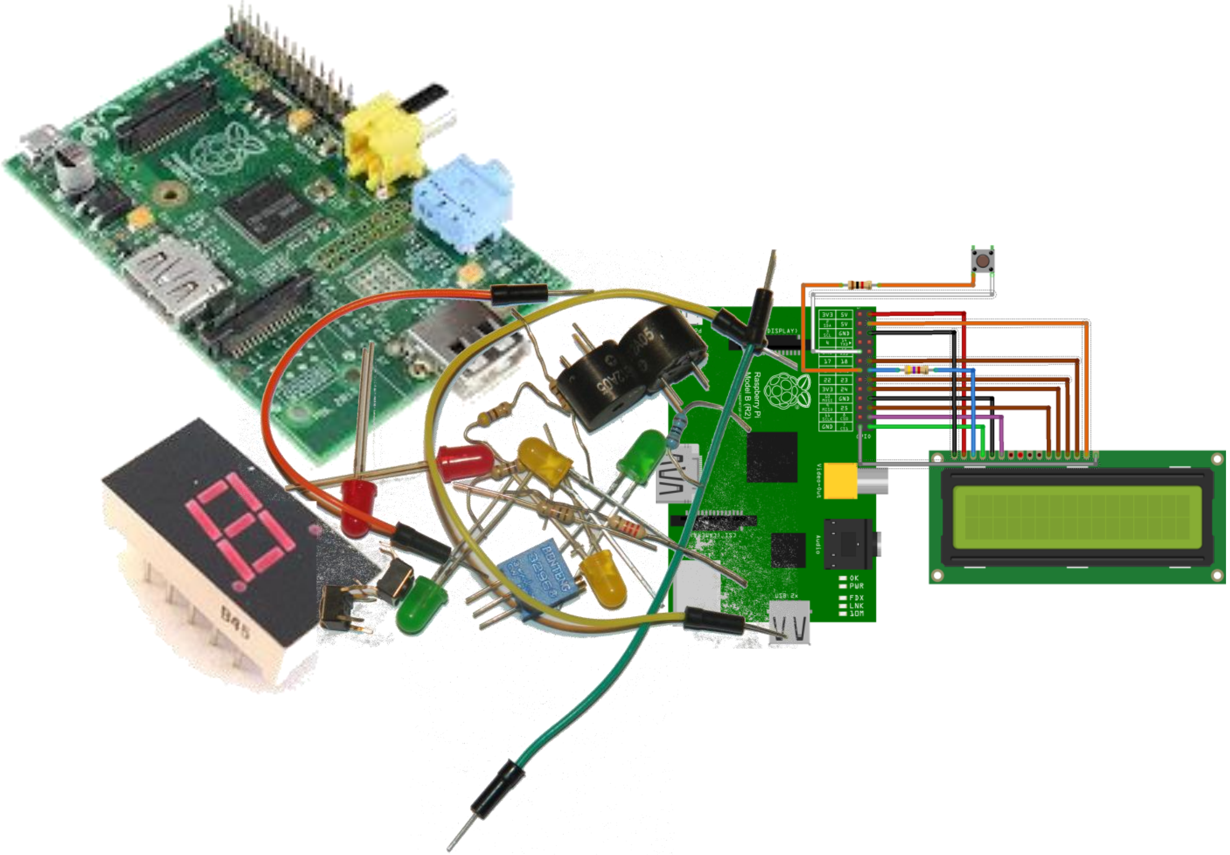
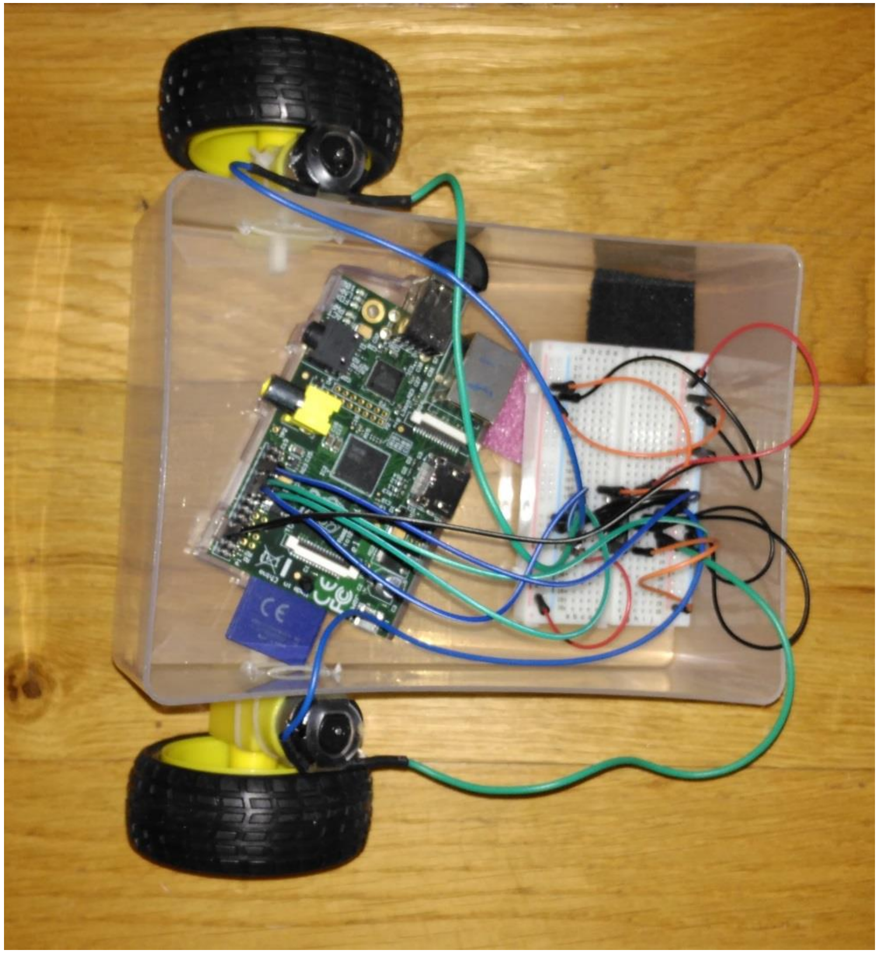
Raspberry Pi and Electronics



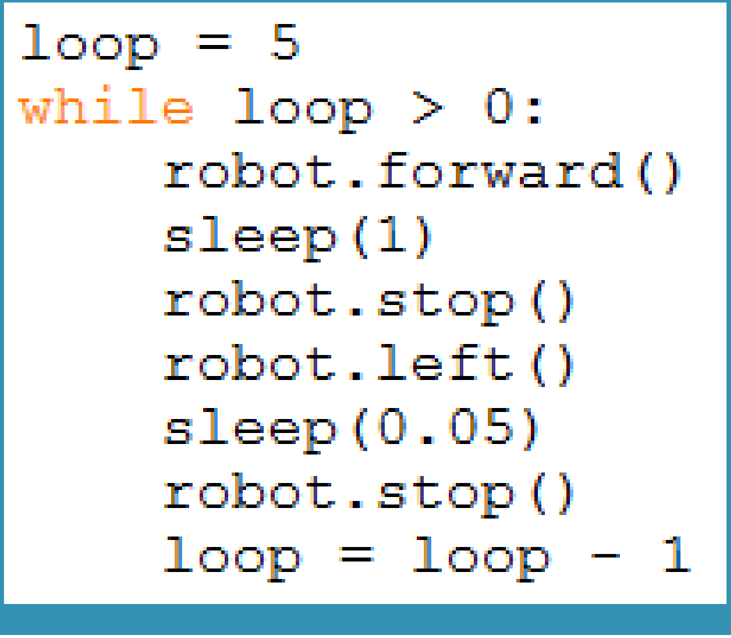
Robot

We saw last week how we could solve the problem of our robots going in arcs by making a small correction. We will take a look at this and then we will build functions that make controlling our robots more user friendly



Robot

We got the robot to go forward for about 30 CM then we made a small correction. We used a while loop to repeat this. The distance travelled would then be 30 CM multiplied by the amount of times the loop is repeated.



```
loop = 5
while loop > 0:
    robot.forward()
    sleep(1)
    robot.stop()
    robot.left()
    sleep(0.05)
    robot.stop()
    loop = loop - 1
```

Functions

Imagine if instead of robot.forward()
sleep(1)
We could do something like robot_forward(5) which would tell our robot to forward by 5 steps(a step could be any distance we decide) or instead of robot.left()
sleep(0.05)
We had robot_turn(-45) to turn our robot to the left by 45 degrees.
Well we can do this using Functions

Functions

Why are functions important?

- The program is easier to understand.
- You can reuse code easily.
- Easier to design and test code.
- Top-down design
- Build from the bottom up

Functions

What can a Function do?

1. It can perform some computational task.
2. It can return a result and/or modify parameters.
3. It can take in Data (called Arguments)

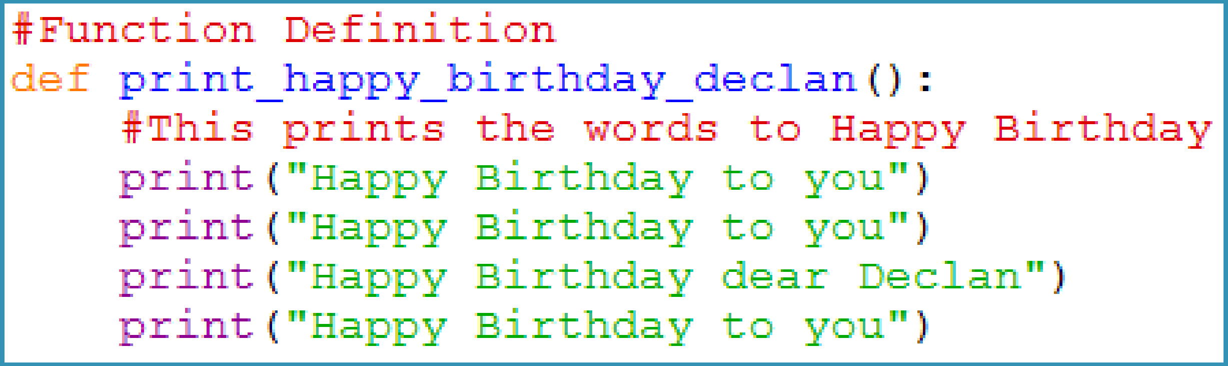
Functions

Function Variables

Just like the values in our programs are forgotten after the program ends, variables created inside a function are forgotten after the execution leaves the function. Not only that, but when execution is inside the function, we cannot change the variables outside of the function, or variables inside other functions. Variables defined inside a function are called **Local Variables.** Variables defined outside of a function are called **Global Variables**

Functions

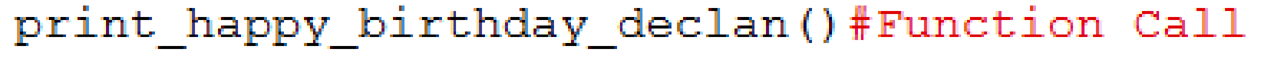
Writing a Function



```
#Function Definition
def print_happy_birthday_declan():
    #This prints the words to Happy Birthday
    print("Happy Birthday to you")
    print("Happy Birthday to you")
    print("Happy Birthday dear Declan")
    print("Happy Birthday to you")
```

This is the function's definition (where we put the def statement and the def-block) it has to come in the code before you call the function.

Functions



```
print_happy_birthday_declan()#Function Call
```

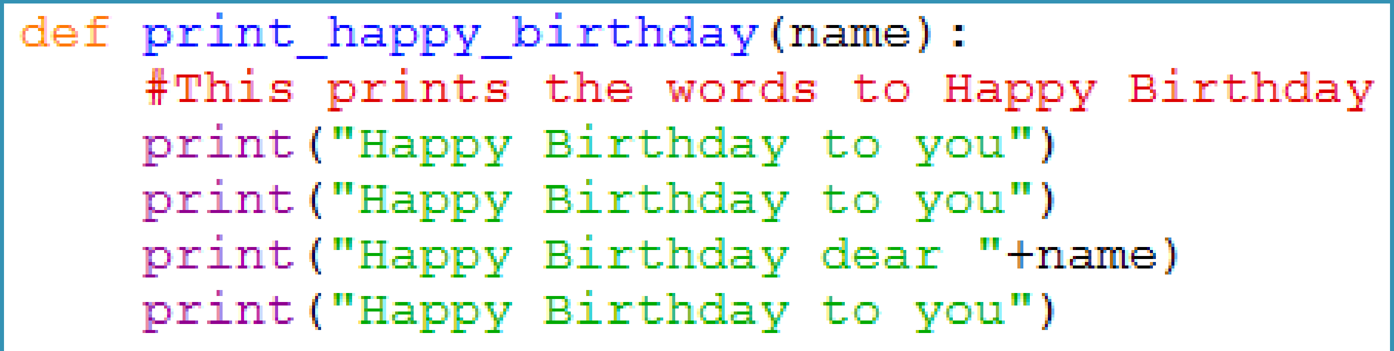
This is the Function call.

We call our own functions the same way we call Python’s built in built functions

Functions

Function Arguments

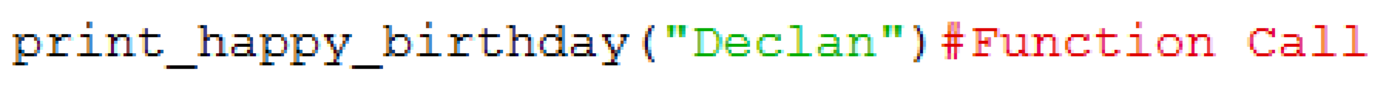
We can sent variables to a function for it to perform some task on, this piece of data is called a Function Argument.



```
def print_happy_birthday(name):
    #This prints the words to Happy Birthday
    print("Happy Birthday to you")
    print("Happy Birthday to you")
    print("Happy Birthday dear "+name)
    print("Happy Birthday to you")
```

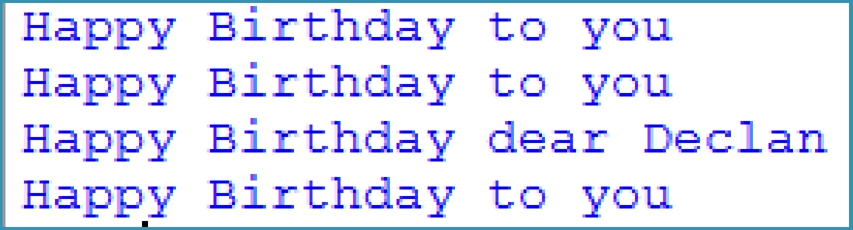
Functions

We add our argument when we call it



```
print_happy_birthday("Declan")#Function Call
```

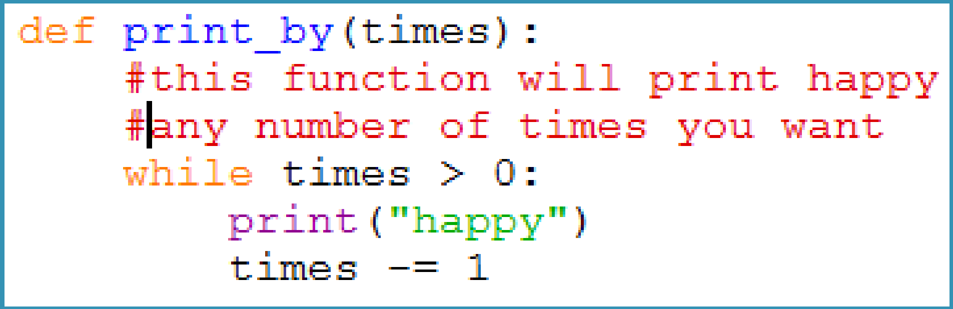
And we get the following.



```
Happy Birthday to you
Happy Birthday to you
Happy Birthday dear Declan
Happy Birthday to you
```

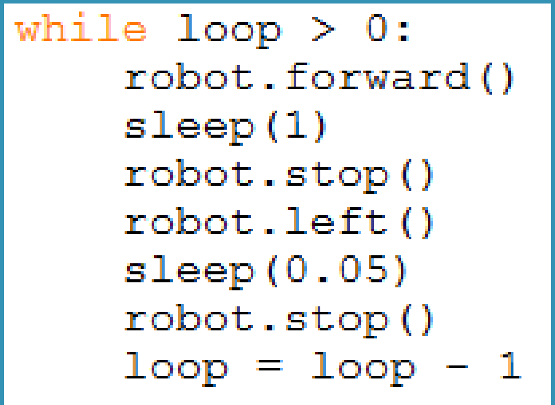
Robot

This function uses an argument to control a loop



```
def print_by(times):
    #this function will print happy
    #any number of times you want
    while times > 0:
        print("happy")
        times -= 1
```

We can use this code with this



```
while loop > 0:
    robot.forward()
    sleep(1)
    robot.stop()
    robot.left()
    sleep(0.05)
    robot.stop()
    loop = loop - 1
```

To create a really useful robot function