Run a Program On Your Raspberry Pi At Startup :

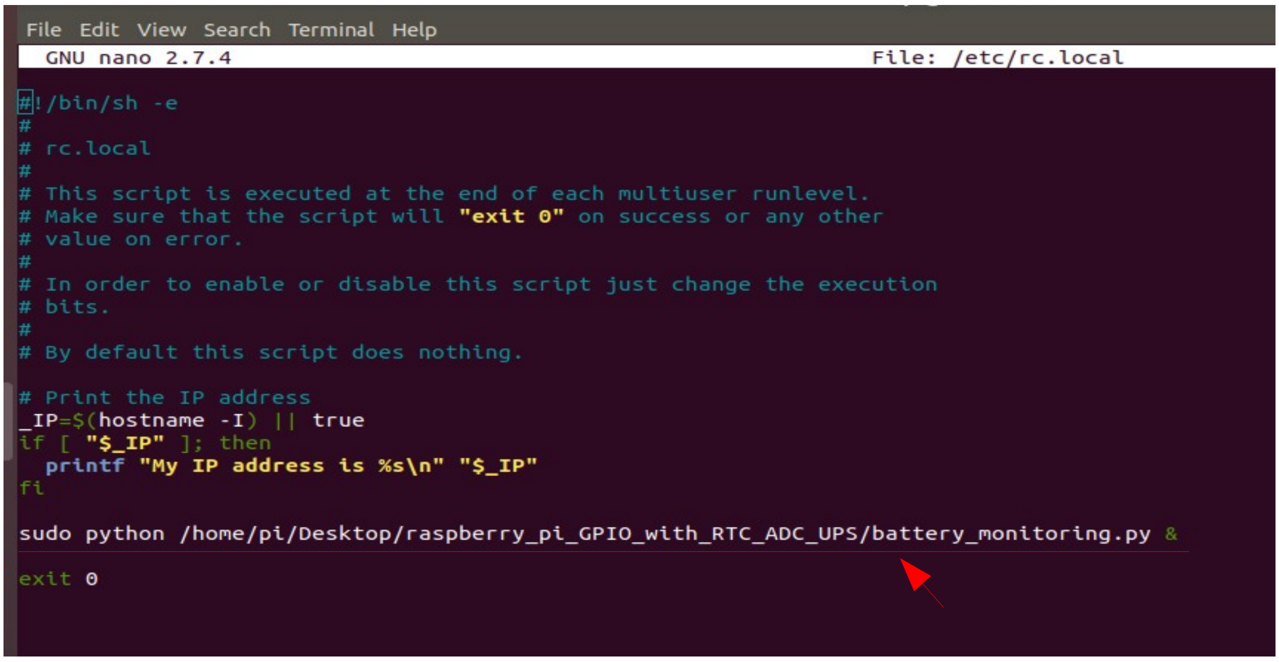
The easy method to run a program on your Raspberry Pi at startup is to use the file **rc.local**. In order to have a command or program run when the Pi boots, you can add commands to the **rc.local** file. This is especially useful if you want to power up your Pi in headless mode (that is without a connected monitor), and have it run a program without configuration or a manual start.

Editing rc.local :

On your Pi, edit the file /etc/rc.local using the editor of your choice. You must edit it with root permissions:

sudo nano /etc/rc.local

Add commands as shown in the figure to execute the python program, preferably using absolute referencing of the file location (complete file path are preferred). Be sure to leave the line **exit 0** at the end, then save the file and exit. In nano, to exit, type Ctrl-x, and then Y.



```
#!/bin/sh -e
#
# rc.local
#
# This script is executed at the end of each multiuser runlevel.
# Make sure that the script will "exit 0" on success or any other
# value on error.
#
# In order to enable or disable this script just change the execution
# bits.
#
# By default this script does nothing.

# Print the IP address
_IP=$(hostname -I) || true
if [ "$_IP" ]; then
  printf "My IP address is %s\n" "$_IP"
fi

sudo python /home/pi/Desktop/raspberry_pi_GPIO_with_RTC_ADC_UPS/battery_monitoring.py &

exit 0
```

If your program runs continuously (runs an infinite loop) or is likely not to exit, you must be sure to fork the process by adding an ampersand ("&") to the end of the command, like:

sudo python /home/pi/Desktop/raspberry_pi_GPIO_with_RTC_ADC_UPS/battery_monitoring.py &

Here in the above case, the python code was in the folder named as raspberry_pi_GPIO_with_RTC_ADC_UPS which is on the desktop so we add added the full file path. Like this you must also enter the file path correctly .
The Pi will run this program at bootup, and before other services are started. If you don’t include the ampersand and if your program runs continuously, the Pi will not complete its boot process. The ampersand allows the command to run in a separate process and continue booting with the main process running.

Now reboot the Pi to test it:

```
sudo reboot
```

Also, be sure to reference absolute file names rather than relative to your home folder. For example use `/home/pi/myscript.py` instead of `myscript.py`.

If you add a script into /etc/rc.local, it is added to the boot sequence. If your code gets stuck then the boot sequence cannot proceed. So be careful as to which code you are trying to run at boot and test the code a couple of times. You can also get the script’s output and error written to a text file (say log.txt) and use it to debug.

```
sudo bash -c 'python
/home/pi/Desktop/raspberry_pi_GPIO_with_RTC_ADC_UPS/battery_monitoring.py >
/home/pi/Desktop/raspberry_pi_GPIO_with_RTC_ADC_UPS/battery_monitoring.log
2>&1' &
```