**Virtual Pi2Go: Using Logic in Conditions**



**AIM:** After completing this worksheet you should be able to use logic to construct more complex conditions for if and while statements.

**You Need:** To complete this worksheet you need to have a virtual Pi2Go simulator (see WS1), and be able to use files to store programs (see WS5). You also need to know the commands to operate the Pi2Go motors, LEDs and sensors (see WS3 & WS4) and how to use if and while statements in Python (see WS7 & WS8).

**If the simulator isn’t already running:**

Start it, and select the Pi2Go robot and default\_world.xml

Now open a new IDLE window.

**Problem:** When you program a real Pi2Go robot it has to be connected to a keyboard and monitor which is very cumbersome.

Ideally you want to have time after the program has started running to disconnect everything, before the robot starts to move.

Consider the program below

import simclient.simrobot as pi2go, time

pi2go.init()

while not (pi2go.getSwitch()):

print("Waiting")

pi2go.setAllLEDs(4095, 4095, 4095)

time.sleep(1)

pi2go.setAllLEDs(0, 0, 0)

**Note:** On the real robot the switch is ON when pressed down and OFF when not pressed down. In the simulator you click on the switch to turn it ON and then click on it again to turn it OFF.



**Question 1:** What does this program do?

**Question 2:** Execute the program. Did it do what you expected? YES/NO.

**Exercise 1:**  Modify the program by adding a second while loop so that the LEDs stay on until the switch is switched on a second time.

**Logic in conditions:** You can use *logic* to make the conditions of your Python while and if statements more flexible. The main *logical operators* are not, and and or.

We can use these to build up logical expressions so, for instance:

|  |  |
| --- | --- |
| pi2go.getSwitch() | returns True if the switch is on |
| not (pi2go.getSwitch()) | returns True if the switch is off |
| (pi2go.getSwitch() or pi2go.irCentre()) | returns True if the switch is on *or* there is an obstacle in the centre |
| (pi2go.getSwitch() and pi2go.irCentre()) | returns True if the switch is on *and* there is an obstacle in the centre |

**Exercise 2:** Modify your program again so that if the switch is on or there is an obstacle then the Pi2Go reverses. Then when the switch is turned on (a second time if it was pressed the first time), the Pi2Go stops moving.

**Question 3:** What expression are you using in the condition (if statement)?

You can make conditions more and more complex by composing the logical operators.

**Exercise 3:** Modify your program again so that if the switch is on and there is not an obstacle then the Pi2Go moves forward. Then when the switch is turned on a second time, Pi2Go stops moving.



**Question 4:** What expression are you using in the condition?

**Remember:** When you have finished working with your robot type: **pi2go.cleanup()**

When you want to exit the simulator, select the simulator window and type Q.



University of Liverpool, 2019

This work is licensed under a [Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License](https://creativecommons.org/licenses/by-nc-sa/4.0/).