

Assignment 0: Hello World

EECS 295

Due SOMETIME

Introduction

This exercise will guide you in setting up your working environment for the quarter. By the end you'll be able to write a simple Python program that outputs "Hello World!" to the screen and will be ready for future assignments. First we'll walk through installing Python (our programming language for this course), then installing Visual Studio Code (our text editor for this course) and finally writing our first Python program.

This guide is pretty long with a lot of instructions. I sincerely apologize for that but please make sure everything works for you at all stages along the way so that you don't have to worry about your environment for the rest of the course. And if you have any issues feel free to ask a peer mentor! Okay with that disclaimer let's get started!

Python

First let's install Python!¹ Python has several versions, for this course we're using **Python 3.6**.² This is the most up to date version. There are many people still using an older variant (2.7) but this has been sunsetted and will no longer be supported after 2020. In the interests of learning what will be the future of the language we're teaching 3.6. If you've used 2.7 before the learning curve for 3.6 will be gentle as there aren't too many changes you'll have to worry about. To install follow the below instructions depending on your OS: ³

Installation

Note: If you feel comfortable using a virtual environment manager like **Conda** feel free to use that for this course instead of the following instructions (if you don't know what this means then keep reading for the standard install instructions).

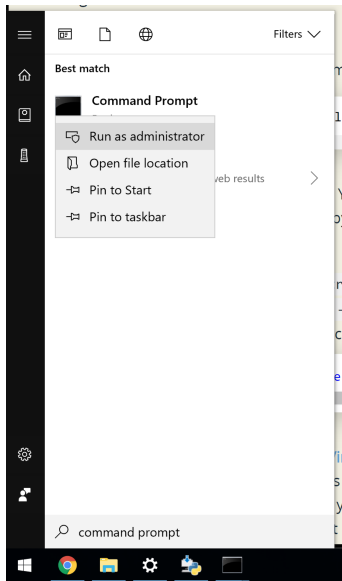
- **macOS:** Python 2.7 comes preinstalled on macOS. Unfortunately, since we want 3.6 we're going to have to install it separately. Our suggested method involves using a package manager called Homebrew. If you have any questions about this method or would like to install in a different way feel free to ask peer mentors about it at office hours. To use our recommended method follow [this guide](#). Do everything up until (not including) the section titled "Pipenv & Virtual Environments".
- **Windows:** Our suggested method involves a package manager called Chocolatey. If you have any questions about this method or would like to install in a different way feel free to ask peer mentors about it at office hours. To use our recommended method follow [this guide](#). Do everything up until (not including) the section titled "Pipenv & Virtual Environments".

¹Python is a widely used programming language older than us. Click [here](#) to view the official documentation.

²Python versions have three numbers, for example Python 3.6.4 but for this course any Python version that starts with 3.6 will work just fine!

³Linux users try the Confirm Installation step and if you get the expected result then you're all good and can keep going with this guide. If not to Sara or one of the peer mentors for help.

Note: For installing Chocolatey I recommend using Command Prompt over Powershell since it involves one less step. Simply click on the “Start” button, type “command prompt” and right click on the search result labeled Command Prompt then choose ‘Run as administrator” in the resulting menu. See below for an example photo:



- **Linux:** There are too many Linux variants to attempt to cover them all but general advice is use a package manager and go to office hours with any questions or difficulties installing.

Confirm Installation

To confirm everything is working right open up your command line (Command Prompt on Windows, Terminal on macOS/Linux) and type `python` then hit Enter. You should see something like the following (this is an example run on Windows though it should look similar on every platform, really the only change is it will list OS specific architecture in the right hand corner, in this case it's `win32`, on recent macOS models it's `darwin`):

```
C:\Users\nathan>python
Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:54:40) [MSC v.1900 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> _
```

If you see something similar and as long as your Python version says 3.6.<anything here> you're all good! Go ahead and close your command line and move on to the next step.⁴ If not, come to see Sara or a peer mentor for help.

Note: If you see 2.7.<anything here> instead, try typing `python3` then hitting enter and seeing if that gives you something like the image above with 3.6.<anything here> for the version. If it does then feel free to ask a peer mentor how to make `python` launch Python 3.6 by default but be aware that you can continue

⁴What we just started by typing `python` and hitting enter is called the Python Interpreter. This is an interactive environment like the REPL we had in 111 where you can write Python code and immediately see the result. For an example type `5+5` and hit enter. The interpreter will display 10 and then wait for your next instruction. Feel free to play around in the shell further if you'd like. To exit the shell and get back to your regular command line type `exit` and hit enter for macOS, on Windows type `exit()` and hit enter.

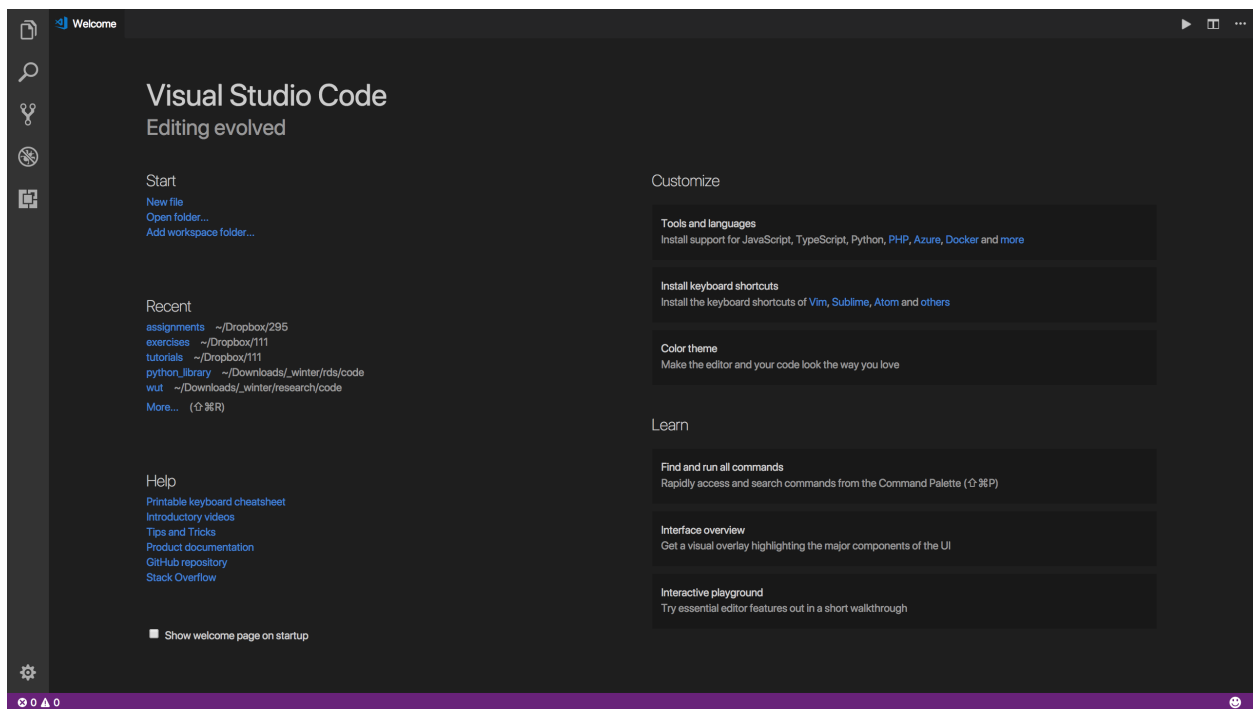
this guide swapping in `python3` anywhere you see `python` and `pip3` anywhere you see `pip`. If neither of these work try restarting your computer and running again, then go to a peer mentor for assistance if that doesn't change anything.

Our Text Editor - Visual Studio Code

Next we're going to install the text editor we're using for this course. There are many different text editors out there that you're welcome to explore but for this course we're going to be using Visual Studio Code (referred to as VSCode from here on).

Installation

Go to [this page](#) and download the appropriate installer. Once the installer downloads follow the prompts to install (simply clicking next on each screen and accepting the defaults will work just fine). Once it's installed open it up and you should see a screen like this:



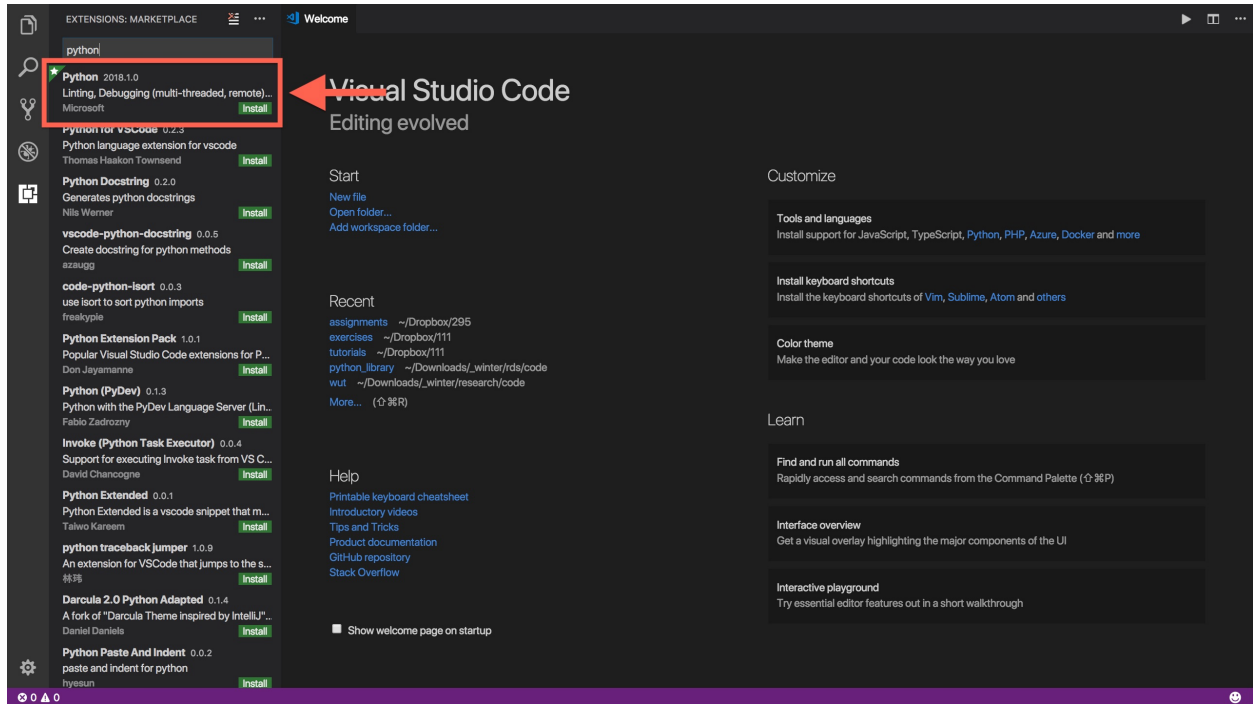
VSCode has almost unlimited configuration and customization options along with numerous capabilities that we won't be using in this class. Feel free to explore these as you'd like but for this guide we'll focus on a small subset of features.

Add Python Support

First things first let's add Python support. To do so we need to install an extension and set VSCode's Python path. Almost everything I'm about to walk through can be done with keyboard shortcuts (which I'll try to list) if you like using them but I'll be prioritizing menu items as these are more universal across OSes. Note: if you see an error "Unknown configuration setting" while working through this section first try restarting VSCode before asking Sara or a TA for assistance.

Extension

First click on View->Extensions (macOS/Win: **Cmd/Ctrl+Shift+x** or click on the square icon at the bottom of the leftmost sidebar) and then type “Python” in the search box. We want to install the extension just titled “Python” which is made by Microsoft:



Click install beside that extension then click the reload button that shows up once installed. Perfect we now have Python support!

Python Path

Open a new file by clicking File->New File (or **Cmd/Ctrl + N**). Next, set the language for this file to Python by clicking View->Command Palette (or **Cmd/Ctrl + Shift + P**) and type “language”. From the list of results that pops up choose “Change Language Mode”. Then type “Python” in the resulting search box and select Python from the list. Now look in the lower left hand corner (see below photo for an example) and confirm that it says **Python 3.6.<anything-else>**. If it does skip ahead to the next section titled **Extension**. If you don’t see anything try restarting VSCode. If it shows **Python 2.<anything-else>** stick around and we’ll fix it. **Note:** for Conda users you’ll want to set this to the path to the **python** instance in the **bin** folder of your virtual environment.



To make it so that you get Python 3.6.<anything-else> open up VSCode settings by clicking on the cog in the lower left then “Settings” or find the Settings option in the menu (Code->Preferences->Settings on macOS, File->Preferences->Settings on Windows) or use the keyboard shortcut Cmd/Ctrl + ,. This will open two panes, in the right one (where it says “User Settings”) we want to add the following code:

```
{  
  "python.pythonPath": "<path_to_python_installation>"  
}
```

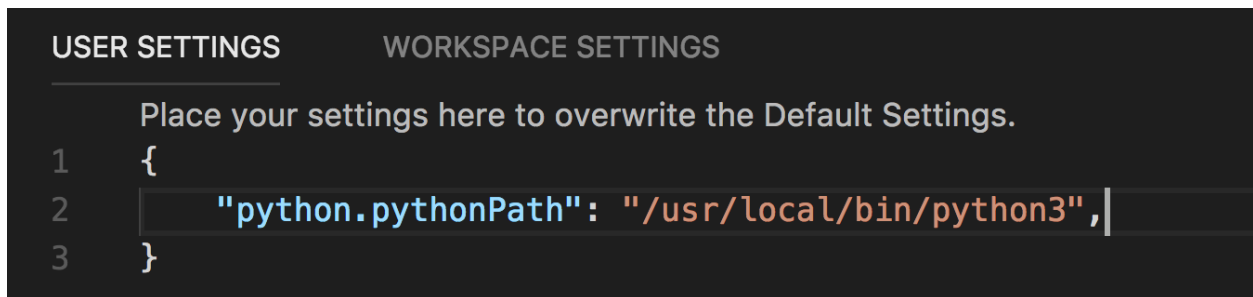
You need to replace <path_to_python_installation> with the actual location of the Python 3 installation on your computer. On macOS for example this is the path: /usr/local/bin/python3. Note: on Windows this path would normally use backslashes but here you should use either forward slashes or double backslashes, e.g. C:/Users/nathan/<rest of path> or C:\\Users\\nathan\\<rest of path> (if you’re curious why ask one of the TAs about escape characters). The easiest way to find where Python 3 is installed on your computer is to run one of the following commands (depending on your OS) in terminal or command prompt:

```
where python # Windows  
which python # macOS/Linux
```

Here’s an example of the output (run on macOS, note I’m running which python3 because I have multiple Python installations, you should not type the 3 unless back in the confirm installation section running python didn’t show 3.6.<anything-else>):

```
/Users/nathan >> which python3  
/usr/local/bin/python3  
/Users/nathan >> _
```

Now write the path that which/where python gives you in place of "<path_to_python_installation>" in the settings pane. At the end you should have something that looks like this:



Confirm that after restarting VSCode and selecting the language Python as before you now see 3.6.<anything> in the lower left hand corner. If not go to mentor hours. Whew, okay let’s move on to setting up the rest of our Python support!

Add Linting

Linting is a form of statically checking programs to make sure they are correct before running them. The linter we install will help by checking types (Python 3’s support for type hints is another reason we’re using it) and preventing silly mistakes.

Setting up this linter requires two steps. First, run the following command in the terminal (or Command Prompt):

```
pip install mypy
```

You should see something like this (run on macOS):

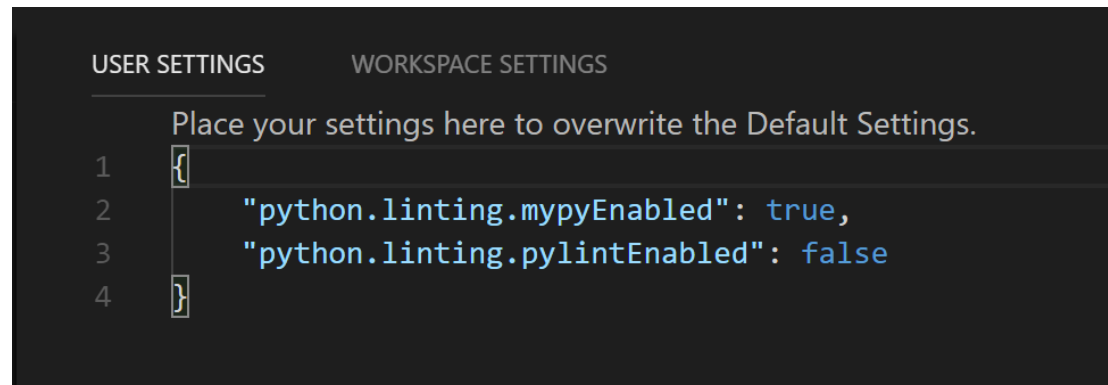
```
/Users/nathan >> pip install mypy
Collecting mypy
  Downloading mypy-0.580-py3-none-any.whl (1.2MB)
    100% |#####| 1.2MB 1.2MB/s
Requirement already satisfied: typed-ast<1.2.0,>=1.1.0 in /usr/local/lib/python3.6/site-packages (from mypy)
Installing collected packages: mypy
Successfully installed mypy-0.580
```

This command will install our linter (if you're curious `pip` is a package manager bundled with Python 3 for installing Python libraries that we'll discuss more later in the course). Next, we need to configure our linter.

To configure our linter we need to open up VSCode's settings. Click on the cog in the lower left then "Settings" or find the Settings option in the menu (Code->Preferences->Settings on macOS, File->Preferences->Settings on Windows) or use the keyboard shortcut `Cmd/Ctrl + ,`. This will open two panes, in the right one (where it says "User Settings") we want to add the following code (**Note:** the whole settings is a single JSON dictionary, so if you added the python path previously the settings below would be two more lines inside the outer curly brackets, not inside a nested pair of brackets):

```
{
    "python.linting.mypyEnabled": true,
    "python.linting.pylintEnabled": false
}
```

After adding these our settings file will look something like this (potentially with an extra setting from modifying our path if you had to do that earlier):



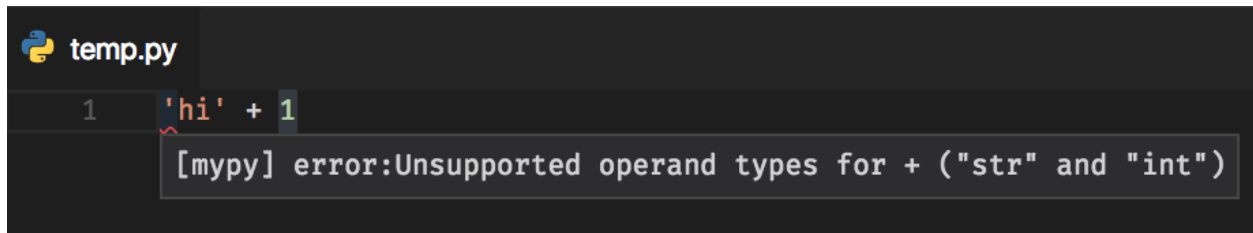
The screenshot shows the VS Code 'USER SETTINGS' pane. At the top, it says 'Place your settings here to overwrite the Default Settings.' Below this, a JSON configuration is entered:

```
1 {
2     "python.linting.mypyEnabled": true,
3     "python.linting.pylintEnabled": false
4 }
```

Let's test this out. Open a file and switch the language to Python by clicking View->Command Palette (or `Cmd/Ctrl + Shift + P`) and type "language". From the list of results that pops up choose "Change Language Mode". Then type "Python" in the resulting search box and select Python from the list. Once we're in Python type the following in your new file:

```
'hi' + 1
```

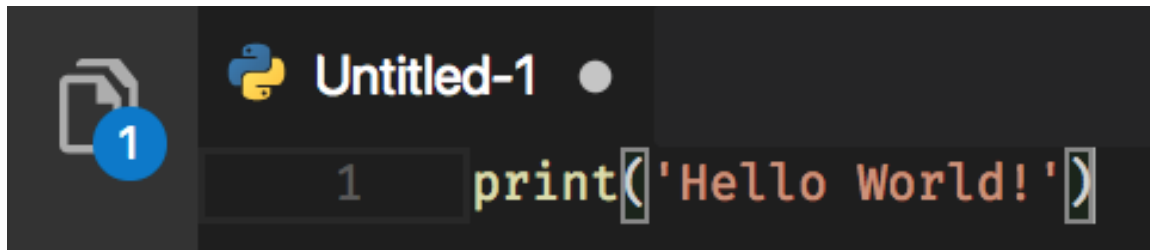
Save the file with any name and you should notice some weird red squiggles under the 'hi' (if you don't see these try restarting VSCode). Hover over this to see the message screenshotted below:



This message is telling us we can't add a `str` (Python's shorthand for `string`) and an `int` (an integer). This sort of type checking will come in super helpful as we continue to write programs later in this class. Okay now with everything set up let's write our first program and run it!

Working in Python

In this class we'll be writing Python code in VSCode and running our programs from either VSCode or the Command Line. First let's write a simple program. Open a new file by going to File->New File and set the language to Python. We'll simply write `print('Hello World!')` on the first line of our file. Like below:



Now save this file (File->Save or `Cmd/Ctrl + S`) calling it whatever you want making sure to have a `.py` extension at the end of it (VSCode should insert this extension for us since we changed the language mode to Python earlier). We've written a Python program, now let's run it!

VSCode

Setting Our Launch Configuration

First we need to set our launch configuration. Open up settings again (see the [Add Linting](#) section) and add the following code:

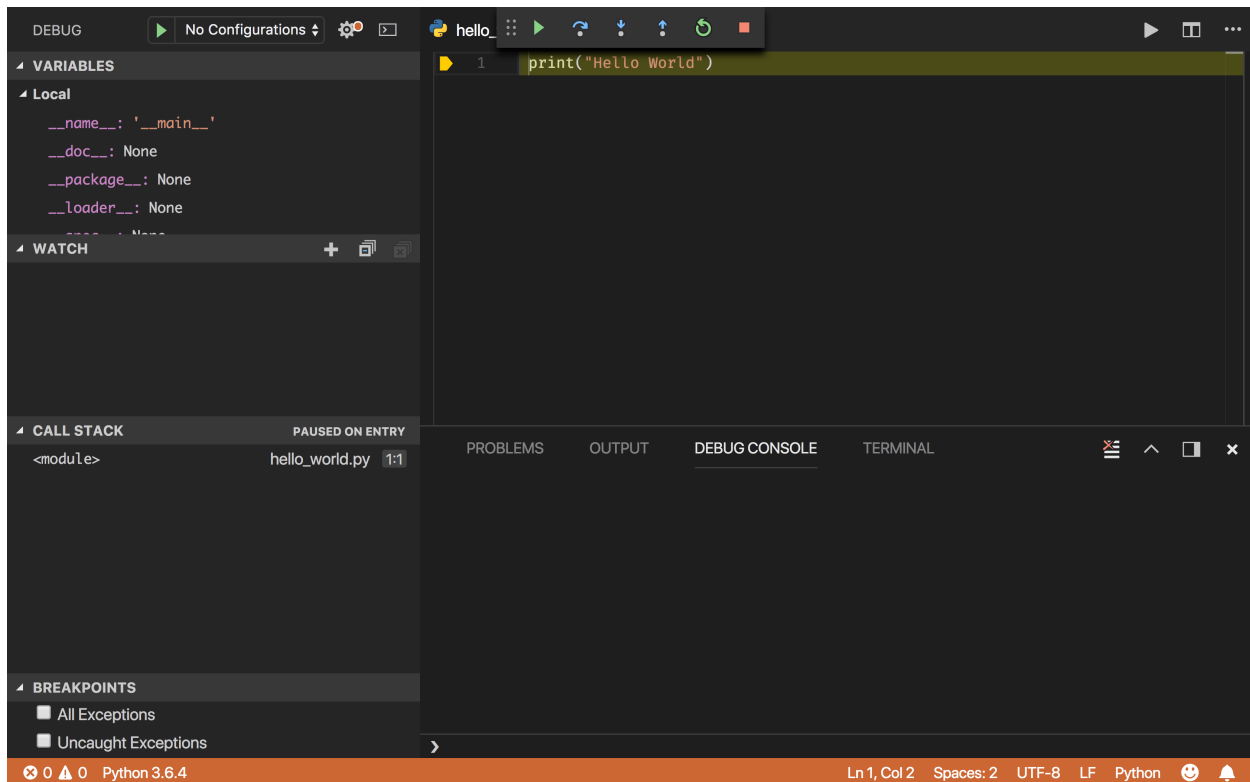
```
"launch": {
  "configurations": [
    {
      "name": "Global Python",
      "type": "python",
      "request": "launch",
      "program": "${file}",
      "stopOnEntry": true,
      "internalConsoleOptions": "openOnSessionStart"
    }
  ]
}
```

If you had to add your Python path earlier your settings should look like the following, if not then they should look the same minus the `python.pythonPath` line:

```
USER SETTINGS      WORKSPACE SETTINGS
Place your settings here to overwrite the Default Settings.
1  {
2    // launch
3    "launch": {
4      "configurations": [
5        {
6          "name": "Global Python",
7          "type": "python",
8          "request": "launch",
9          "program": "${file}",
10         "stopOnEntry": true,
11         "internalConsoleOptions": "openOnSessionStart"
12       }
13     ]
14   },
15
16   // python path - only if you added this in the `Python Path` section
17   "python.pythonPath": "/usr/local/bin/python3",
18
19   // linting
20   "python.linting.mypyEnabled": true,
21   "python.linting.pylintEnabled": false
22 }
```

Running Code

Okay now let's run our code! Go to View->Debug (or click on the icon with the crossed out bug in the left sidebar). In the resulting view click the dropdown next to the green play button in the top left and from the menu select **Global Python** (the name of the launch configuration we just added in settings). Now click the green play button and you should see something like this (**Note:** if you see a dropdown with multiple options such as **Python Experimental** choose the option that just says **Python**):



Now click the green play button in the center at the top of your screen to view the output of your program in the Debug Console on the bottom of the screen. Don't worry about what everything in this view means, we'll get to it all over the first couple weeks of class. Whew, you're officially done with the first assignment! ⁵ There isn't anything to turn in so as long as you've successfully completed everything in this guide you're all good! The rest of this guide is optional if you're interested in playing around more with python or VSCode. Feel free to do as much or little of it as you'd like.

Just For Fun

Command Line: Integrated Terminal

So we just ran Python using the VSCode debugger now let's try running it through the command line! VSCode actually has an integrated command line that we can open by going to View->Integrated Terminal. Note: for Windows users this will likely open to Powershell, change this by clicking View->Command Palette (or `Cmd/Ctrl + Shift + P`) then type "select default shell", choose the "Terminal: Select default shell" option and in the resulting menu select "Command Prompt".

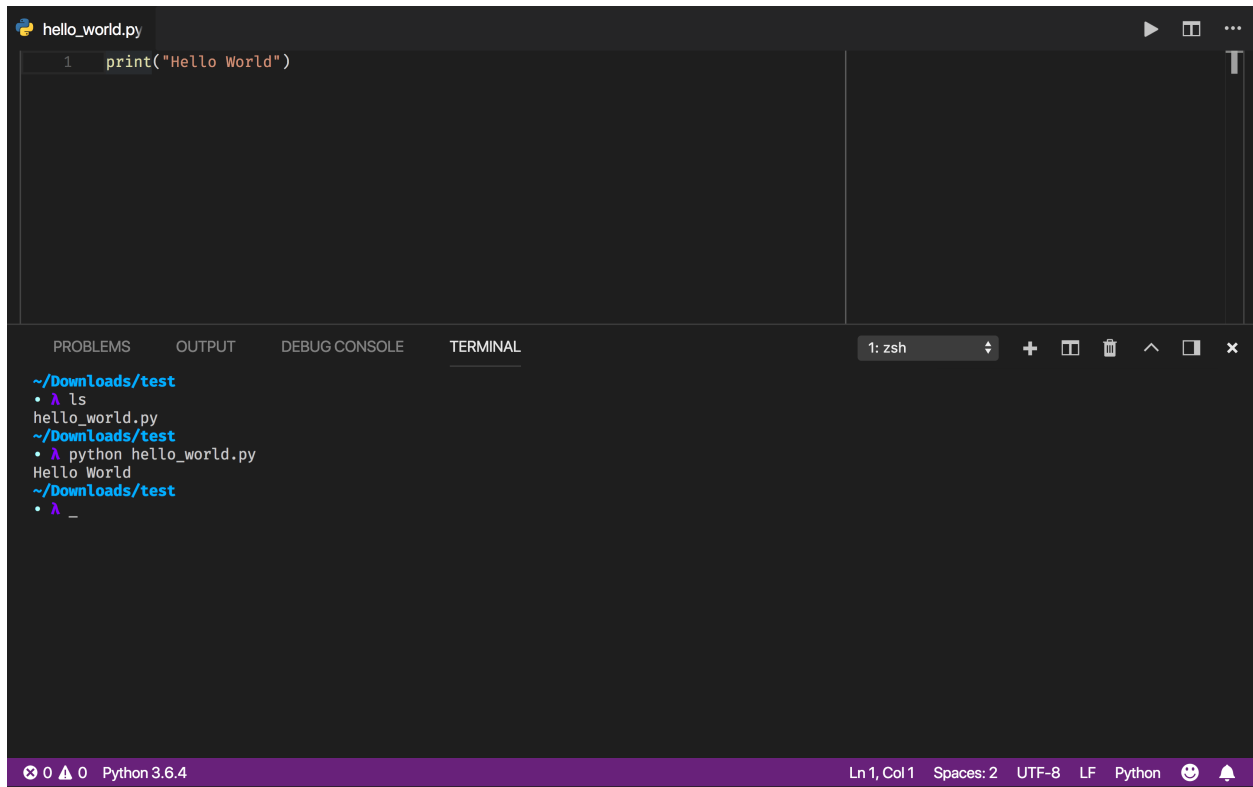
For running Python files from the command line we need to be in the same folder as the file we want to run. To do so we need to be able to navigate the command line. Sara will cover how to navigate the command line during the first lecture and for more help see the following guides for navigating the command line (or come to peer mentor hours):

- [Windows](#)
- [macOS and Linux](#)

Once we're in the folder that our Python file is in simply type `python <filename>`. Do this with the file you've just written, so if you named the file `hello_world.py` you'd write `python hello_world.py` and you

⁵[Hip hip hooray!](#)

should see ‘Hello World!’ outputted to the screen like below (note that you want to write the full filename with the “.py” extension):

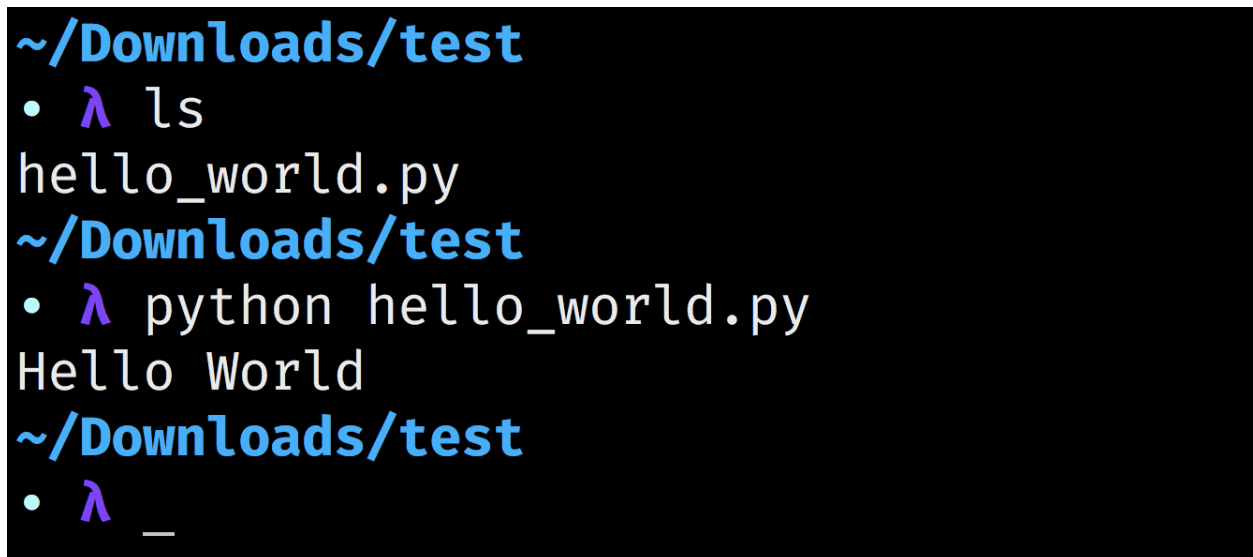


```
hello_world.py
1 print("Hello World")

~/.Downloads/test
• λ ls
hello_world.py
~/.Downloads/test
• λ python hello_world.py
Hello World
~/.Downloads/test
• λ _
```

Command Line Part II: This Time It’s External

We’ve now run python code from the integrated terminal but we can also run code from an external terminal like we did to confirm installation. Open up your terminal (Command Prompt for Windows) like we did before, navigate to the folder just like we did for the integrated terminal and once again type `python <filename>` to see ‘Hello World!’ like below:

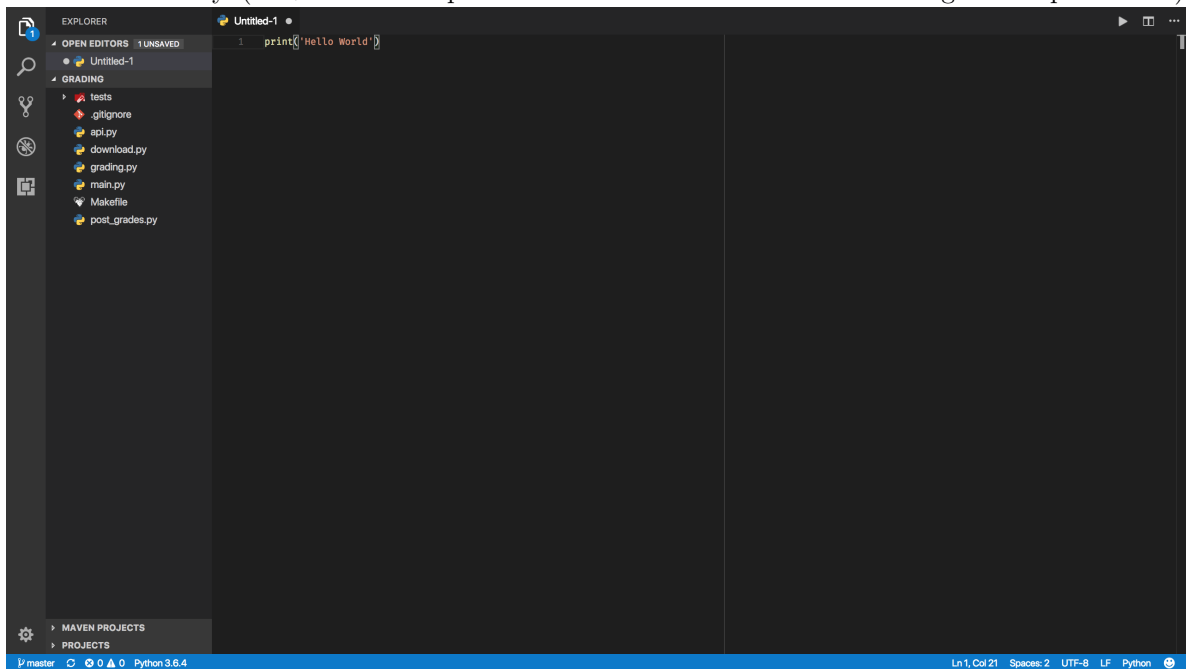


```
~/.Downloads/test
• λ ls
hello_world.py
~/.Downloads/test
• λ python hello_world.py
Hello World
~/.Downloads/test
• λ _
```

VSCode Tips

Here's a few tips and things to explore in VSCode:

1. Go to View->Explorer (or click on the top icon in the left sidebar that looks like overlapping documents) to see a list of the current open files and a button to open a folder. Click on that button and choose a random folder on your computer to see that folder and its contents and be able to navigate it and open files from it within VSCode. This has the added benefit that after opening a folder in VSCode the integrated terminal will automatically open to the folder you opened for easier running of your code. The below screenshot shows my VSCode with a folder called **grading** open. I can then see and open any of the subfolders and files in this folder more easily (Cmd/Ctrl + P opens a file searcher that searches through the open folder).



2. Go to File->Preferences->Color Theme if you'd like to adjust the colors of the editor. More themes can be found [here](#).
3. Go to File->Preferences->File Icon Theme if you'd like icons in tabs or next to files/folders in the Explorer view. More icon themes can be found [here](#).
4. Go to File->Preferences->Keyboard Shortcuts or File->Preferences->Settings to play around with the default settings or keybindings.