**Raspberry Pi Session – 4th May**

**What we are going to do today**

We are going to write a program to accept input from the keyboard, check that the input is valid. Based on input, a command is sent to the serial port and check for a response. Take some action depending on the response.

We are going to start with a simple program to accept input and do some validation checks.

Then we will create a function which will return either True or False, depending on whether the input command is valid or not

Finally we are going to format the input command and send it to the serial port. Check for a response from the serial port and take action depending on response.

Remember

* Use meaningful names for variables
* Put comments in the program to explain what is happening.
* A good Python reference site is <http://www.learnpython.org/>

**Step 1 – Basic program to accept input and validate that it is in the correct format.**

The format of the input command should be

Character 1 is either U or L

Characters 2,3,4 must be numeric and should be less than 255

*Valid commands are U123, L255, L001*

*Invalid commands are A234, U345*

The program needs to have the following logic:

Allow a command to be entered from keyboard

Perform some validation checks on the command entered. These should include the following:

* Check that the length of the command is 4 characters
* The first character must be either U or L
* The remaining 3 characters must be numeric and should be less than 255

If any of the above checks fail, exit the program

**Step 2 – Create a function to do the validation checks**

A function is a block of code that can be used to perform an action. You can pass data (arguments) to a function, which can be used by code in the function. A function can also return a value. Python contains many built-in functions ie. print(), len().

Our function will be called *ValidateInput* and will accept 1 argument call *StrToValidate*. The function will contain a number of the validation checks that were created in step 1 above (cut and paste these in to the body of the function).

You also need the function to return True or False, depending on whether the *StrToValidate* is valid or not.

Sample code to create ValidateInput function.

*def ValidateInput (StrToValidate):*

*if len(StrToValidate) != 4;*

*return False*

Add further validation checks to ensure 1st character is U or L and remaining 3 characters must be numeric and should be less than 255

**Step 3 – Add functionality to format command and write to serial port. Read response from serial port.**

The code to format the command that will be send to the Arduino is listed below, which you will need to include in your program.

You will also need to open the serial port, set the speed so it matches the Arduino.

Write the command to the serial port

Read the serial port and depending on value take some action.

**Further work**

Create loop to allow commands to be entered.

If QUIT is entered then exist program. Suggest doing this check in ValidateInput function and setting global variable.

**Sample Program –Session 3-1.py**

# Program to accept input from keyboard.

# The format of the input string should be

# Input string should be 4 chracters

# 1st character is either U or L

# 2nd, 3rd and 4th characters are numeric and less than 255

# Import modules for serial, time and sys

import serial

import time

import sys

# Define variable to determine if input string is valid or not.

# Initially set this to True. It may be set to False during the validation checks

# InputStrValid = True

# Accept input from keyboard and store in RawInputCommand

RawInputCommand = raw\_input("Please enter a command ")

# Validate the input string.

# If input string is not 4 character then print message and exit

if len(RawInputCommand) != 4:

print "Input command must be 4 characters"

exit(0)

# Check that 1st character is either U or L. You need to write the code for this check

# Check that characters 2,3,4 are digits. You need to write the code for this check

# Check that characters 2,3,4 are less than 255. # You need to write the code for this check

# All validation checks have been successful, print message

print ("Input command is valid")

**Sample Program –Session 3-3.py**

# Program to accept input from keyboard.

# Call function ValidateInput to perform the following checks

# The format of the input string should be

# Input string should be 4 chracters

# 1st character is either U or L

# 2nd, 3rd and 4th characters are numeric and less than 255

# Import modules for serial, time and sys

import serial

import time

import sys

# Function to validate input string

def ValidateInput (StrToValidate):

# If input string is not 4 character , print message and return False

if len(StrToValidate) != 4:

print "Command must be 4 characters"

return False

# If 1st character is not U or L, print message and return False

if (StrToValidate[0] != "L") and (StrToValidate[0] != "U"):

print "First character must be L or U"

return False

# If 2,3, & 4 character are not digits, print message and return False

if not ((StrToValidate[1] + StrToValidate[2] + StrToValidate[3]).isdigit()):

print "Not Digits"

return False

# If 2,3, & 4 character are greater than 255, print message and return False

if int(StrToValidate[1] + StrToValidate[2] + StrToValidate[3]) > 255:

print "character 2,3,4 must be less than 255"

return False

# After all validation checks have been run and passed return True

return True

# Create and open the serial port, setting the communication rate

# For this to work properly, the speed you specify here needs to match exactly with the speed on the Arduino

# Note that not all values are possible ... look up the possible values for serial baud rate online

# Serial ports on Linux/Unix like machines have names like /dev/ttyXXXX

# Serial ports on Windows machines have names like COM3 or COM8

# Modify this to matcht the serial port that the Arduino is actually connected to

# serialport = serial.Serial("/dev/ttyAMA0", 115200, timeout=0.5)

# serialport = serial.Serial("COM3", 115200, timeout=0.5)

serialport = serial.Serial('/dev/ttyACM0', 11520, timeout=0.5)

# Wait a 2 seconds, in case the arduino reboots on serial port opening.

time.sleep(2) # short pause in case arduino reboots on serial port opening, value can be changed or line removed

# Accept input from keyboard and store in RawInputCommand

RawInputCommand = raw\_input("Please enter a command ")

# Call ValidateInput function and pass input string as parameter.

if ValidateInput(RawInputCommand):

# All validation checks have been successful, print message

print ("Input command is valid")

# In this example, we will make a binary string to send to the serial port

# This matches with a command receiver on our Arduino which expects to see a command string in a certain format

# Our standard simple format is <sync byte><length byte><command bytes>

# We use the Python "bytearray" to hold our string of bytes to send to the Arduino

# Our command is 4 bytes long:

# .. byte[0] 0xFC (a special marker indicating the start of the command)

# .. byte[1] 2 (the length: the number of bytes to come)

# .. byte[2] 'U' (the character U, which is a command that the Arduino understands

# .. byte[3] 0 (a "parameter" for the command U, which the Arduino understands

# commandValue holds the numeric parameter - characters 2,3,4 in input string

commandValue =int(RawInputCommand[1] + RawInputCommand[2] + RawInputCommand[3])

commandBuffer = bytearray()

commandBuffer.append(0xFC)

commandBuffer.append(2)

commandBuffer.append(ord(RawInputCommand[0]))

commandBuffer.append(commandValue)

serialport.write(commandBuffer) # write the command buffer

# In response to this command, the Arduino sends back a response.

# Just print it out. To really deal with this response, it should have a defined format, and we should process it to understand what it means

response = serialport.readlines(None) # reads from serial port

print(response) # outputs data received from serial port