Practical 6:

Setting up Wireless Access Point using Raspberry Pi

Step 1: Install and update Raspbian

check for updates and ugrades:

```
sudo apt-get update
sudo apt-get upgrade
```

If you get an upgrade, It’s a good idea to reboot with **sudo reboot**.

Step 2: Install hostapd and dnsmasq

These are the two programs we’re going to use to make your Raspberry Pi into a wireless access point. To get them, just type these lines into the terminal:

```
sudo apt-get install hostapd
sudo apt-get install dnsmasq
```

Both times, you’ll have to hit y to continue. hostapd is the package that lets us create a wireless hotspot using a Raspberry Pi, and dnsmasq is an easy-to-use DHCP and DNS server.

We’re going to edit the programs’ configuration files in a moment, so let’s turn the programs off
before we start tinkering:

```
sudo systemctl stop hostapd
sudo systemctl stop dnsmasq
```

Step 3: Configure a static IP for the wlan0 interface

For our purposes here, I’m assuming that we’re using the standard home network IP addresses, like 192.168.###.###. Given that assumption, let’s assign the IP address **192.168.1.10** to the wlan0
interface by editing the dhcpcd configuration file. Start editing with this command:

```
sudo nano /etc/dhcpcd.conf
```

Now that you’re in the file, add the following lines at the end:

```
interface wlan0
static ip_address=192.168.1.10/24
denyinterfaces eth0
denyinterfaces wlan0
```

(The last two lines are needed in order to make our bridge work –- but more on that in Step 8.)

After that, press **Ctrl+X**, then **Y**, then **Enter** to save the file and exit the editor.

Step 4: Configure the DHCP server (dnsmasq)

We're going to use dnsmasq as our DHCP server. The idea of a DHCP server is to dynamically distribute network configuration parameters, such as IP addresses, for interfaces and services.

dnsmasq's default configuration file contains a lot of unnecessary information, so it's easier for us to start from scratch. Let's rename the default configuration file and write a new one:

```
sudo mv /etc/dnsmasq.conf /etc/dnsmasq.conf.orig
sudo nano /etc/dnsmasq.conf
```

You'll be editing a new file now, and with the old one renamed, this is the config file that dnsmasq will use. Type these lines into your new configuration file:

```
interface=wlan0
  dhcp-range=192.168.0.11,192.168.0.30,255.255.255.0,24h
```

The lines we added mean that we're going to provide IP addresses between 192.168.0.11 and 192.168.0.30 for the wlan0 interface.

Step 5: Configure the access point host software (hostapd)

Another config file! This time, we're messing with the hostapd config file. Open 'er up:

```
sudo nano /etc/hostapd/hostapd.conf
```

This should create a brand new file. Type in this:

```
interface=wlan0
bridge=br0
hw_mode=g
channel=7
wmm_enabled=0
macaddr_acl=0
auth_algs=1
ignore_broadcast_ssid=0
wpa=2
wpa_key_mgmt=WPA-PSK
wpa_pairwise=TKIP
rsn_pairwise=CCMP
ssid=RpiIoTwiFi
wpa_passphrase=Rpi@wifi123
```

Note that where I have “NETWORK” and “PASSWORD,” you should come up with your own names. This is how you’ll join the Pi’s network from other devices.

We still have to show the system the location of the configuration file:

```
sudo nano /etc/default/hostapd
```

In this file, track down the line that says #DAEMON_CONF=”” – delete that # and put the path to our config file in the quotes, so that it looks like this:

```
DAEMON_CONF="/etc/hostapd/hostapd.conf"
```

The # keeps the line from being read as code, so you’re basically bringing this line to life here while giving it the right path to our config file.

Step 6: Enable internet connection

Now the Raspberry Pi is acting as an access point to which other devices can connect. However, those devices can’t use the Pi to access the internet just yet. To make the possible, we need to build a bridge that will pass all traffic between the wlan0 and eth0 interfaces.

To build the bridge, let’s install one more package:

```
sudo apt-get install bridge-utils
```

We’re ready to add a new bridge (called br0):

```
sudo brctl addbr br0
```

Next, we’ll connect the eth0 interface to our bridge:

```
sudo brctl addif br0 eth0
```

Finally, let’s edit the interfaces file:

```
sudo nano /etc/network/interfaces
```

...and add the following lines at the end of the file:

```
auto br0
iface br0 inet manual
bridge_ports eth0 wlan0
```

Step 9: Reboot

Now that we’re ready, let’s reboot with **sudo reboot.**

Now your Pi should be working as a wireless access point. Try it out by hopping on another device and looking for the network name you used back in step 5.