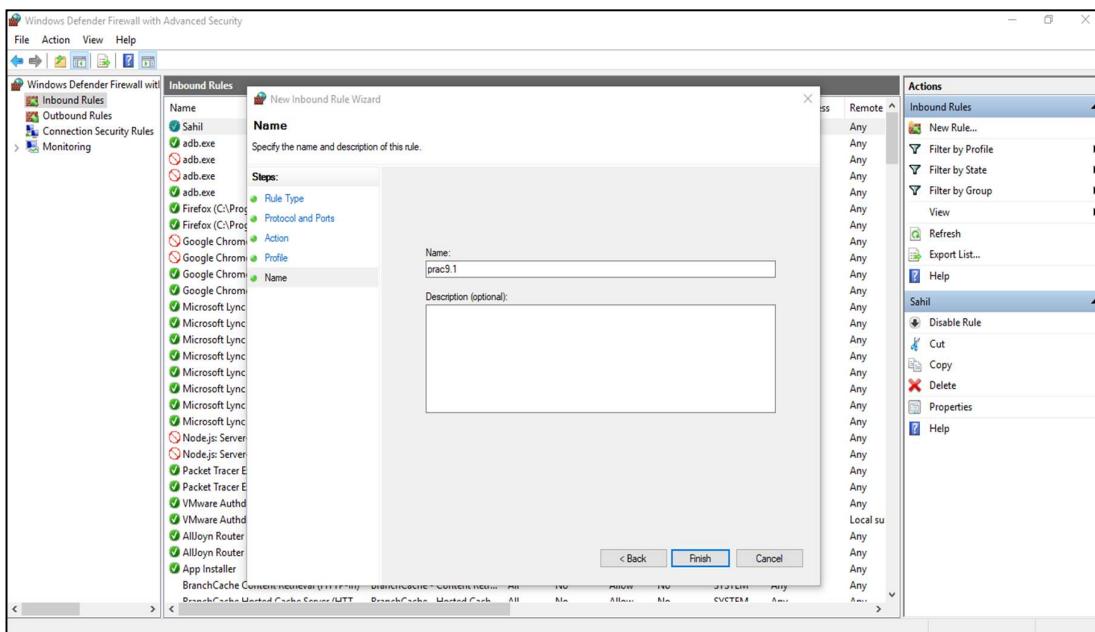
Step 4: Go to New Rule in Right Hand Side Of The Window.

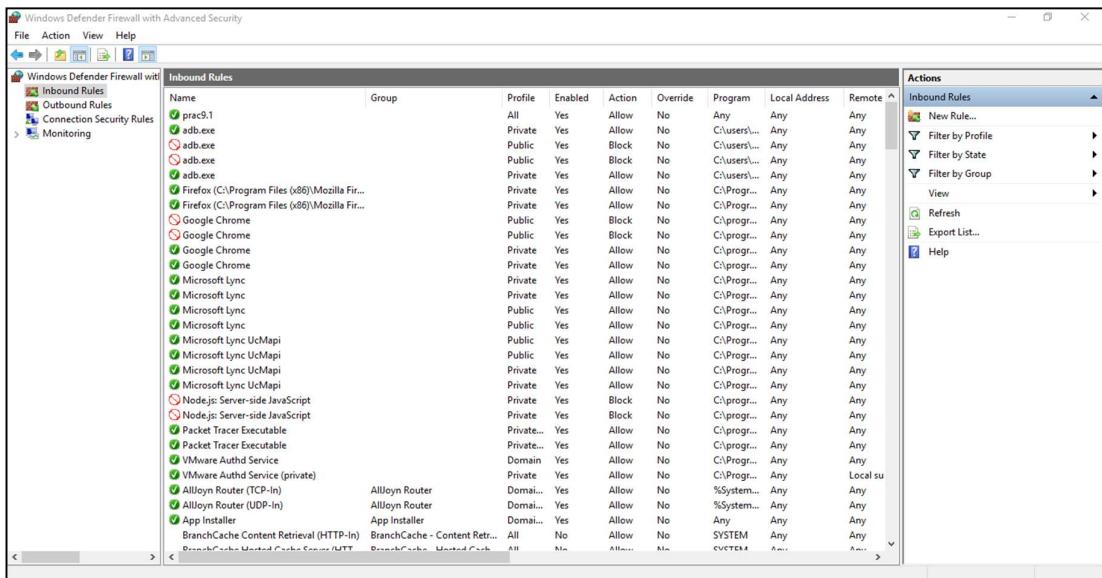


Step 5: Select Port and then select TCP and select Specific Local Ports and enter the Port Number, which you want to block(Eg:80)



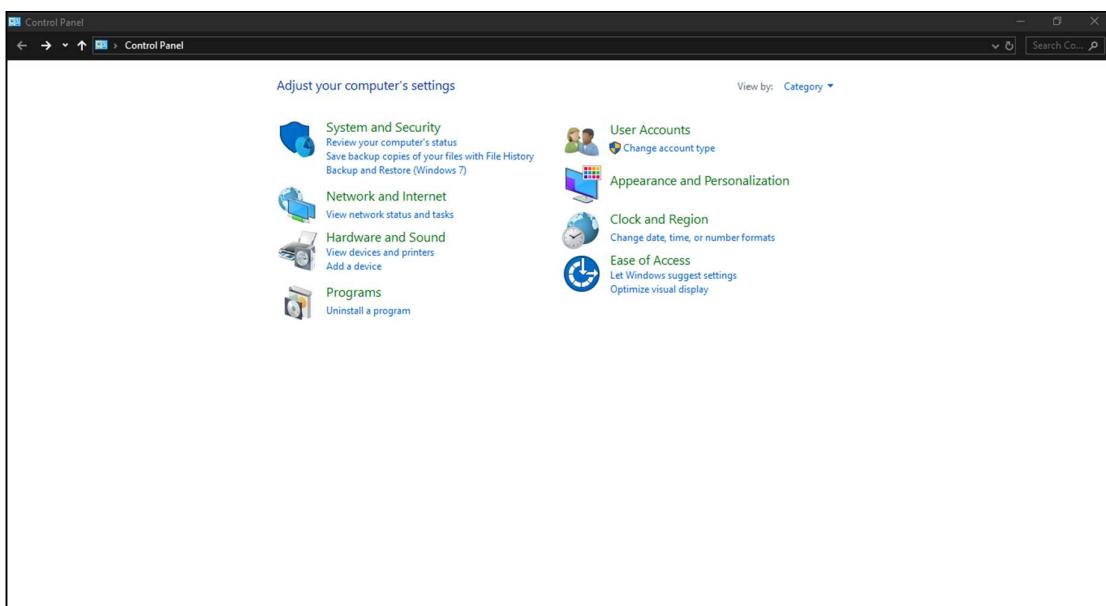
Step 6: Click Next till the last Dialog Box appears, and Name the New Inbound Rule as you wish and Click on Finish:



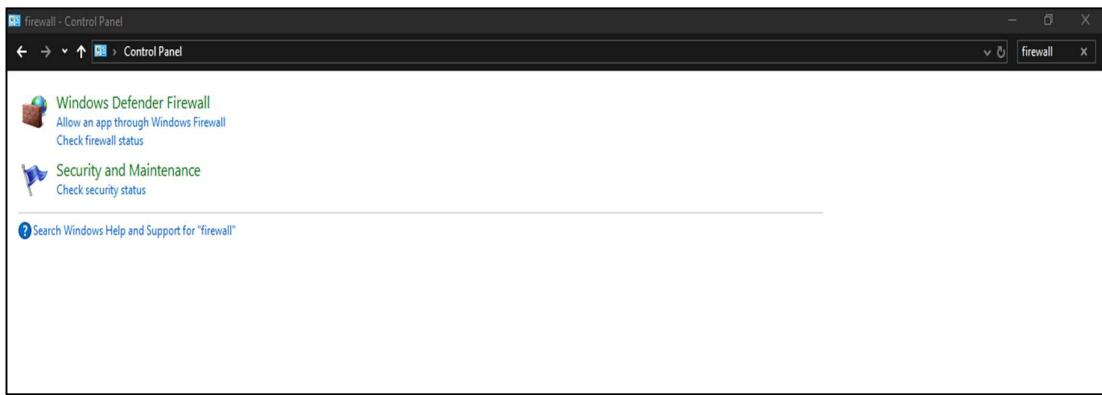


2) A Program:

Step 1: Open Control Panel.



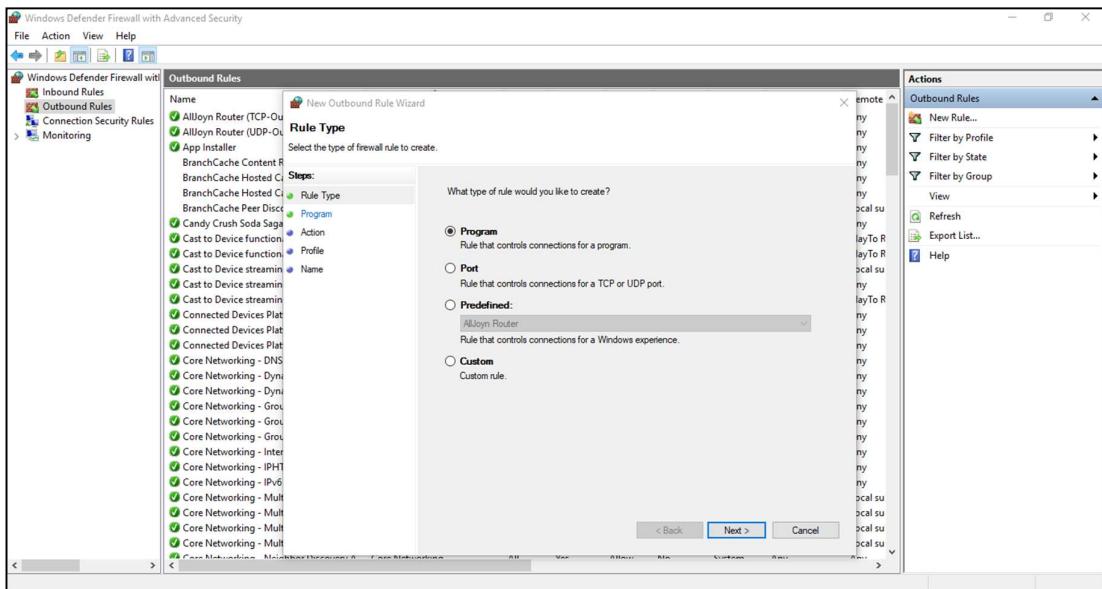
Step 2: Go To Firewall in Search Box.



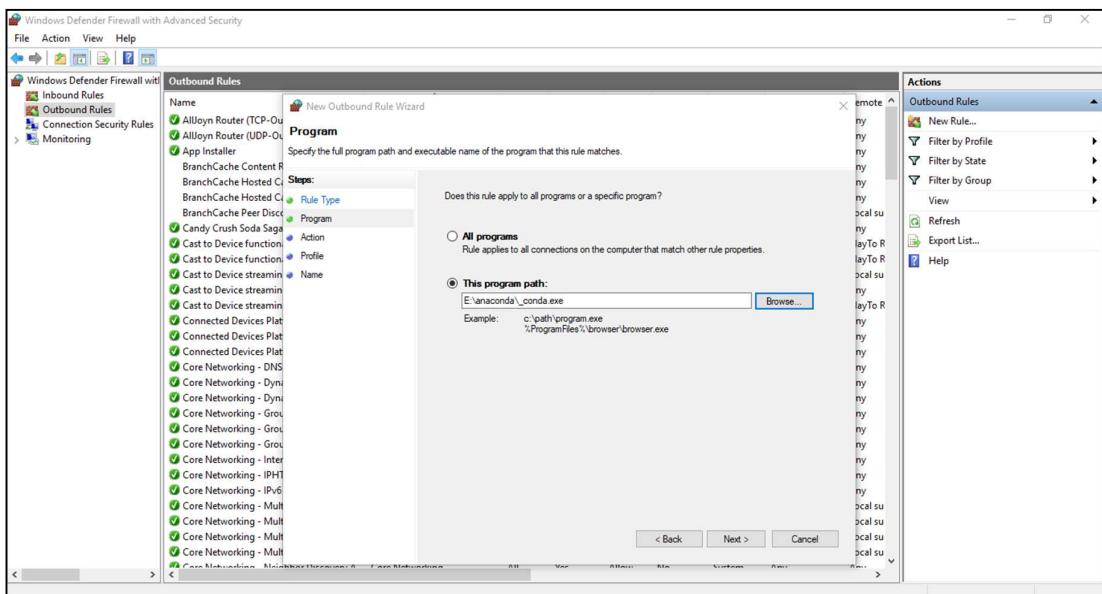
Step 3: In the Windows Firewall, Click on advanced setting and then click on Outbound Rule.

A screenshot of the "Windows Defender Firewall with Advanced Security" application. The left pane shows a navigation tree with "Outbound Rules" selected. The right pane displays a table titled "Outbound Rules" with columns: Name, Group, Profile, Enabled, Action, Override, Program, Local Address, and Remote. The table lists numerous rules, such as "AllJoyn Router (TCP-Out)", "AllJoyn Router (UDP-Out)", and various "Core Networking" entries. On the far right, a sidebar titled "Actions" contains options like "New Rule...", "Filter by Profile", "Filter by State", "Filter by Group", "View", "Refresh", "Export List...", and "Help".

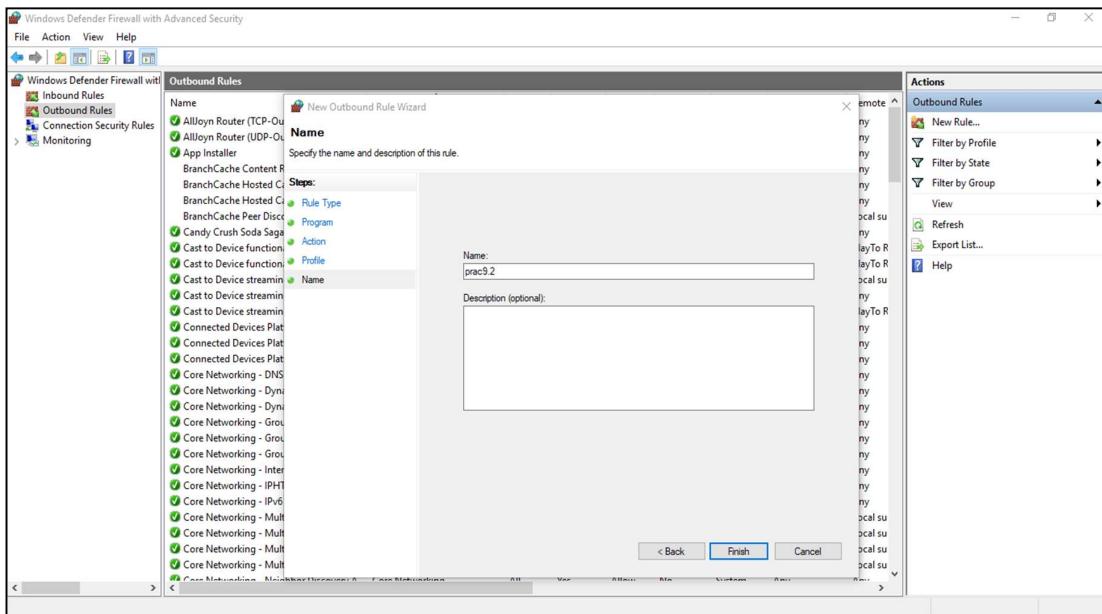
Step 4: Go to New Rule in Right Hand Side Of The Window.



Step 5: Select Program then, Select This Program Path then, browse the files which you want to block.



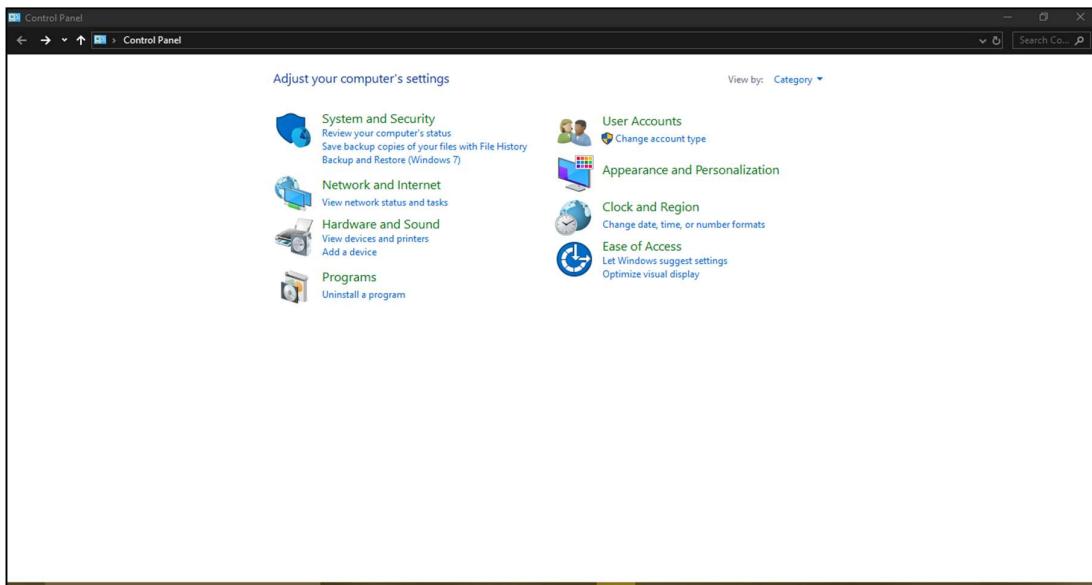
Step 6: Click Next till the last Dialog Box appears, and Name the New Outbound Rule as you wish and Click on Finish:



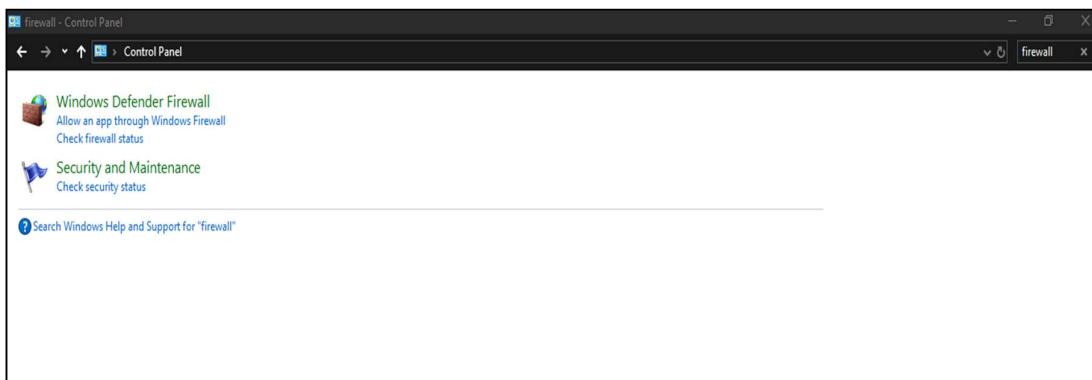
Name	Group	Profile	Enabled	Action	Override	Program	Local Address	Remote
prac9.2		All	Yes	Block	No	%elanaco...	Any	Any
Alloyyn Router (TCP-Out)	Alloyyn Router	Domai...	Yes	Allow	No	%System...	Any	Any
Alloyyn Router (UDP-Out)	Alloyyn Router	Domai...	Yes	Allow	No	%System...	Any	Any
App Installer	App Installer	All	Yes	Allow	No	Any	Any	Any
BranchCache Content Retrieval (HTTP-...	BranchCache - Content Retr...	All	No	Allow	No	SYSTEM	Any	Any
BranchCache Hosted Cache Client (HTT...	BranchCache - Hosted Cach...	All	No	Allow	No	SYSTEM	Any	Any
BranchCache Hosted Cache Server(HTT...	BranchCache - Hosted Cach...	All	No	Allow	No	SYSTEM	Any	Any
BranchCache Peer Discovery (WSD-Out)	BranchCache - Peer Discove...	All	No	Allow	No	%System...	Any	Local su...
Candy Crush Soda Saga	Candy Crush Soda Saga	All	Yes	Allow	No	Any	Any	Any
Cast to Device functionality (qWave-TCP...	Cast to Device functionality	Private...	Yes	Allow	No	%System...	Any	PlayTo R...
Cast to Device functionality (qWave-UDP...	Cast to Device functionality	Private...	Yes	Allow	No	%System...	Any	PlayTo R...
Cast to Device streaming server (RTP-Stre...	Cast to Device functionality	Private	Yes	Allow	No	%System...	Any	Local su...
Cast to Device streaming server (RTP-Stre...	Cast to Device functionality	Domain	Yes	Allow	No	%System...	Any	Any
Connected Devices Platform (Wi-Fi Dire...	Connected Devices Platform	Public	Yes	Allow	No	%System...	Any	PlayTo R...
Connected Devices Platform (TCP-Out)	Connected Devices Platform	Domai...	Yes	Allow	No	%System...	Any	Any
Connected Devices Platform (UDP-Out)	Connected Devices Platform	Domai...	Yes	Allow	No	%System...	Any	Any
Core Networking - DNS (UDP-Out)	Core Networking	All	Yes	Allow	No	%System...	Any	Any
Core Networking - Dynamic Host Config...	Core Networking	All	Yes	Allow	No	%System...	Any	Any
Core Networking - Dynamic Host Config...	Core Networking	All	Yes	Allow	No	%System...	Any	Any
Core Networking - Group Policy (LSASS-...	Core Networking	Domain	Yes	Allow	No	%System...	Any	Any
Core Networking - Group Policy (NP-Out)	Core Networking	Domain	Yes	Allow	No	System	Any	Any
Core Networking - Group Policy (TCP-Out)	Core Networking	Domain	Yes	Allow	No	%System...	Any	Any
Core Networking - Internet Group Mana...	Core Networking	All	Yes	Allow	No	System	Any	Any
Core Networking - IPHHTPS (TCP-Out)	Core Networking	All	Yes	Allow	No	%System...	Any	Any
Core Networking - IPv6 (IPv6-Out)	Core Networking	All	Yes	Allow	No	System	Any	Any
Core Networking - Multicast Listener Do...	Core Networking	All	Yes	Allow	No	System	Any	Local su...
Core Networking - Multicast Listener Rep...	Core Networking	All	Yes	Allow	No	System	Any	Local su...
Core Networking - Multicast Listener Rep...	Core Networking	All	Yes	Allow	No	System	Any	Local su...

3) A Website:

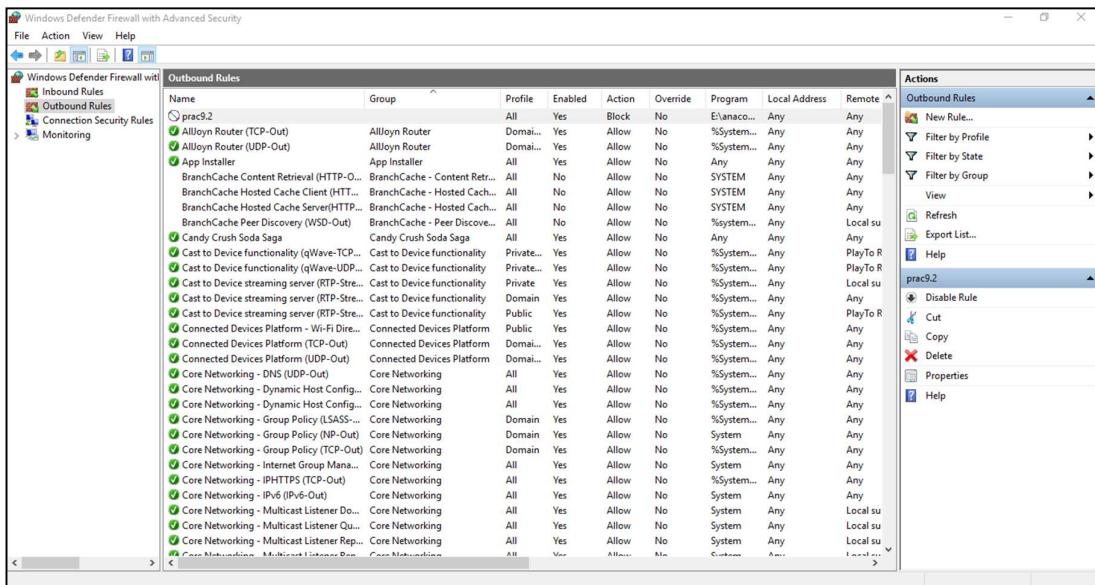
Step 1: Open Control Panel.



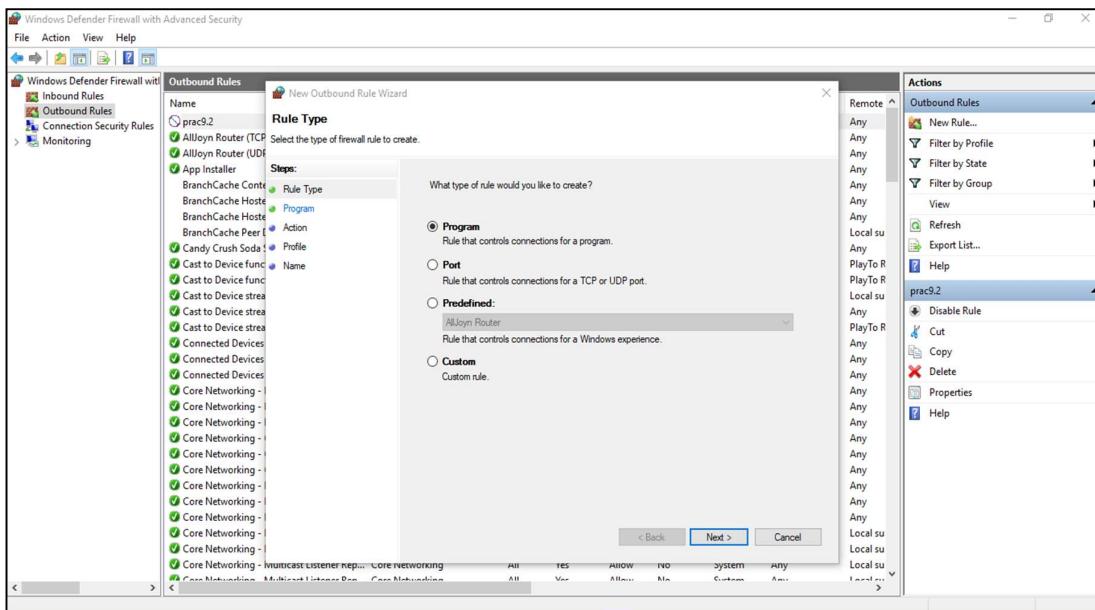
Step 2: Go To Firewall in Search Box.



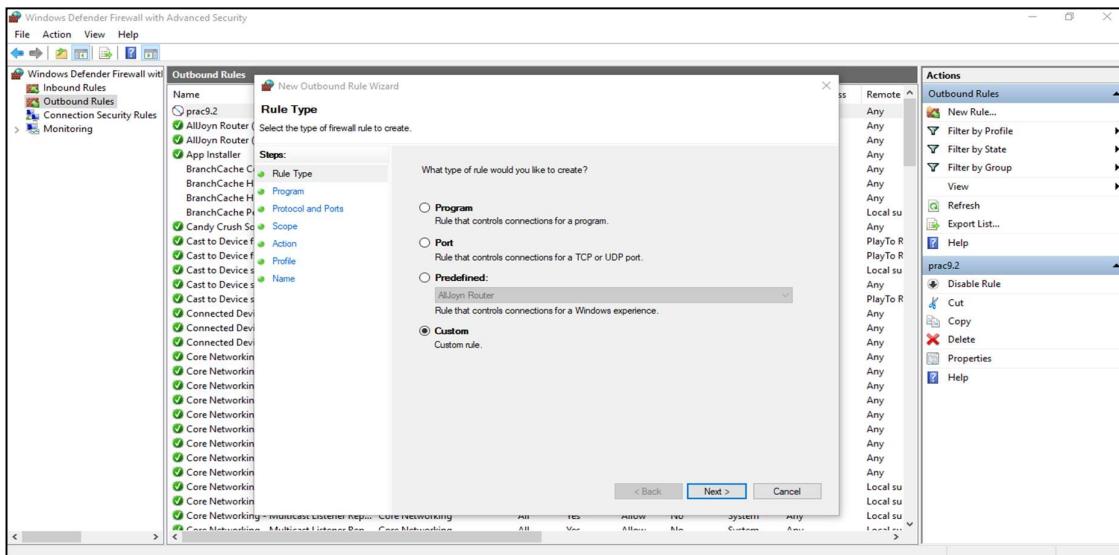
Step 3: In the Windows Firewall, Click on advanced setting and then click on Outbound Rule.



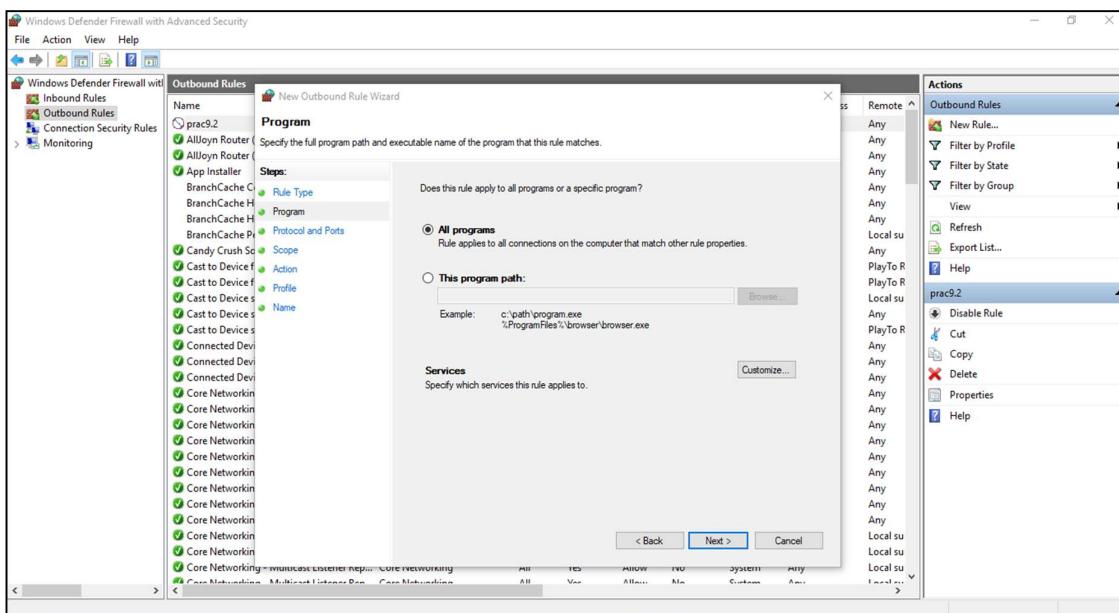
Step 4: Go to New Rule in Right Hand Side Of The Window.



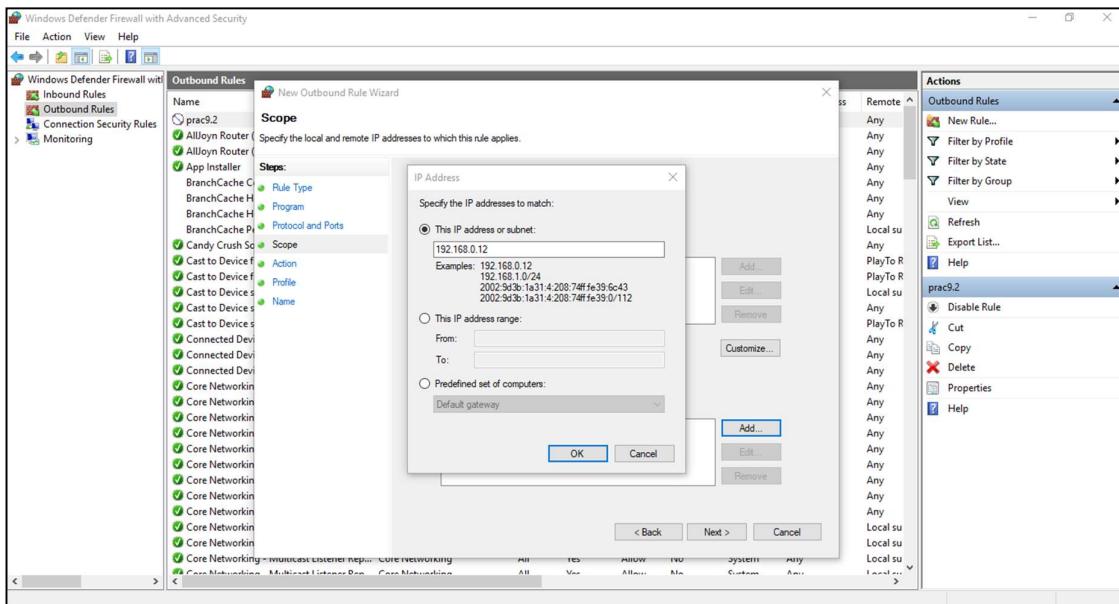
Step 5: Select Custom.



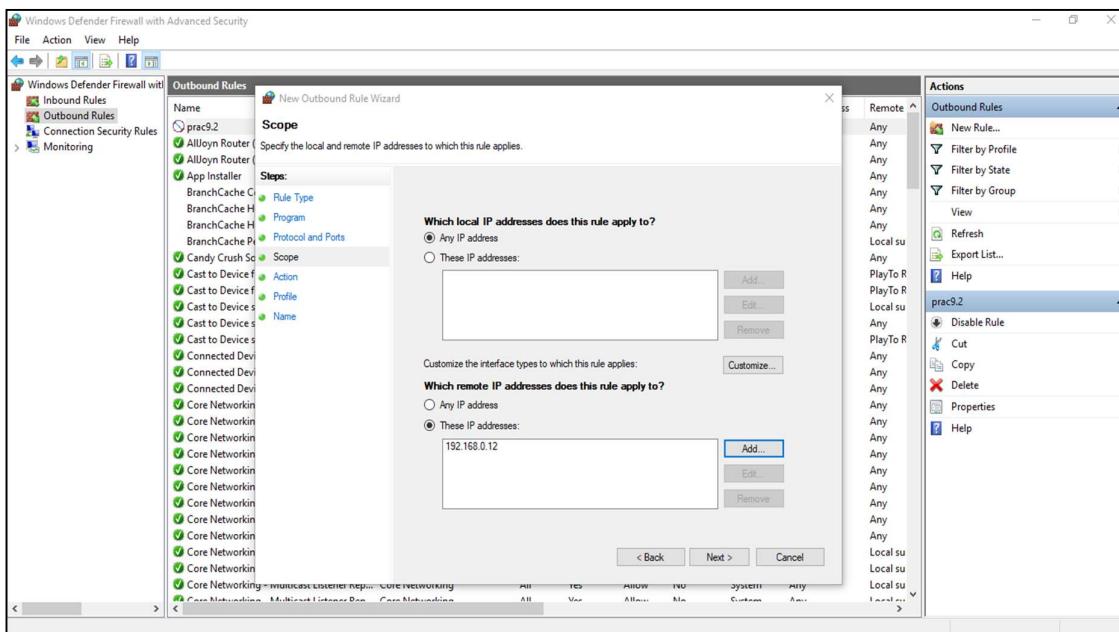
Step 6: Select All programs.

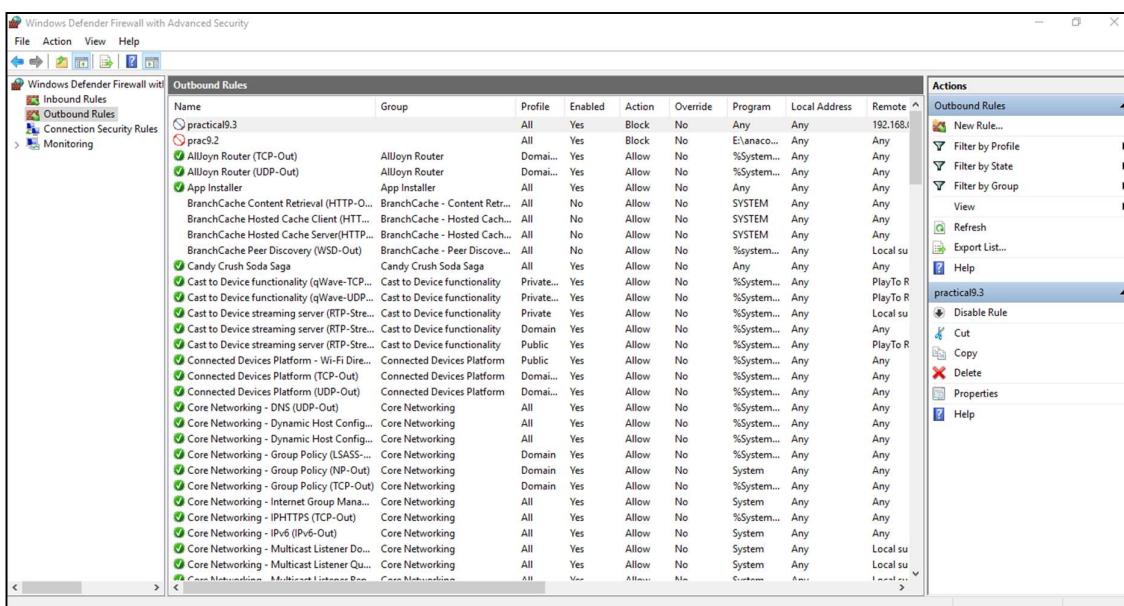
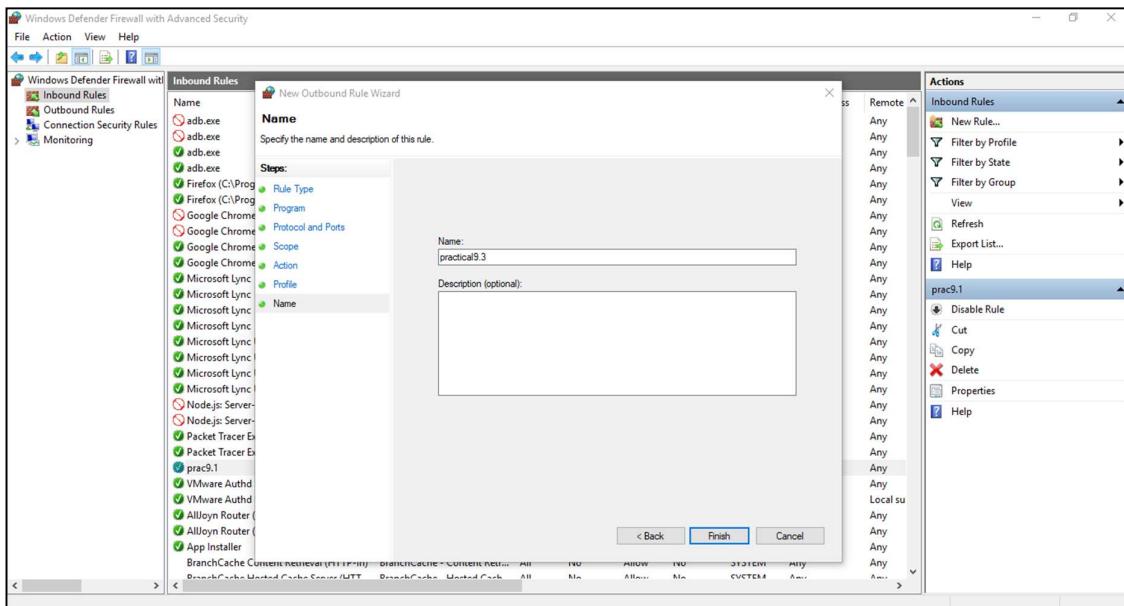


Step 7: Select Any, then select “These IP Address” then Click on Add and enter IP Address and click on OK.



Step 8: Click on Next Until Naming Box Appears and Name it as you wish, and click on Finish.





Practical 10

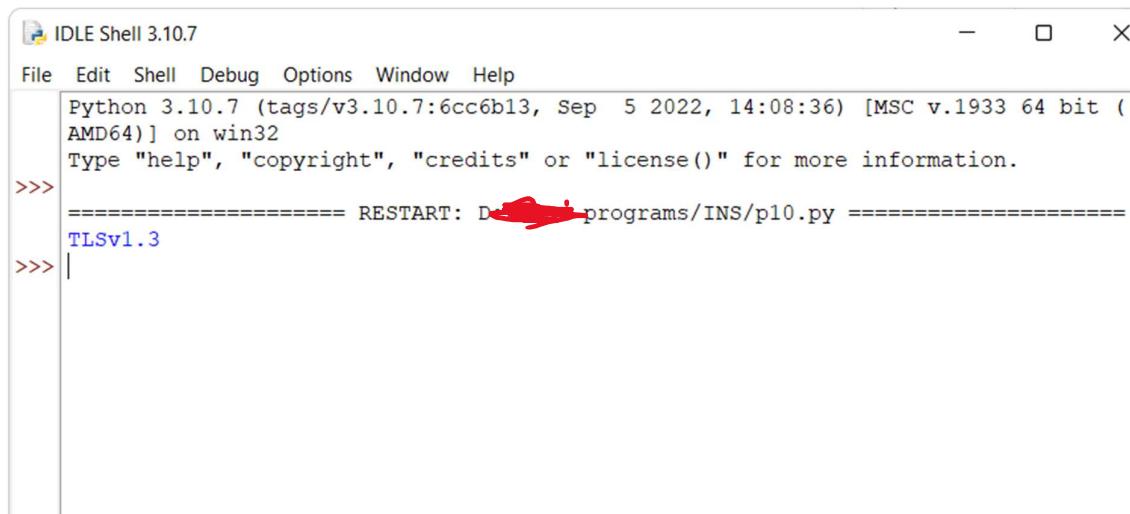
AIM: Write a Program to Implement SSL.

CODE:

```
import socket
import ssl

hostname='www.python.org'
context=ssl.create_default_context()
with socket.create_connection((hostname,443))as sock:
    with context.wrap_socket(sock,server_hostname=hostname)as sssock:
        print(sssock.version())
```

OUTPUT:



The screenshot shows the Python IDLE Shell interface. The title bar reads "IDLE Shell 3.10.7". The menu bar includes File, Edit, Shell, Debug, Options, Window, and Help. The main window displays the following text:

```
File Edit Shell Debug Options Window Help
Python 3.10.7 (tags/v3.10.7:6cc6b13, Sep 5 2022, 14:08:36) [MSC v.1933 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> ===== RESTART: D:\programs\INS\p10.py =====
TLSv1.3
>>> |
```