

Lab7 – Create a Simple Game with Python IDLE

Objectives

Part 1: Launch VirtualBox and Enter the I2IoT server VM

Part 2: Create a Simple Game with Python IDLE

Part 3: IDLE for Python

Background

Python, a programming language, allows for simpler statements. Python is very easy to use, powerful, and versatile. It has become the language of choice for many IoT developers. One of the main reasons for the popularity of Python is the developer community; Python developers have created and made available many specific modules that can be imported into any program to immediately lend added functionality.

Scenario

In this lab, you will create a simple game using Python IDLE.

Required Resources

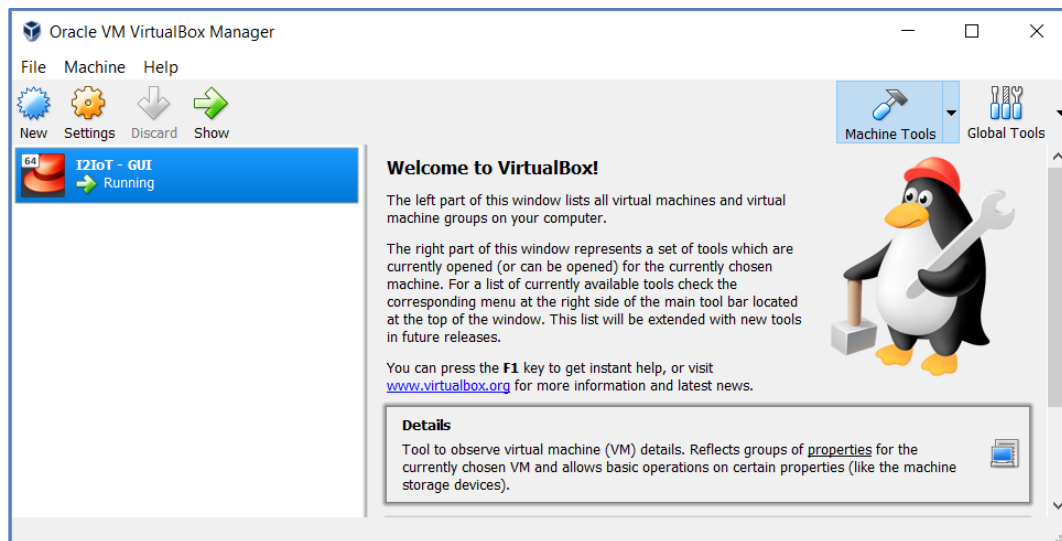
- A modern personal computer with sufficient RAM and with internet access.
- VirtualBox with I2IoT server installed.

Part 1: Launch VirtualBox and Enter the I2IoT server VM

In Part 1, you launch virtualization software VirtualBox and login to the I2IoT server VM.

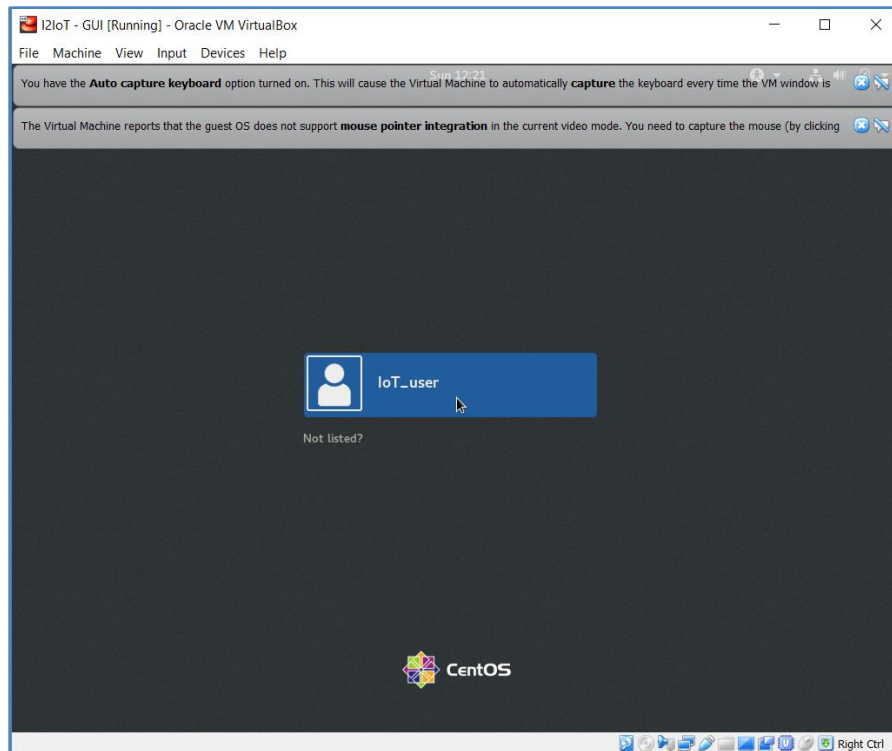
Step 1: Launch VirtualBox.

- After VirtualBox is installed (see Lab 2.1.2.a), the VirtualBox icon should appear on the Desktop. Click the icon to launch the VirtualBox.

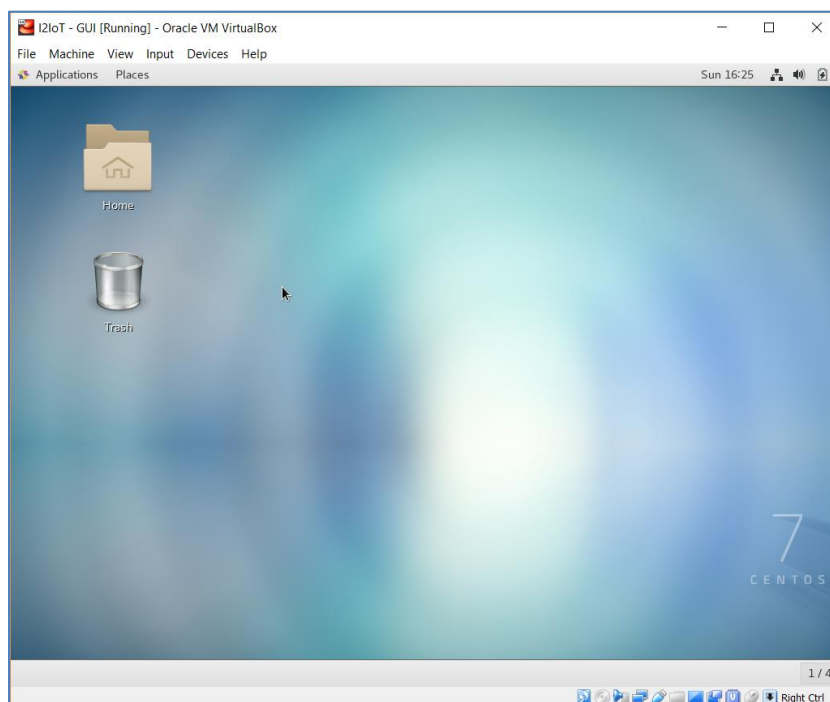


- Click **I2IoT – GUI** on the left pane to launch the server VM.

Lab7 – Create a Simple Game with Python IDLE

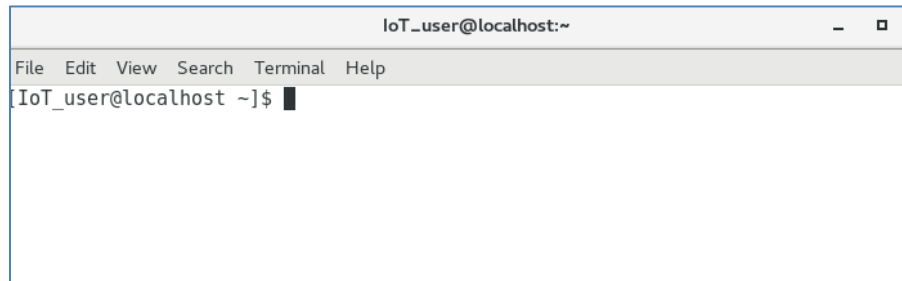


- c. The default username is IoT_user with no password. Click the blue bar **IoT_user** in the middle of the screen to log in to the VM.



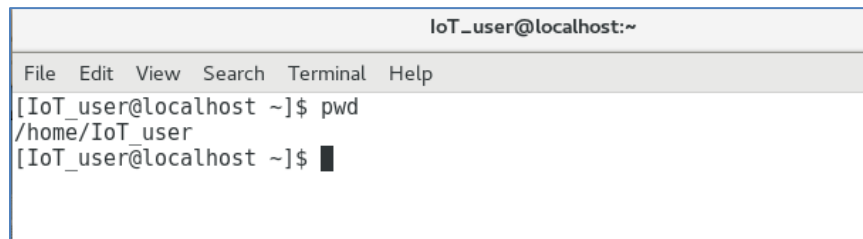
Step 2: Navigate to the user Document directory

- a. To access the command line interface, click **Application** on the menu bar and choose **Terminal**.



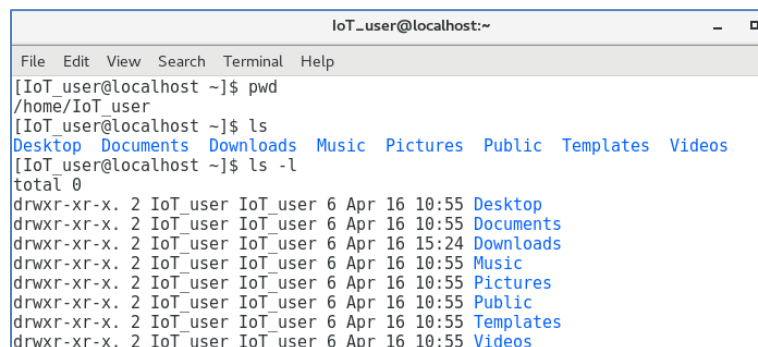
```
IoT_user@localhost:~  
File Edit View Search Terminal Help  
[IoT_user@localhost ~]$
```

- b. Use the **pwd** command to display the current directory.



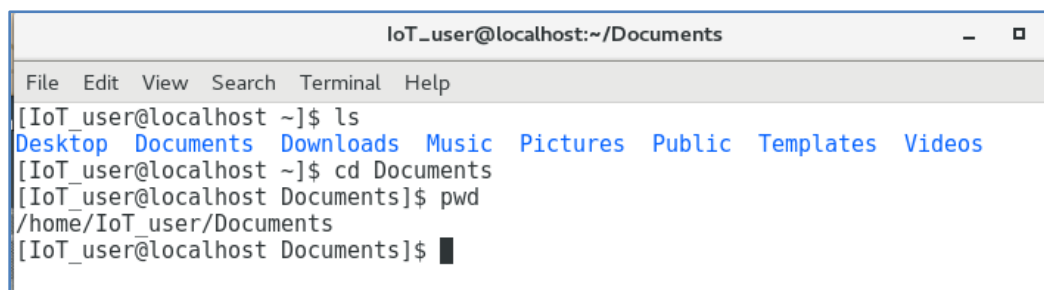
```
IoT_user@localhost:~  
File Edit View Search Terminal Help  
[IoT_user@localhost ~]$ pwd  
/home/IoT_user  
[IoT_user@localhost ~]$
```

- c. Use the **ls** command to show the list of contents in the current directory. Use the **ls** command with the **-l** option to show detailed information about the contents.



```
IoT_user@localhost:~  
File Edit View Search Terminal Help  
[IoT_user@localhost ~]$ pwd  
/home/IoT_user  
[IoT_user@localhost ~]$ ls  
Desktop Documents Downloads Music Pictures Public Templates Videos  
[IoT_user@localhost ~]$ ls -l  
total 0  
drwxr-xr-x. 2 IoT_user IoT_user 6 Apr 16 10:55 Desktop  
drwxr-xr-x. 2 IoT_user IoT_user 6 Apr 16 10:55 Documents  
drwxr-xr-x. 2 IoT_user IoT_user 6 Apr 16 15:24 Downloads  
drwxr-xr-x. 2 IoT_user IoT_user 6 Apr 16 10:55 Music  
drwxr-xr-x. 2 IoT_user IoT_user 6 Apr 16 10:55 Pictures  
drwxr-xr-x. 2 IoT_user IoT_user 6 Apr 16 10:55 Public  
drwxr-xr-x. 2 IoT_user IoT_user 6 Apr 16 10:55 Templates  
drwxr-xr-x. 2 IoT_user IoT_user 6 Apr 16 10:55 Videos
```

- d. Use the **cd Documents** command to change the directory to the /home/IoT_user/Documents directory. Verify by using the command **pwd**.



```
IoT_user@localhost:~/Documents  
File Edit View Search Terminal Help  
[IoT_user@localhost ~]$ ls  
Desktop Documents Downloads Music Pictures Public Templates Videos  
[IoT_user@localhost ~]$ cd Documents  
[IoT_user@localhost Documents]$ pwd  
/home/IoT_user/Documents  
[IoT_user@localhost Documents]$
```

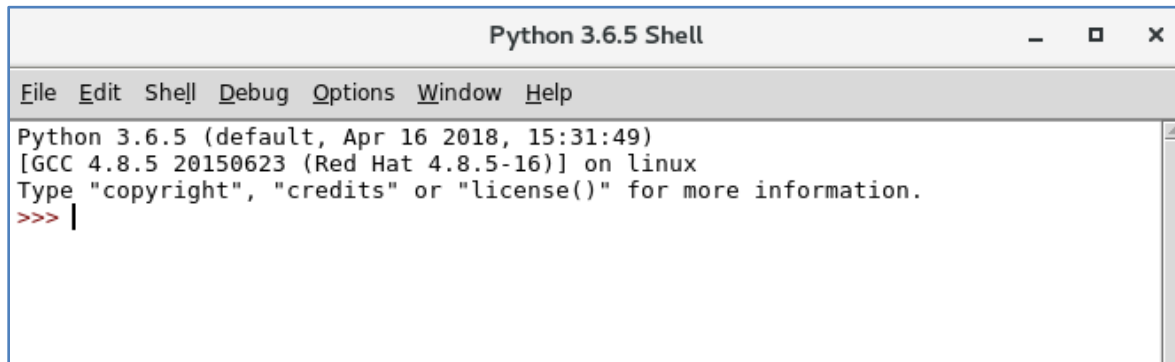
Part 2: Create a Simple Game with Python IDLE

In Part 2, you will create a simple game. The goal of the game is to find a number chosen by a player within 0 and 1024 using the bisection method.

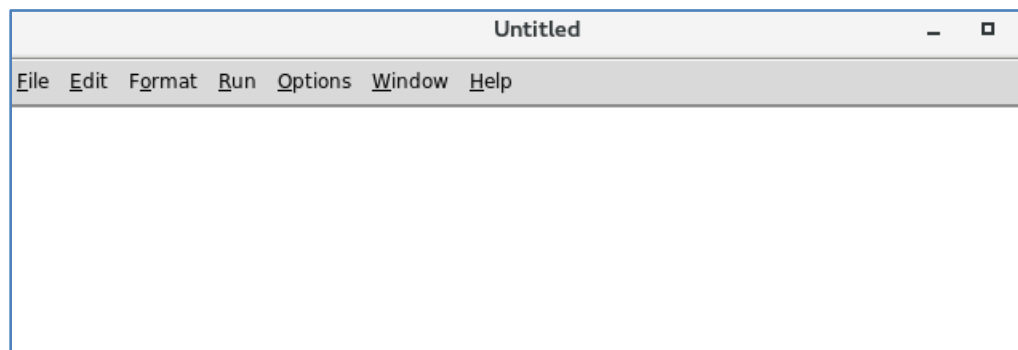
Step 1: Launch IDLE.

- Use the **idle3** command to launch IDLE. By default, it starts in Python Shell, or interactive interpreter, window. You are already familiar with the interactive interpreter.

```
[IoT_user@stueverj-vm2 Documents]$ idle3
```

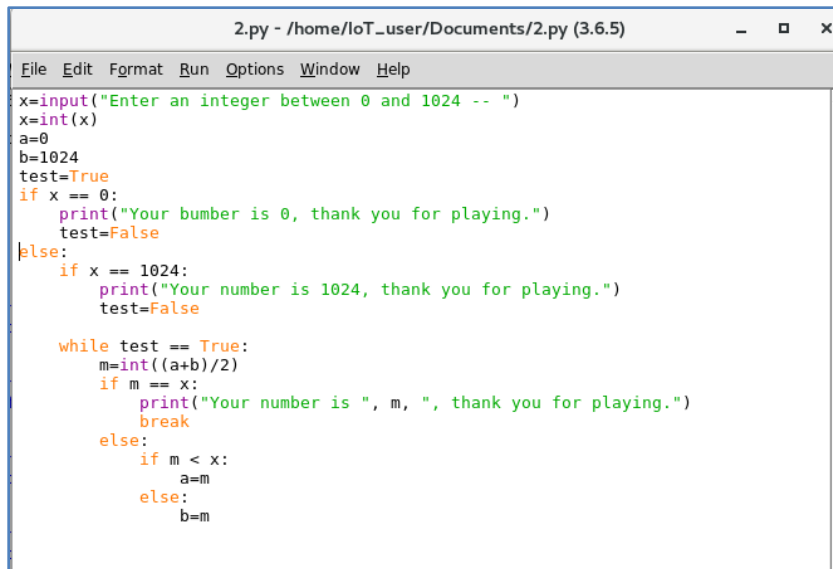


- Click **File** -> **New File** to open a new (Untitled) Python script.



- Type the codes in the script, and note that the codes are color coded with open and close parentheses matched.

Lab7 – Create a Simple Game with Python IDLE



```
2.py - /home/IoT_user/Documents/2.py (3.6.5)
File Edit Format Run Options Window Help
x=input("Enter an integer between 0 and 1024 -- ")
x=int(x)
a=0
b=1024
test=True
if x == 0:
    print("Your bumber is 0, thank you for playing.")
    test=False
else:
    if x == 1024:
        print("Your number is 1024, thank you for playing.")
        test=False
    while test == True:
        m=int((a+b)/2)
        if m == x:
            print("Your number is ", m, ", thank you for playing.")
            break
        else:
            if m < x:
                a=m
            else:
                b=m
```

- d. Click **File** -> **Save**, and save the current script as 2.py in the current directory. Click the **Save** button.
- e. Click **Run** -> **Run Module** (or press F5). The shell window will display the result.



```
Python 3.6.5 Shell
File Edit Shell Debug Options Window Help
>>>
===== RESTART: /home/IoT_user/Documents/2.py =====
Enter an integer between 0 and 1024--463
Your number is 63 , thank you for playing.
>>>
===== RESTART: /home/IoT_user/Documents/2.py =====
Enter an integer between 0 and 1024 --0
Your bumber is 0, thank you for playing.
>>>
===== RESTART: /home/IoT_user/Documents/2.py =====
Enter an integer between 0 and 1024 -- 1024
Your number is 1024, thank you for playing.
>>>
===== RESTART: /home/IoT_user/Documents/2.py =====
Enter an integer between 0 and 1024 -- 72
Your number is 72 , thank you for playing.
>>>
===== RESTART: /home/IoT_user/Documents/2.py =====
Enter an integer between 0 and 1024 -- 12
Your number is 12 , thank you for playing.
>>>
```

- f. Troubleshoot if an error occurs during code syntax evaluation.

Reflection

How to catch if the player enters a number beyond the range of 0 to 1024?

The input should be verified to be within the right range immediately after the player enters the number.

How to catch if the player enters a float number?

The input should be verified to be an integer immediately after the player enters the number.