

Name-Surname: Chih Li

ID: 64011378

Introduction to Computers and Programming, SE Programme

Lab Direction #1

9th August 2021

Part 1: An Introduction to Python

Section 1.1 Why Python?

Python is a modern programming language that was developed by Guido van Rossum (now at Google) in 1990 and first released in 1991. While there are a number of programming languages that might be used for a first course in programming, Python offers a number of features that make it particularly applicable:

- It is interpreted, which for the introductory student means that they can type program commands into a console and immediately see the results. This makes learning details of the language much easier
- The syntax of Python is generally simpler than other languages. The Python's philosophy is "one way to do it" so that the number of details the student has to learn/remember is reduced. This makes code much more natural and more readable. The focus of learning code development should be on its readability, and python supports this.
- Python provides high level data structures and methods that make many programming tasks much easier. This means that students are productive, writing significant code, much more quickly.
- Python has many libraries which make tasks easier. In particular, there are many libraries specific to a field (biology, chemistry, physics, etc.) that make the language useful for "getting work done". A student who knows Python therefore has a host of software available to them for doing anything from graphics, to gene structure matching, to financial accounting, to music and anything in between.
- Python is free! Students can download it from the web for their personal machines. Furthermore, Python does not depend on a particular operating system. Thus students are free to develop their code on Windows, MacOS or Linux and the code will (for the most part) run anywhere

Section 1.2 Where to get it

Python is available for any platform (Windows, Mac, Linux) for free download. The main Python site is <http://www.python.org> and from there you can download a full installation of Python.

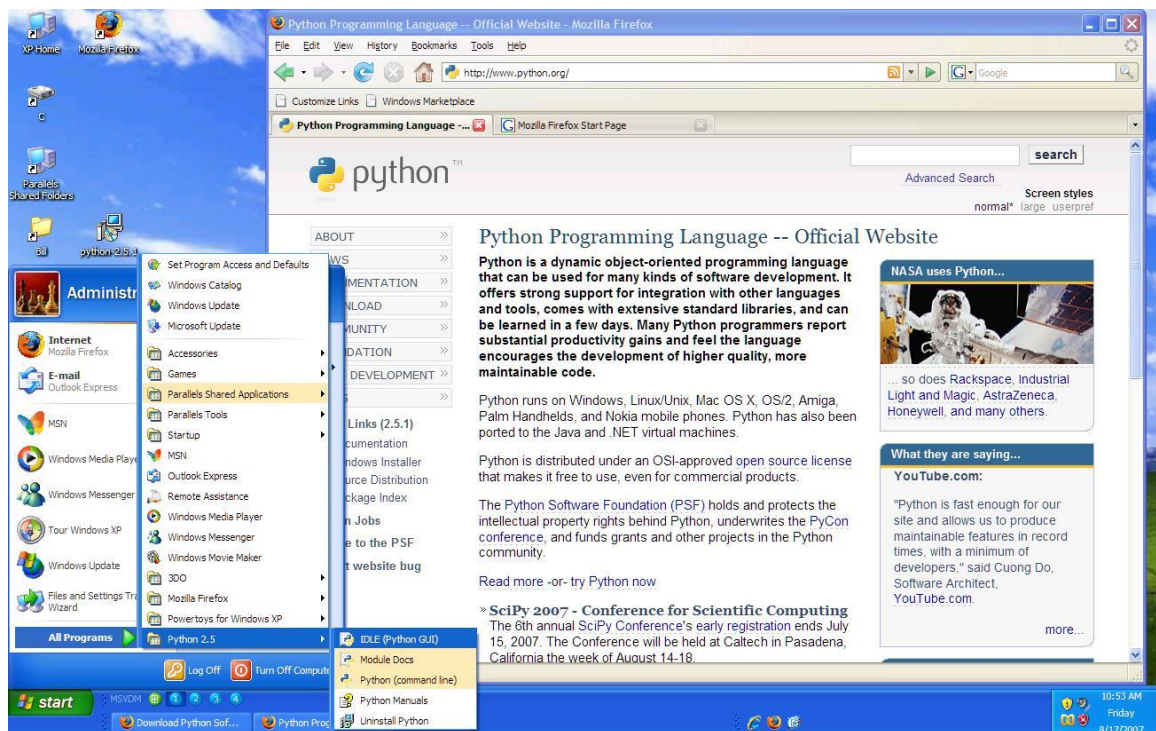
Section 1.3 What you get

When you install python for your computer, you get a number of features:

- a python “shell”, a window into which you can directly type python commands and where interaction between you and programs you write typically occurs.
- a simple editor called IDLE, in which you can type programs, update them, save them to disk and run them. IDLE’s interface is essentially the same on any machine you run it because it is a Python program!
- you get access to all the python documentation on your local computer including:
 - a tutorial to get you started
 - language reference for any details you might want to investigate
 - library reference for modules you might wish to import and use
 - other nifty items

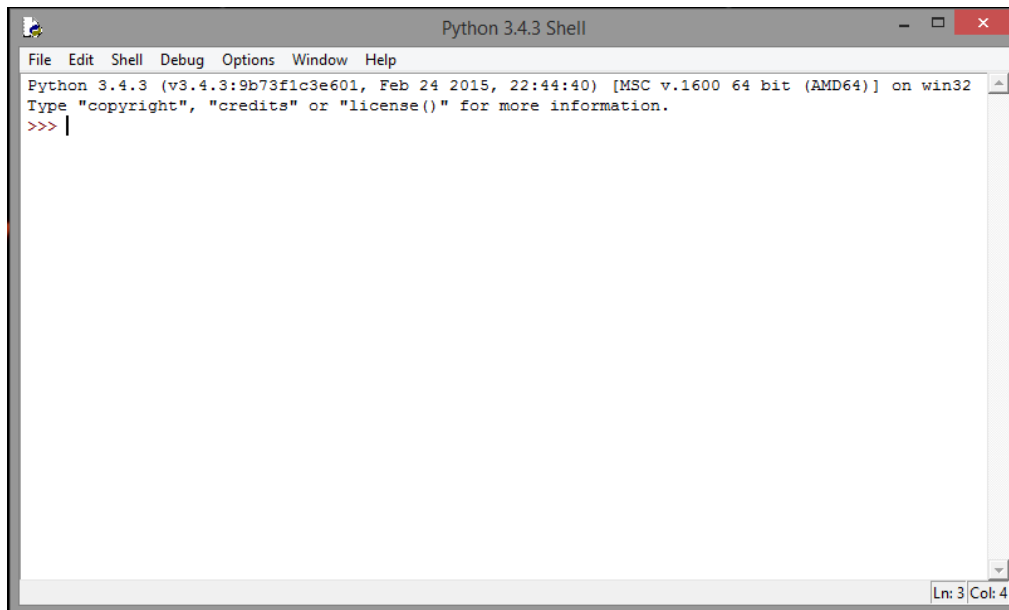
Part 2: Let’s get started

Let’s get started with Python. To start the Python/IDLE combination on Windows, go to Start Menu ⇒ All Programs ⇒ Python 3.X ⇒ IDLE (Python GUI) as shown below



Section 2.1 Working with the shell

The good new is that no matter how you start it, you should get a window that looks like pretty much like the following (this one from XP)

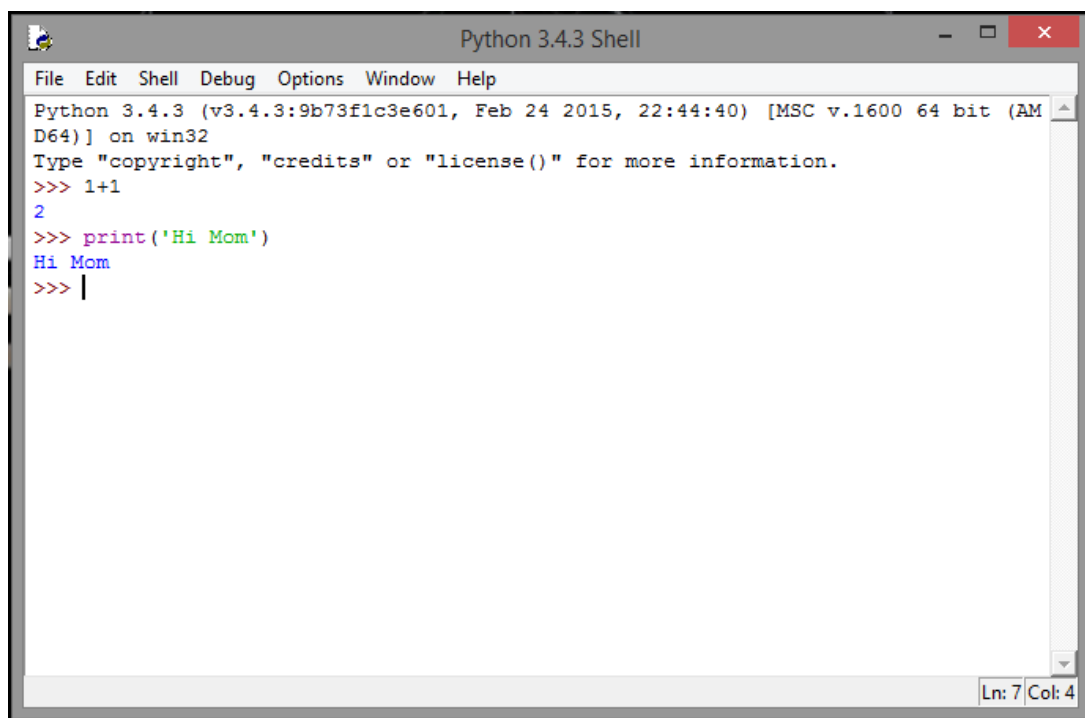


This is the python shell. The shell is interactive; you can type Python commands in the shell and Python will execute them, generating a result. Try typing:

`1 + 1` <Enter Key>

`print('Hi Mom')` <Enter Key>

What output do you get? It should look like the window below



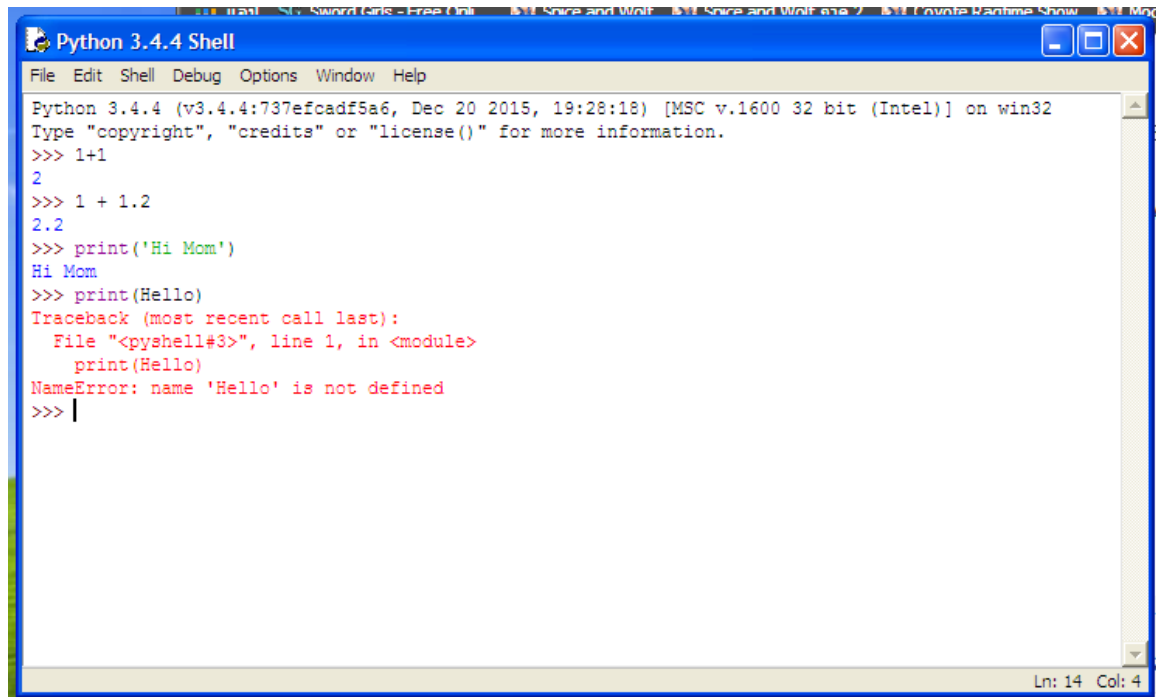
What you type shows up in front of the >>> prompt. When you type something and hit the Enter key, the result shows up on the next line(s). Sometimes you can get some surprising results, sometimes an error. For example, try entering:

1 + 1.2

or perhaps

print(hello)

The results would look like the following:



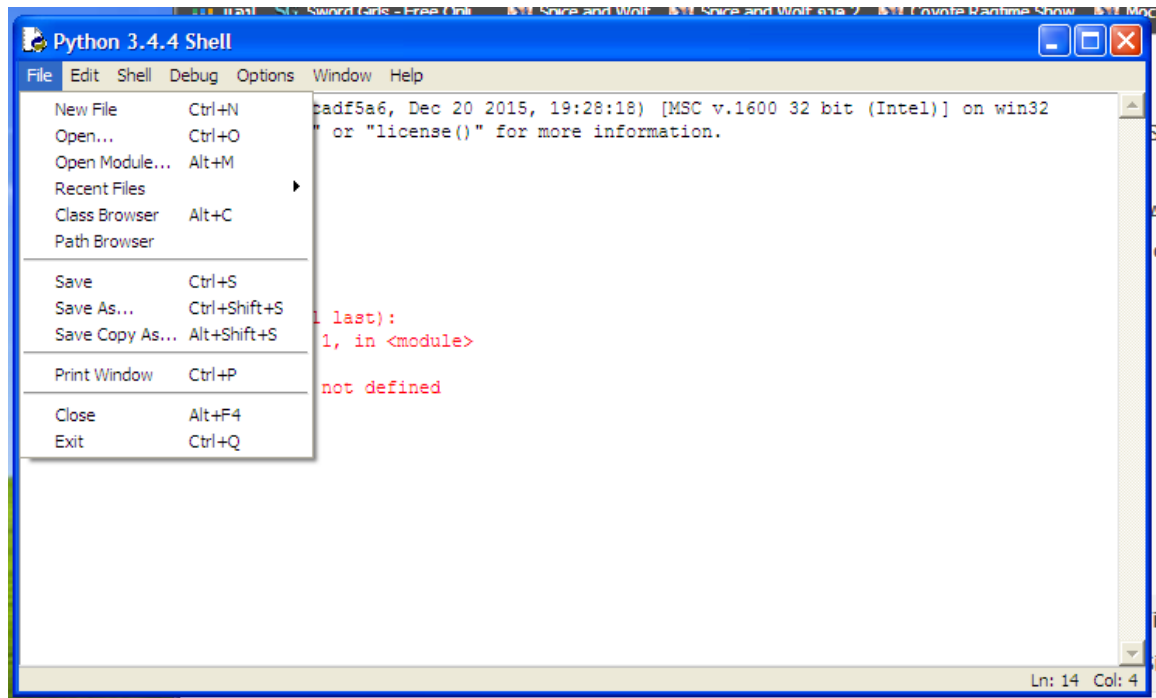
```
Python 3.4.4 Shell
File Edit Shell Debug Options Window Help
Python 3.4.4 (v3.4.4:737efcadf5a6, Dec 20 2015, 19:28:18) [MSC v.1600 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> 1+1
2
>>> 1 + 1.2
2.2
>>> print('Hi Mom')
Hi Mom
>>> print(Hello)
Traceback (most recent call last):
  File "<pyshell#3>", line 1, in <module>
    print(Hello)
NameError: name 'Hello' is not defined
>>> |
```

The last shows that an error occurred. hello was neither a variable nor did it have quotes around it, so print failed to perform (we'll see more on this later).

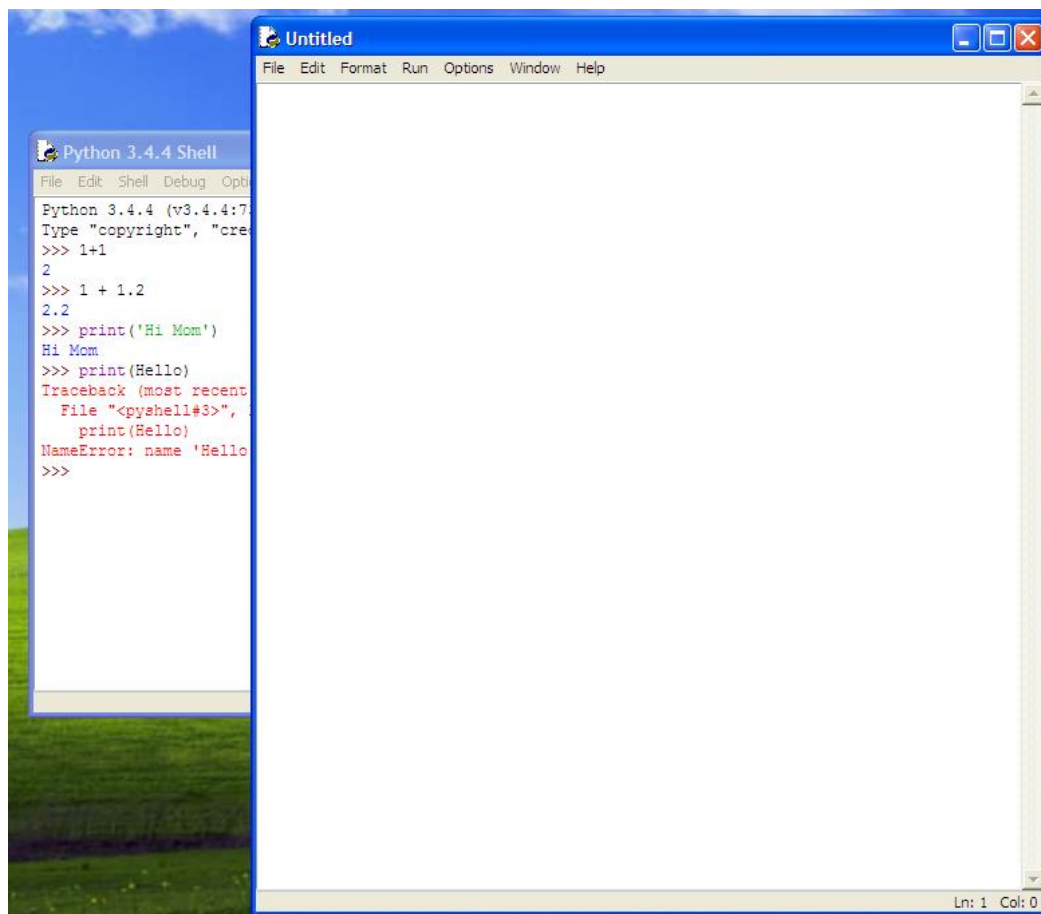
Section 2.2 Making a program

Typing into the shell is useful, but the commands that you write there are not saved as a file and therefore cannot be reused. We need to save our commands in a file so we can run the program again and again, and more importantly turn it in!

To open a file, go to the shell window, left-click on File \Rightarrow New File as shown below



A second window will appear into which you can type python commands. This window is an editor window into which we can type our first program.



There is a tradition in computer science. The first program you run in a new language is the Hello World program. This program does nothing but print the words “Hello World”. It is a tradition because it does very little except focus on the mechanics of writing your first program and running it. Look at the wikibooks page

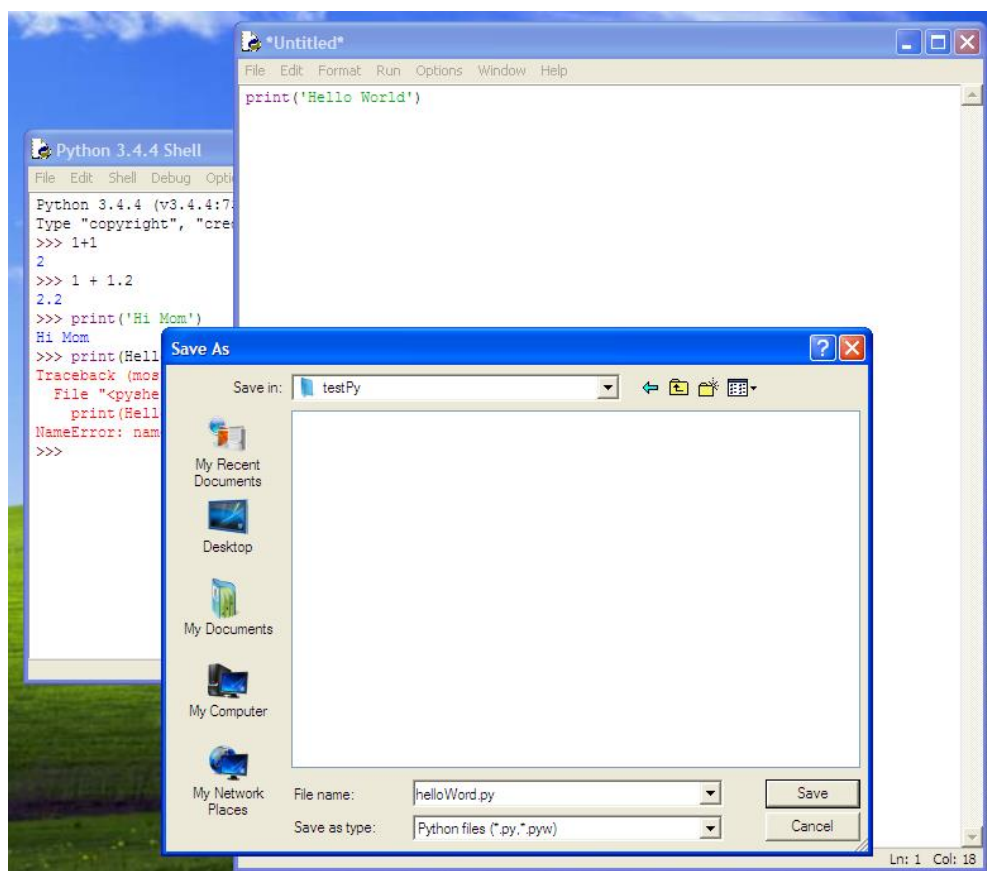
http://en.wikibooks.org/wiki/List_of_hello_world_programs

for more than 200 programming language examples of hello world

In python helloWorld is very easy. Type the following in the “Untitled” window

```
print('Hello World')
```

The phrase after print should be in quotes. Having done that, we should save our first program so that we can use it again. How and where you save a file differs depending on your operating system. For XP, you left click in the Untitled Window the menu File⇒Save As. Your system will look something like the following:

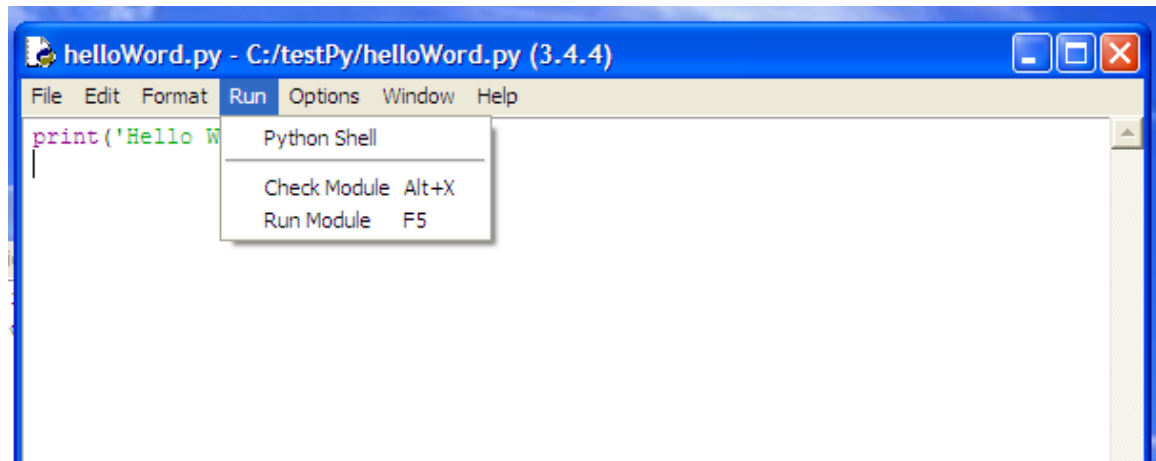


This is on the way to saving the file, not quite completed. It shows the file dialog and there are a few things to note. First, we type the name of the file at the bottom of the dialog. The name I have chosen is helloWorld.py . The .py file extension is important to add. If you do not, then all the nice coloring and formatting you get in the idle editing window will be lost. Second, note that I am selecting the My Computer ⇒ punch ... (H:) drive.

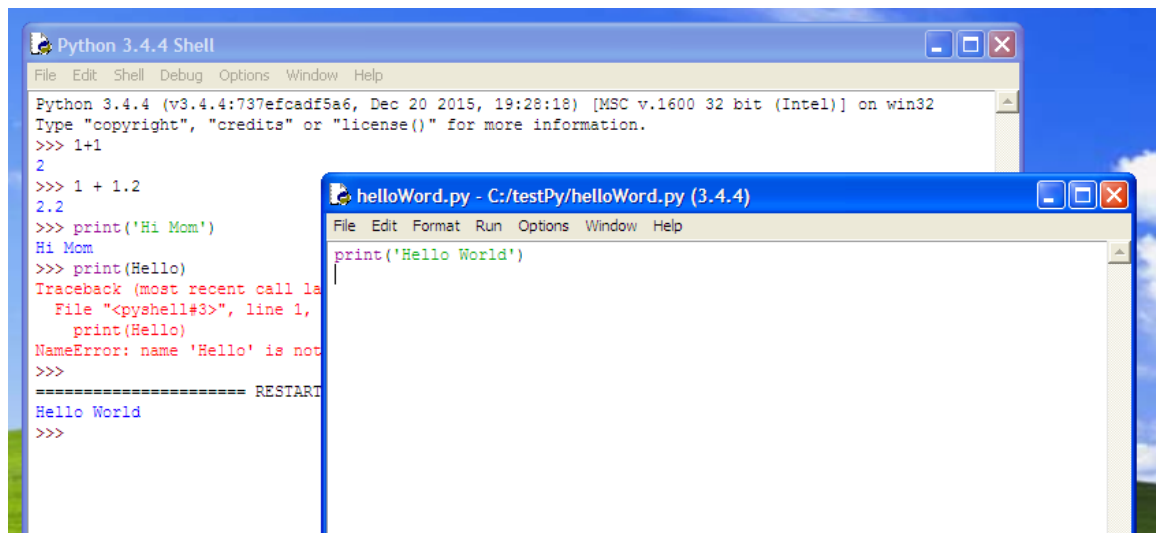
Your H: drive is file space that is stored on the CSE servers. This is most important! You should never save files on the machine you are sitting at (the C: drive). If you do, then you can never get those files except when you are on that particular machine.

The H: drive however is available from any machine you sit at. So **ALWAYS** save your files to your H: drive!

Once saved, you will notice that the title of the editing window changes to the file name you used to save your program. You can then run your module to see what your program produces. To run the program, select the editing window menu Run⇒Run Module.



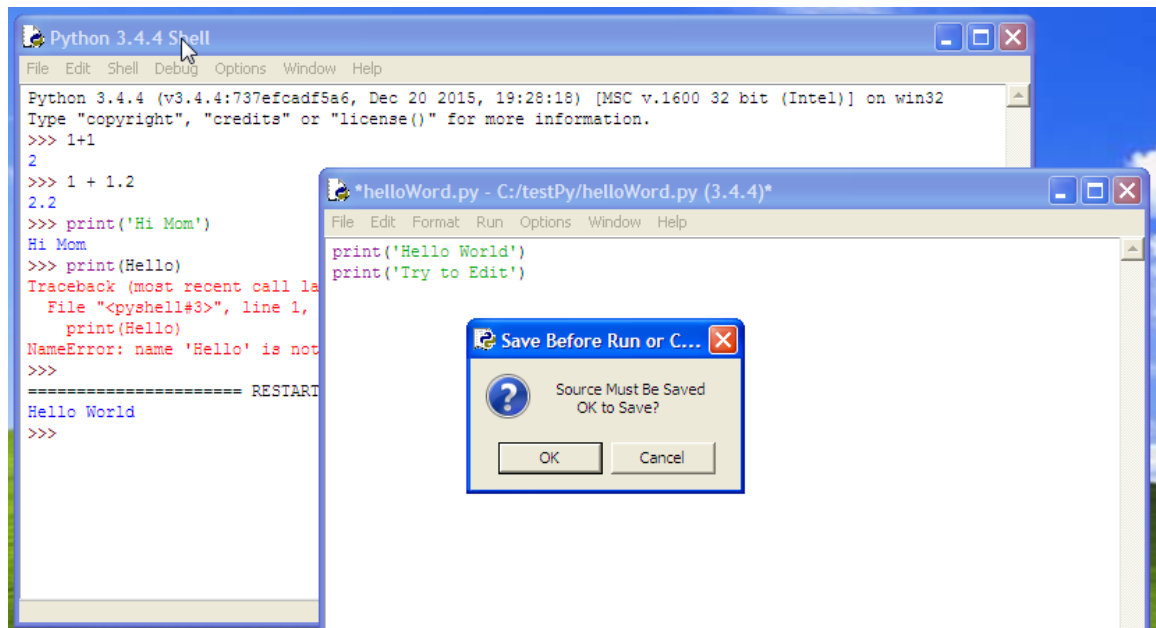
Resulting in new output showing up in your python shell, as shown below



Congratulations, you wrote your first python program.

Section 2.3 Some more programming hints

Everytime you make a change to a file, idle asks you to save the file before you run it. If you try to run it before you save, it will display the following dialog.



At that point, you will then be able to run the program.

After some time, you will come to find two keyboard shortcuts very useful. After you save the file for the first time, the keyboard combination Ctrl-S will resave the file under the present file name. The key F5 will then run the program. So the combo Ctrl-S followed by F5 will help you clean up errors as you go.

Section 2.4 Experiments with a python module

On your console, type the following:

```
>>> import math
>>> math.pi
>>> math.sqrt(25)
>>> 3 + math.pi + 2
```

Checkpoint 1 - Write the output that you see in the box below:

3.141592653589793

5.0

8.141592653589793

ok

Part 3: An Example Program

There are two examples program for this part. You can try to run each of them.

1. `tdemo_clock_demo.py`
2. `tdemo_clock2.py`

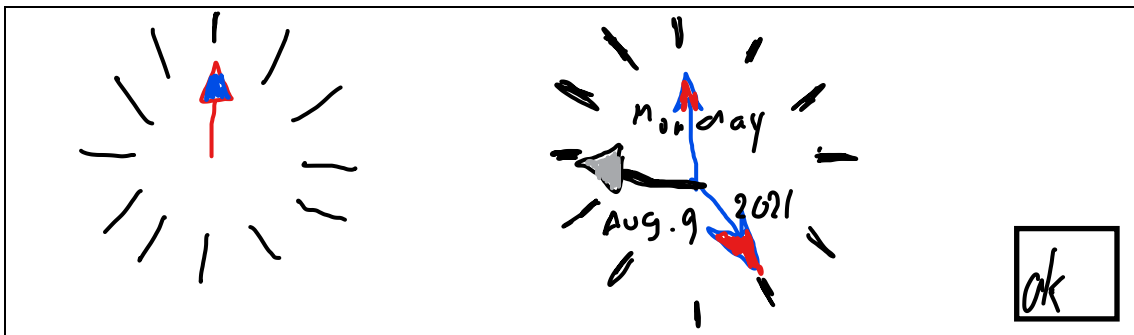
Open and run the program

Start Python if you have not already, and in the Python shell window select menu File⇒Open. Find the example file and open it.

As before, when you open a program an new window appears containing the code of the example. You can look at that code, and when ready run the code by selecting the editing window menu Run⇒Run Module. Show the results to the lab staff.

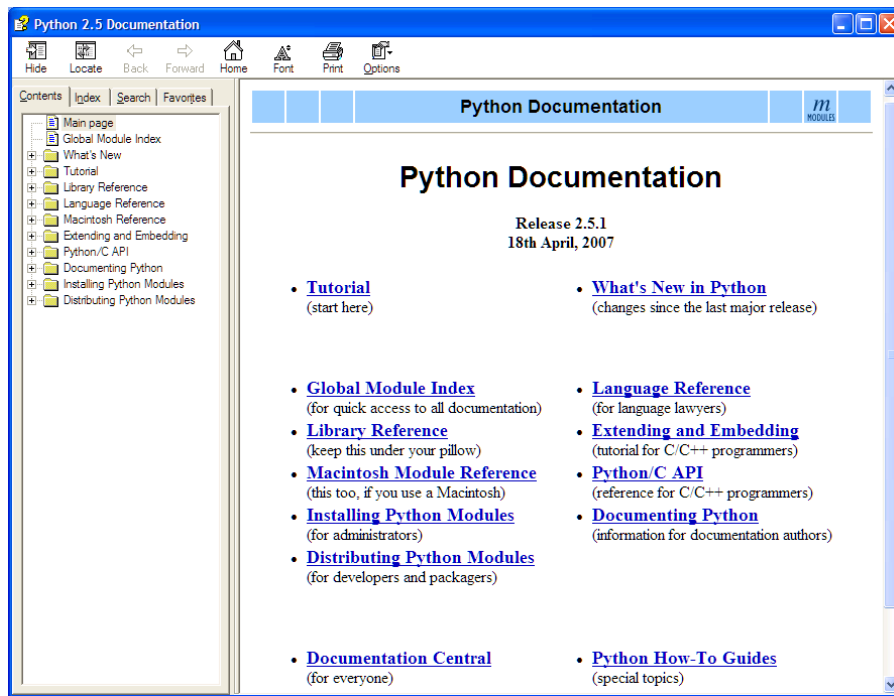
Note that after you run a program, a new file is created. It will have the same name as your original file, but with the extension `.pyc` . For example, when you run `Exempl01.py`, you also create a file `Example01.pyc`. This is a file that is the “compiled” version of your file.

Checkpoint 2 - draw the results of each example in the box below:



Part 4: Getting Help

IDLE/Python installs a host of help documents. In the Python Shell, if you select Help⇒Python_Docs you will get a set of web pages in your default browser (in this case FireFox) as shown below



As you can see, there is a lot of help available! Note that there is a Tutorial available to help you with getting started with Python. There is also a Language Reference for the details of Python and a Library Reference to give you the detail of modules you can import.

Finally, www.python.org has a large amount of help available including Forums. Python folks are usually very helpful to courteous people who have searched the forums before asking a question (to see if the question has been answered many times before).

Part 5: Python Program Development with PyCharm

Prerequisites

Note: Software packages required for this course is provided in the lab. In case you want to install these packages for your own machine, follow the instructions in this section.

Python

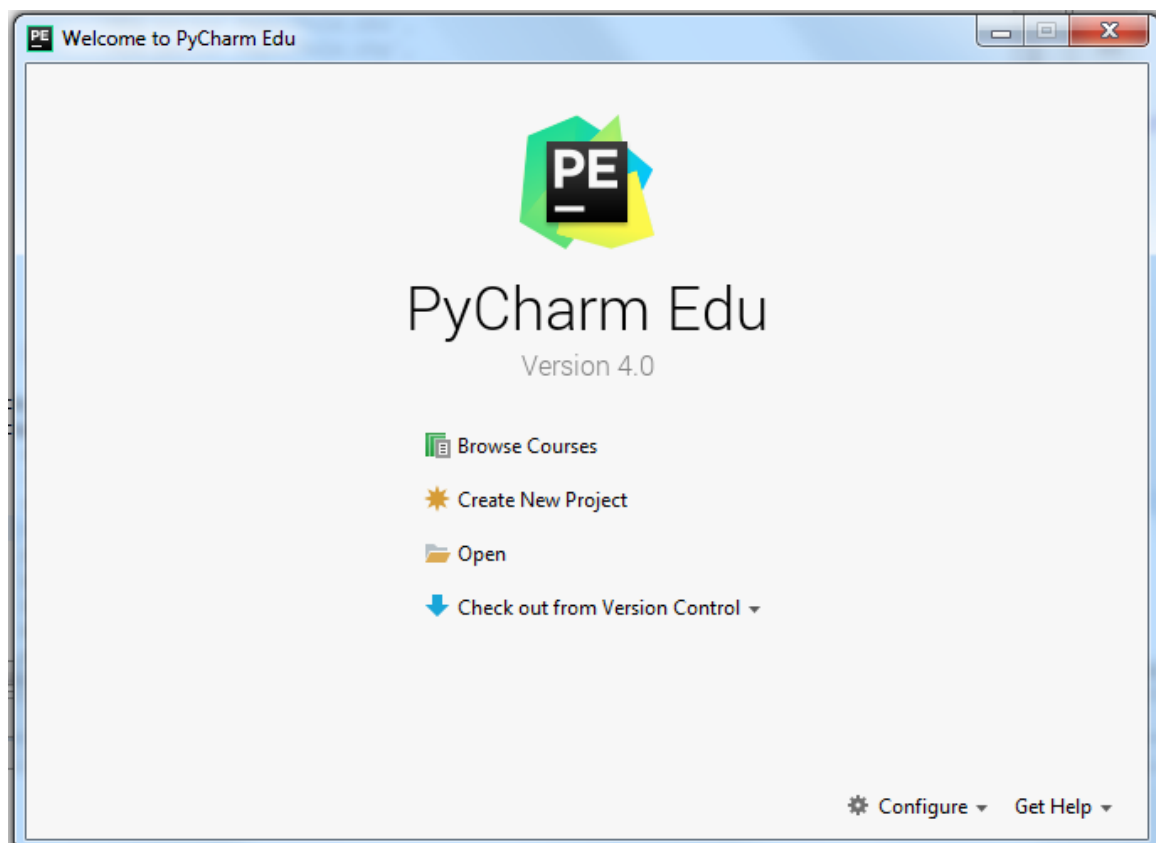
- Download Python (current version is 3.7.4) from <http://www.python.org/>
- For windows, simply use the installer file and follow the setup wizard

PyCharm

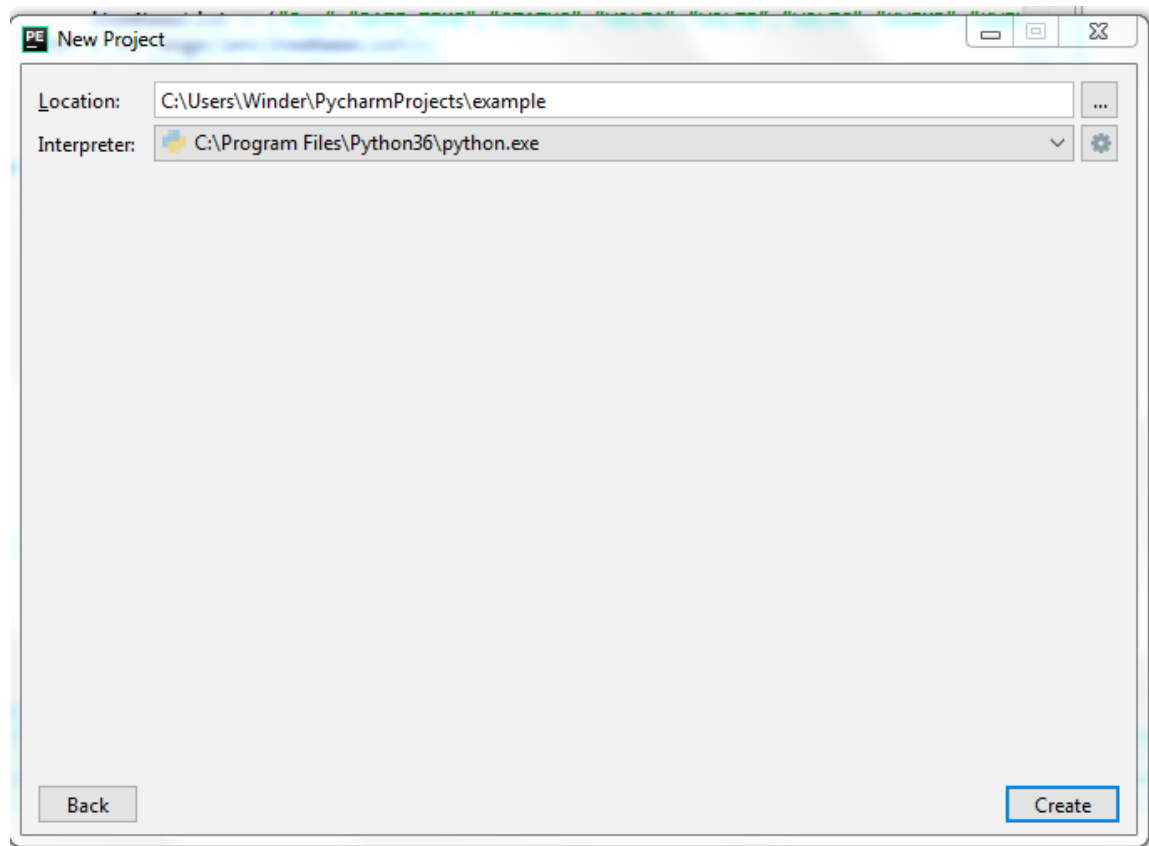
- Download PyCharm from <http://www.jetbrains.com/pycharm/download/>
- Install PyCharm

Configuring Python in PyCharm Environment

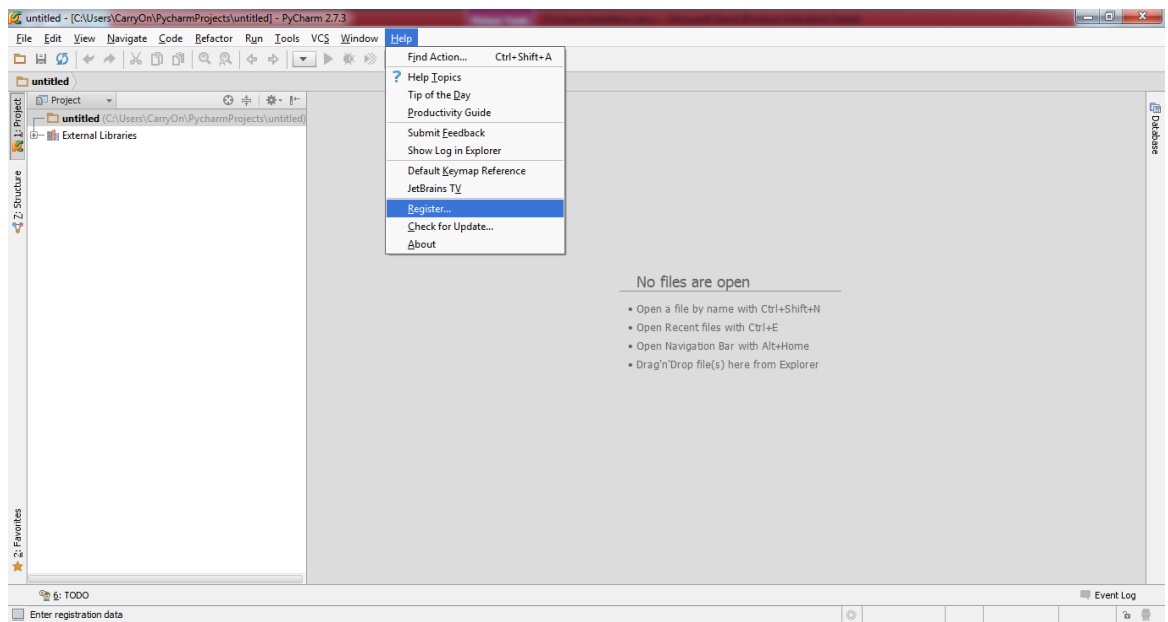
- 1) Open PyCharm and click “Create New Project”



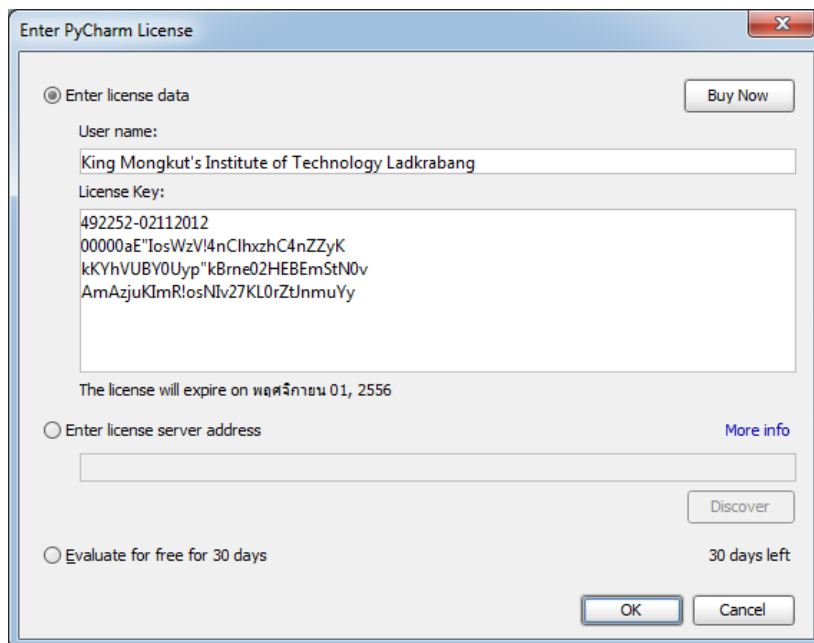
2) Edit Location then Click “Create”



3) Click “Help” and then click “Register”

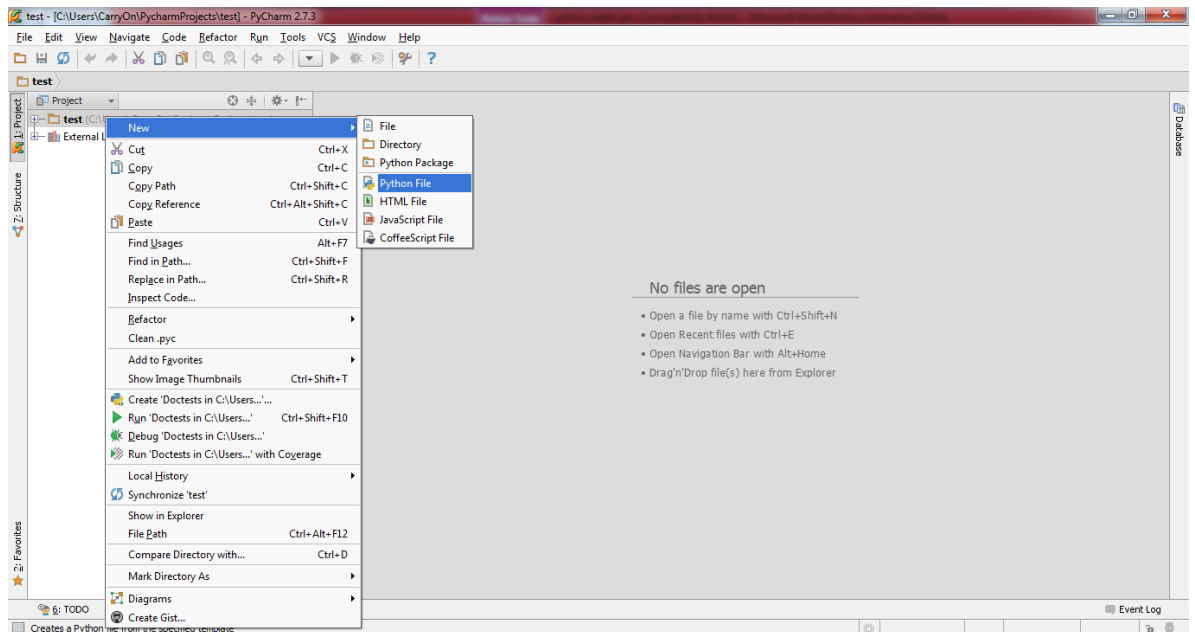


4) Enter your license data



Debugging

- 1) Right-click on your project and select “New->Python File”



- 2) Add your file name and click “ok”

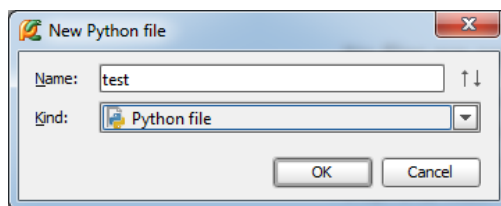


Table 2. Debugging Key bindings

Command	Description
Shift+F10	Use Shift+10 to compile and run your program without debugging.
Shift+F9	Use Shift+F9 to go to the next breakpoint. If no further breakpoint is encountered, then the program will normally run. (Left-click in front of the line of your code to mark a breakpoint.)
F8	F8 will step over the call, e.g. it will call a method / function without entering the associated code.
F7	F7 will go to the caller of the method/ function. So this will leave the current code and go to the calling code.

Checkpoint 5 - Create the following source code in PyCharm, run, and write down the result.

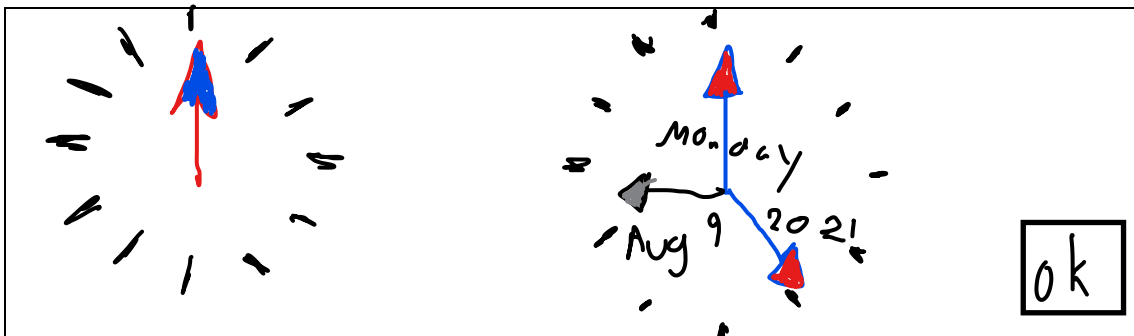
```
def swap(a, b):  
    a, b = b, a  
    return a, b  
  
def isEqual(a, b):  
    return True if a==b else False  
  
print(swap(1,2))  
print(isEqual(3,6))  
print(isEqual(3,3))
```

ok

Checkpoint 6 - Run each of the 2 programs in PyCharm.

1. tdemo_clock_demo.py
2. tdemo_clock2.py

draw the results of each example in the box below:



ok