National University of Singapore
School of Computing
CS1010S: Programming Methodology
Semester I, 2022/2023

Mission 0
Setting Up Python

Release date: 4th August 2022
Due: 16th August 2022, 23:59

Required Files

- mission00-template.py

The objective of this mission is to guide you in the installation and setting up of the programming tool, called IDLE, that you will be using for the rest of the semester. In Part 2, we include a simple exercise for you to familiarize yourself with the basics of Python, as well as learn how to submit homework on Coursemology.

This mission consists of only **one** task.

Part 1: Installing Python 3.10.6 and required packages

Before you start on your quest to master the Python programming language, you have to install the necessary tools. Please follow the following instructions to set up your programming environment for the class.

PIL is required to render images that will be used in the later missions. We will be using PILLOW, which is a modern replacement for PIL. The NumPy package are also required in the later missions.

Note: The highest priority package to install is **PILLOW**, as we will be using it over the next few weeks.

Now, please follow the following instructions carefully:

(*For Windows*) (*For Mac*) (*For Linux*)

Troubleshooting

You can contact the teaching staff or check the Coursemology forum if you face any difficulty during the setup/installation process.

Please provide the following information to help us quickly identify your problem:

1. The operating system and version that you are using. MacOS, Windows 10, 32-bit or 64-bit, etc.
2. Which step in the instructions given below did it fail or produce an error?
3. Take a screen shot of the failure or error.

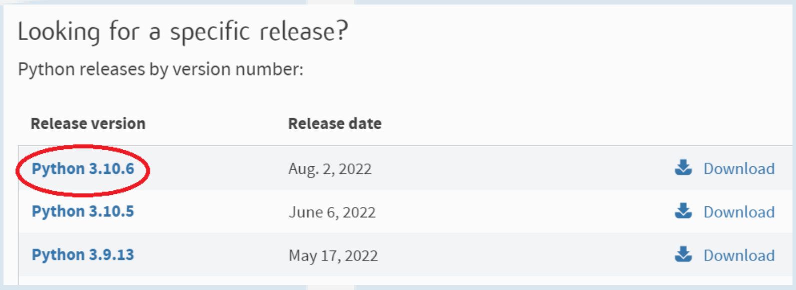
Download Python

IMPORTANT: You will need to be **connected to the Internet** for all the installation steps as the required packages will have to be downloaded.

Windows Users:

Step 1: Install Python for Windows

Head over to `https://www.python.org/downloads/` and you should see something similar to this:

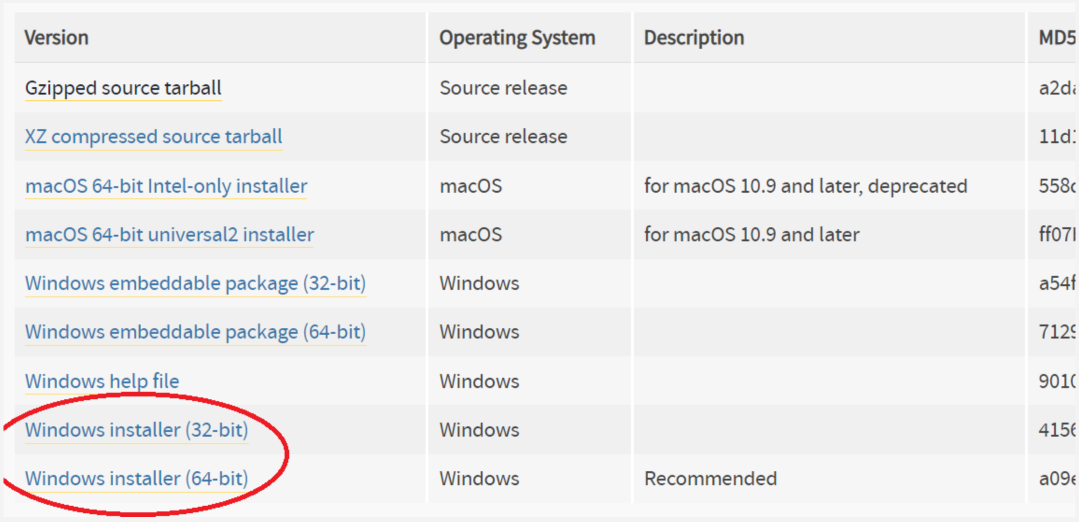


Looking for a specific release?

Python releases by version number:

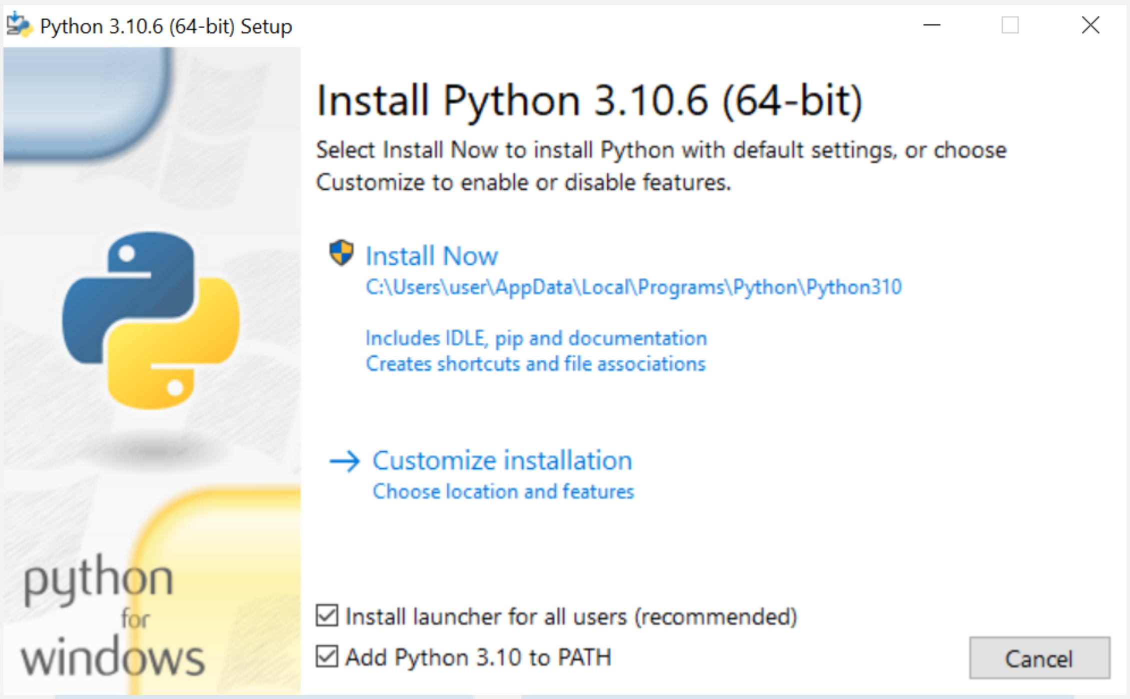
Release version	Release date	
Python 3.10.6	Aug. 2, 2022	Download
Python 3.10.5	June 6, 2022	Download
Python 3.9.13	May 17, 2022	Download

Choose **Python 3.10.6** and then choose **Windows installer (x-bit)** (depends whether you are on 32-bit or 64-bit).



Version	Operating System	Description	MD5
Gzipped source tarball	Source release		a2d
XZ compressed source tarball	Source release		11d
macOS 64-bit Intel-only installer	macOS	for macOS 10.9 and later, deprecated	558
macOS 64-bit universal2 installer	macOS	for macOS 10.9 and later	ff07
Windows embeddable package (32-bit)	Windows		a54
Windows embeddable package (64-bit)	Windows		712
Windows help file	Windows		901
Windows installer (32-bit)	Windows		415
Windows installer (64-bit)	Windows	Recommended	a09

The **executable file** should be inside your downloads folder now. Upon running it, you should see something like this:



Python 3.10.6 (64-bit) Setup

Install Python 3.10.6 (64-bit)

Select Install Now to install Python with default settings, or choose Customize to enable or disable features.

Install Now
C:\Users\user\AppData\Local\Programs\Python\Python310

Includes IDLE, pip and documentation
Creates shortcuts and file associations

Customize installation
Choose location and features

☑ Install launcher for all users (recommended)
☑ Add Python 3.10 to PATH

Important: You want to be sure to check the box that says **Add Python 3.10 to PATH** as shown to ensure that the interpreter will be placed in your execution path.

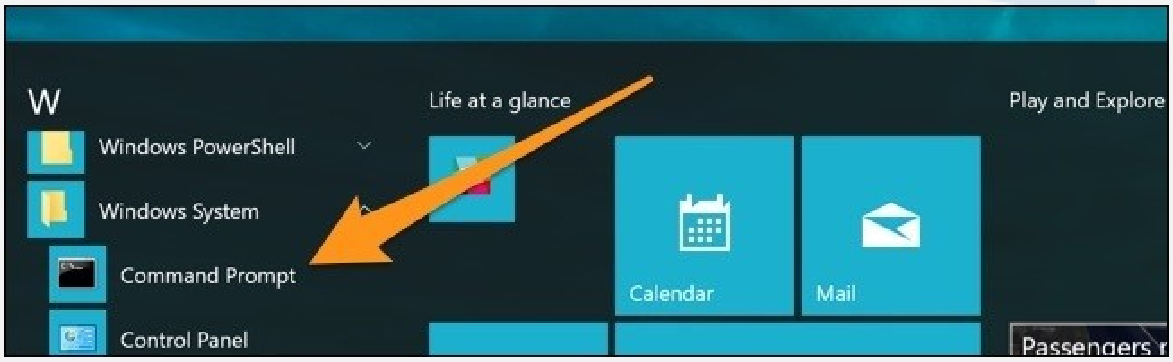
You do not need to install for all users if you have no admin access.

Click **Install Now** and it will complete in a few minutes. You should see a new program **IDLE** in your programs menu.

Note: While you may install the 64-bit version if you are certain that you are running 64-bit Windows (most modern computers are), the 32-bit version is sufficient for our class.

Step 2: Installing the packages

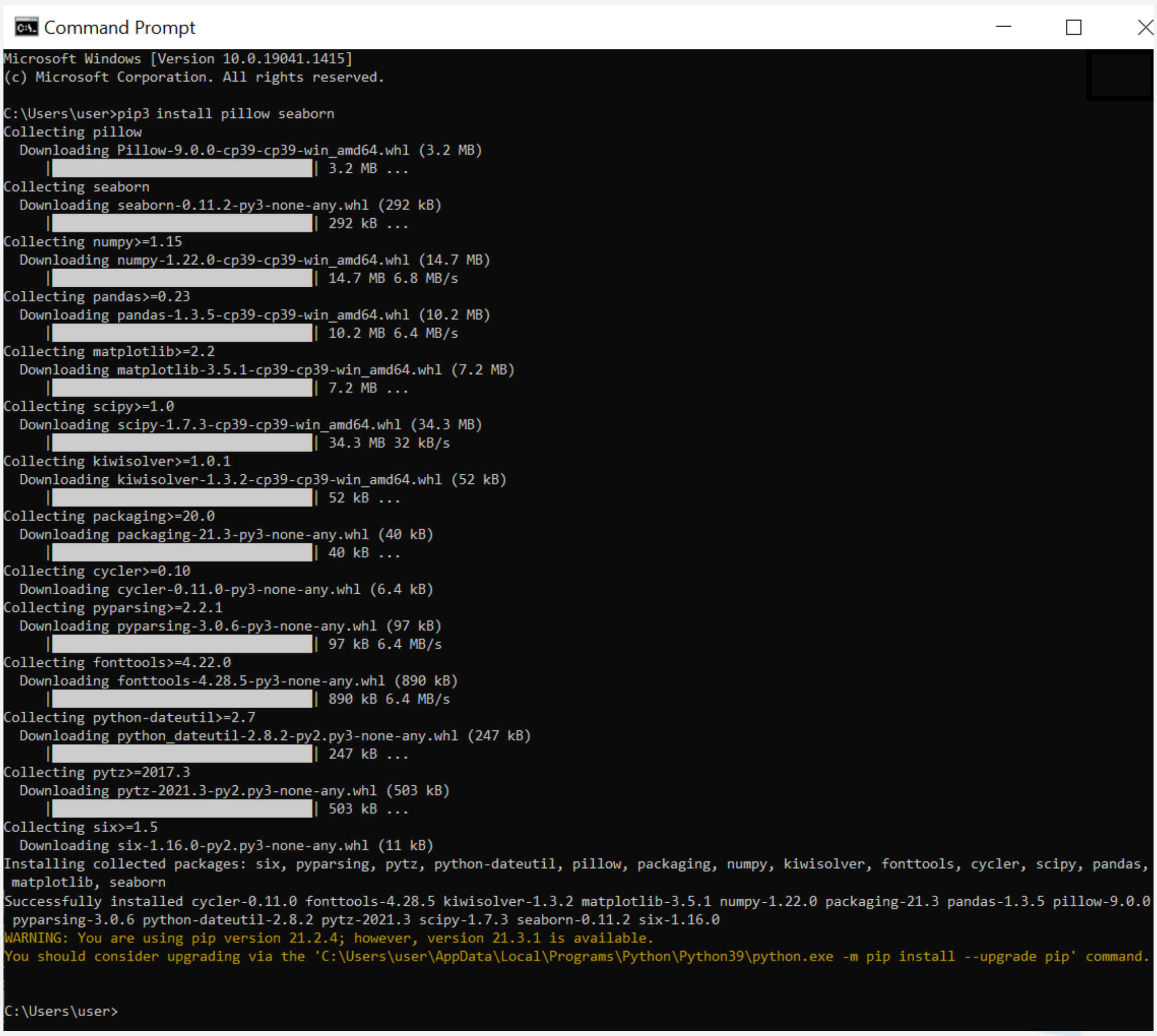
Open the **Command Prompt** from your Start menu.



Enter the following command as one line and press enter:

```
pip3 install pillow seaborn cocos2d --no-cache-dir
```

This will install the `pillow` and `seaborn` packages and all other packages that are not stated but they depend on. If successful, you should see a line saying “Successfully installed...” like this:



```
Microsoft Windows [Version 10.0.19041.1415]
(c) Microsoft Corporation. All rights reserved.

C:\Users\user>pip3 install pillow seaborn
Collecting pillow
  Downloading Pillow-9.0.0-cp39-cp39-win_amd64.whl (3.2 MB)
     |████████████████████████████████| 3.2 MB ...
Collecting seaborn
  Downloading seaborn-0.11.2-py3-none-any.whl (292 kB)
     |████████████████████████████████| 292 kB ...
Collecting numpy>=1.15
  Downloading numpy-1.22.0-cp39-cp39-win_amd64.whl (14.7 MB)
     |████████████████████████████████| 14.7 MB 6.8 MB/s
Collecting pandas>=0.23
  Downloading pandas-1.3.5-cp39-cp39-win_amd64.whl (10.2 MB)
     |████████████████████████████████| 10.2 MB 6.4 MB/s
Collecting matplotlib>=2.2
  Downloading matplotlib-3.5.1-cp39-cp39-win_amd64.whl (7.2 MB)
     |████████████████████████████████| 7.2 MB ...
Collecting scipy>=1.0
  Downloading scipy-1.7.3-cp39-cp39-win_amd64.whl (34.3 MB)
     |████████████████████████████████| 34.3 MB 32 kB/s
Collecting kiwisolver>=1.0.1
  Downloading kiwisolver-1.3.2-cp39-cp39-win_amd64.whl (52 kB)
     |████████████████████████████████| 52 kB ...
Collecting packaging>=20.0
  Downloading packaging-21.3-py3-none-any.whl (40 kB)
     |████████████████████████████████| 40 kB ...
Collecting cycler>=0.10
  Downloading cycler-0.11.0-py3-none-any.whl (6.4 kB)
Collecting pyparsing>=2.2.1
  Downloading pyparsing-3.0.6-py3-none-any.whl (97 kB)
     |████████████████████████████████| 97 kB 6.4 MB/s
Collecting fonttools>=4.22.0
  Downloading fonttools-4.28.5-py3-none-any.whl (890 kB)
     |████████████████████████████████| 890 kB 6.4 MB/s
Collecting python-dateutil>=2.7
  Downloading python_dateutil-2.8.2-py2.py3-none-any.whl (247 kB)
     |████████████████████████████████| 247 kB ...
Collecting pytz>=2017.3
  Downloading pytz-2021.3-py2.py3-none-any.whl (503 kB)
     |████████████████████████████████| 503 kB ...
Collecting six>=1.5
  Downloading six-1.16.0-py2.py3-none-any.whl (11 kB)
Installing collected packages: six, pyparsing, pytz, python-dateutil, pillow, packaging, numpy, kiwisolver, fonttools, cycler, scipy, pandas,
 matplotlib, seaborn
Successfully installed cycler-0.11.0 fonttools-4.28.5 kiwisolver-1.3.2 matplotlib-3.5.1 numpy-1.22.0 packaging-21.3 pandas-1.3.5 pillow-9.0.0
 pyparsing-3.0.6 python-dateutil-2.8.2 pytz-2021.3 scipy-1.7.3 seaborn-0.11.2 six-1.16.0
WARNING: You are using pip version 21.2.4; however, version 21.3.1 is available.
You should consider upgrading via the 'C:\Users\user\AppData\Local\Programs\Python\Python39\python.exe -m pip install --upgrade pip' command.

C:\Users\user>
```

You may ignore the yellow text if any. However, if you see a bunch of red text, the packages might not be installed properly. Please take a look at the section next page. As a last resort, take a screenshot and check the Coursemology forums for assistance.

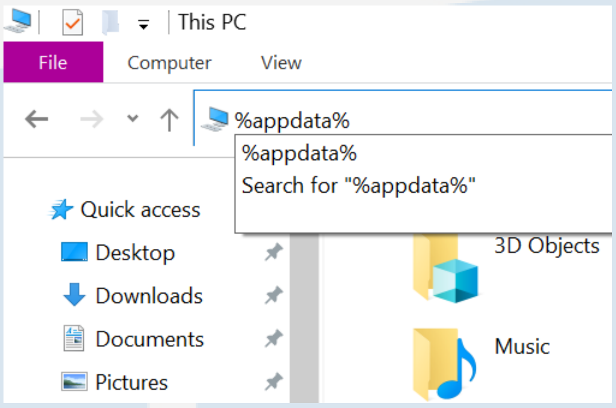
Otherwise, you may skip the extra troubleshooting part and proceed to *the next part*.

More Troubleshooting

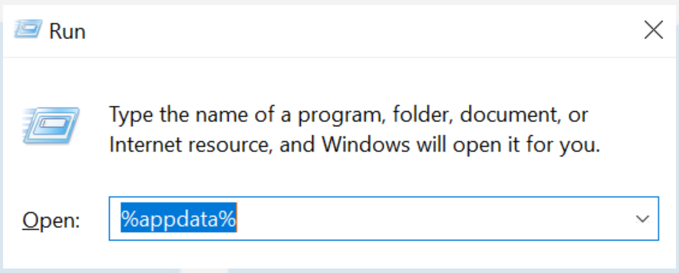
Here are some commonly found issues that might help you. **Please read them carefully as your problem might be included below.**

1. **Checking if Python is installed**
   Your Python path should be similar to
   `C:\Users\user\AppData\Local\Programs\Python\Python310`.
   Here's how to navigate you there.

   - Method 1: Go to Windows Explorer, and type `%appdata%` on the current directory.

   - Method 2: Key in `Win+R`, and then type `%appdata%`.

   
   Run

   Type the name of a program, folder, document, or Internet resource, and Windows will open it for you.

   Open: %appdata%

   Both methods will take you to the `C:\Users\user\AppData\Roaming` directory, the closest we can get from the destination path. Navigate yourself to the destination path from there, starting from exiting the `Roaming` folder and go into the `Local` folder.

2. **pip or pip3 is not recognized as an internal or external command**
   If this issue happens, try interchanging `pip3` with `pip` or `pip3.10`.

   If the error still persists, you can try prepending the pip commands with either `python -m` or `py -m`. For example,

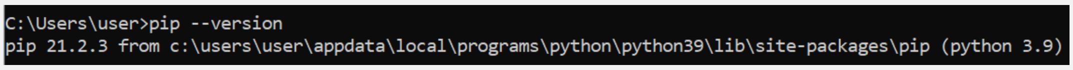
   ```
   python -m pip3 install pillow seaborn cocos2d --no-cache-dir
   ```

   For reference, please take a look at this Medium article.

3. **Checking if pip is installed**
   With your command prompt, try running

   `pip3 --version` or `pip --version`

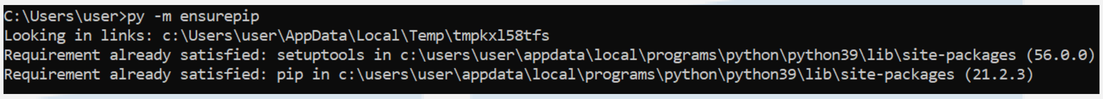
If you have more than one version of Python, try running `pip3.10 --version`. If either one of these commands produce an output similar to the one below, then you are good to go.



```
C:\Users\user>pip --version
pip 21.2.3 from c:\users\user\appdata\local\programs\python\python39\lib\site-packages\pip (python 3.9)
```

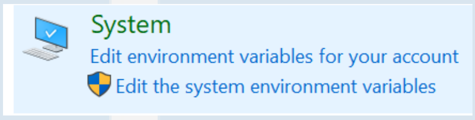
If all commands fail, try prepending these commands with either `python -m` or `py -m`.

Alternatively you can run either `python -m ensurepip` or `py -m ensurepip`. The output should be something similar to this.



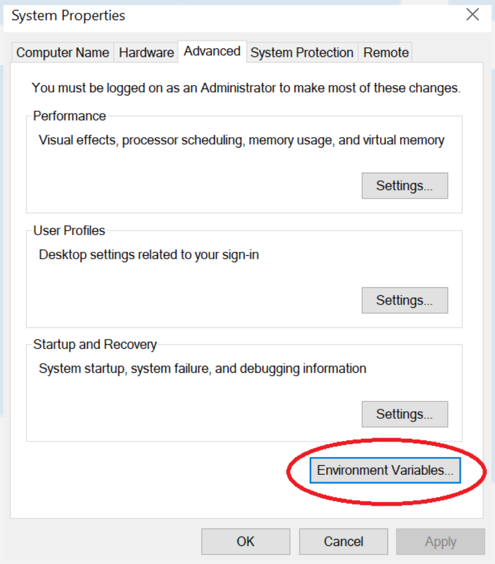
```
C:\Users\user>py -m ensurepip
Looking in links: c:\Users\user\AppData\Local\Temp\tmpkxl58tfs
Requirement already satisfied: setuptools in c:\users\user\appdata\local\programs\python\python39\lib\site-packages (56.0.0)
Requirement already satisfied: pip in c:\users\user\appdata\local\programs\python\python39\lib\site-packages (21.2.3)
```

4. **Adding Python to PATH manually**
   Go to Control Panel, and find this option by using the search bar.



System
Edit environment variables for your account
Edit the system environment variables

Open `Edit the system environment variables`, then `Environment Variables`. For each step, your display should be like the ones below.



System Properties

Computer Name | Hardware | Advanced | System Protection | Remote

You must be logged on as an Administrator to make most of these changes.

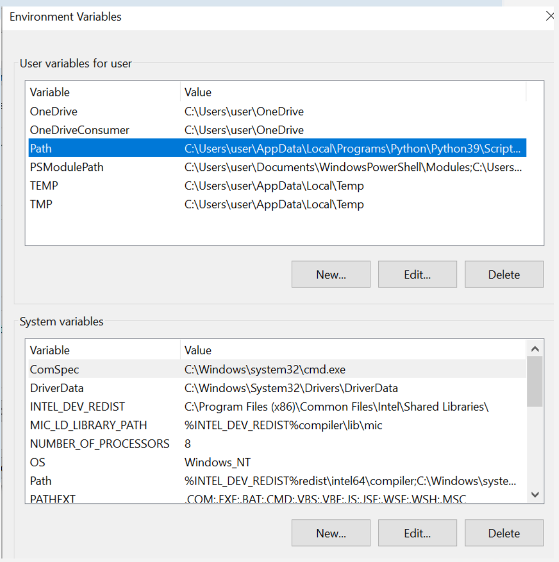
Performance
Visual effects, processor scheduling, memory usage, and virtual memory
Settings...

User Profiles
Desktop settings related to your sign-in
Settings...

Startup and Recovery
System startup, system failure, and debugging information
Settings...

Environment Variables...

OK Cancel Apply



Environment Variables

User variables for user

Variable	Value
OneDrive	C:\Users\user\OneDrive
OneDriveConsumer	C:\Users\user\OneDrive
Path	C:\Users\user\AppData\Local\Programs\Python\Python39\Script...
PSModulePath	C:\Users\user\Documents\WindowsPowerShell\Modules;C:\Users...
TEMP	C:\Users\user\AppData\Local\Temp
TMP	C:\Users\user\AppData\Local\Temp

New... Edit... Delete

System variables

Variable	Value
ComSpec	C:\Windows\system32\cmd.exe
DriverData	C:\Windows\System32\Drivers\DriverData
INTEL_DEV_REDIST	C:\Program Files (x86)\Common Files\Intel\Shared Libraries\
MIC_LD_LIBRARY_PATH	%INTEL_DEV_REDIST%compiler\lib\mic
NUMBER_OF_PROCESSORS	8
OS	Windows_NT
Path	%INTEL_DEV_REDIST%redist\intel64\compiler;C:\Windows\syste...
PATHEXT	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC

New... Edit... Delete

Select `Path` from `User variables for user`, then `Edit`. You can add Python310's path by clicking `New`.

To check if Python's path is already added to the PATH variable, go back to the command prompt and type `echo %PATH%`. It will give you a list of directories added to the PATH variable. Make sure the copied path is already there.

5. **Checking if the packages are installed**
   One of these commands should work on your command prompt.

   - `pip freeze`

- `pip3 freeze`
- `pip3.10 freeze`

What these commands do is checking all the Python packages you've installed so far. The result should be a list with a format of

```
<package-name>==<package-version>
```

For example, `scipy==1.7.3`. With this, you can see whether the required packages are installed correctly.

Alternatively, you can also use `pip list` instead of `pip freeze`.

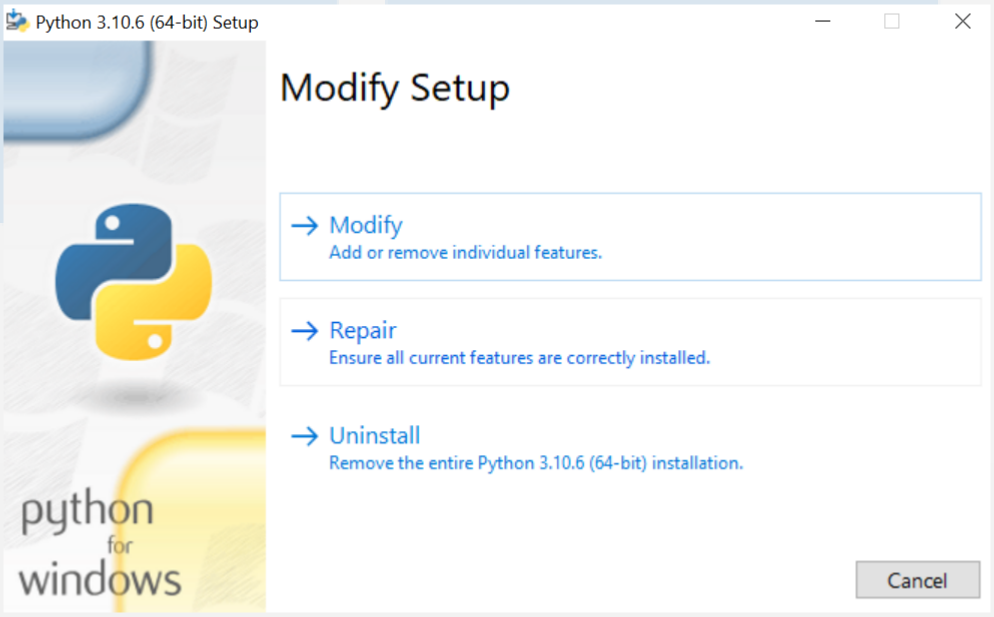
6. **Upgrading pip**
During installing the packages, you might receive a warning in yellow that pip must be upgraded. This is not necessary as long as you can work on with the Python packages normally. However, should you need it, then run this line on the command prompt.

```
pip --install upgrade pip
```

If this fails, try prepending this command with either `python -m` or `py -m`.

7. **Reinstalling Python**
Reinstalling Python might be needed if you have two or more instances of Python (e.g. Anaconda, Microsoft Store) and many errors appear during package installation due to certain clashes, which forces you to uninstall/delete them. Here's what to do **suppose you have to reinstall Python 3.10.6 again**.

   - Reopen the **executable file** that you obtained from Python's website. This should be the display the moment you open it.



Python 3.10.6 (64-bit) Setup

Modify Setup

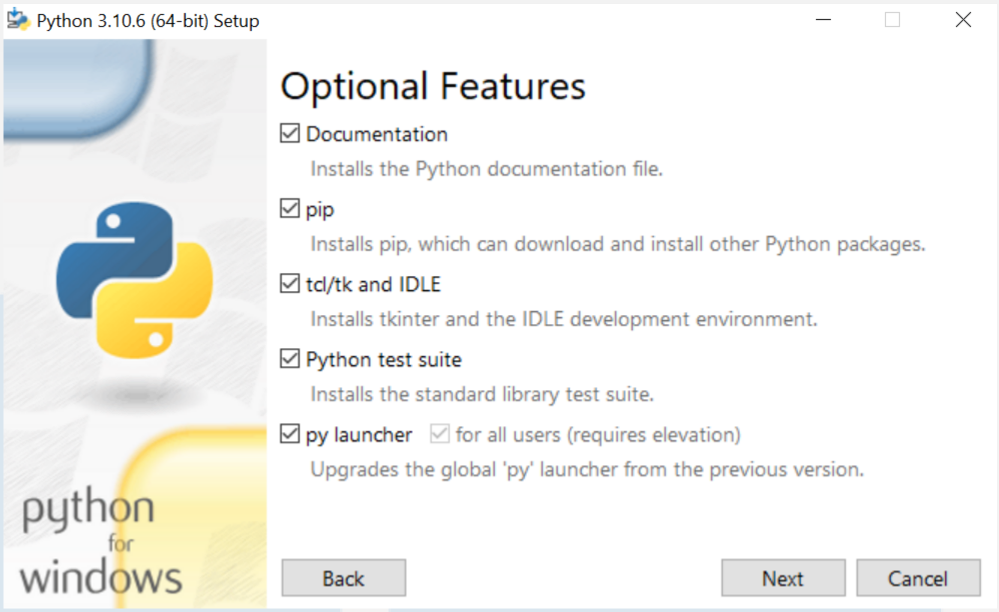
→ Modify
Add or remove individual features.

→ Repair
Ensure all current features are correctly installed.

→ Uninstall
Remove the entire Python 3.10.6 (64-bit) installation.

Note that you can also uninstall Python as seen in the last option above.

- Click `Modify` and check off **everything**. The last option is unchecked by default, so ensure that this is also checked off.



Optional Features

☑ Documentation
Installs the Python documentation file.

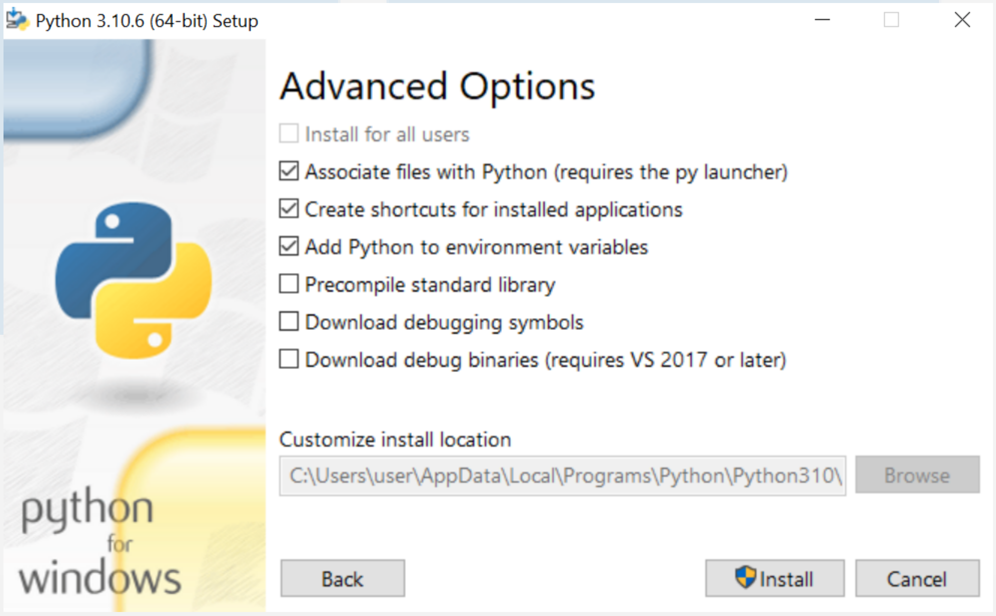
☑ pip
Installs pip, which can download and install other Python packages.

☑ tcl/tk and IDLE
Installs tkinter and the IDLE development environment.

☑ Python test suite
Installs the standard library test suite.

☑ py launcher ☑ for all users (requires elevation)
Upgrades the global 'py' launcher from the previous version.

- Click `Next`, and tick **the first three tickable options**. It is optional to tick the last three. Finally, click `Install`.



Advanced Options

☐ Install for all users
☑ Associate files with Python (requires the py launcher)
☑ Create shortcuts for installed applications
☑ Add Python to environment variables
☐ Precompile standard library
☐ Download debugging symbols
☐ Download debug binaries (requires VS 2017 or later)

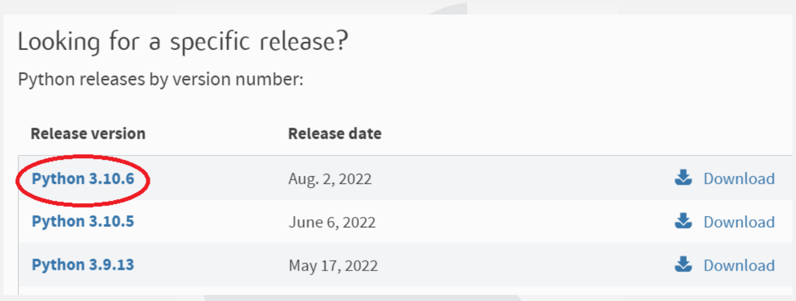
Customize install location
C:\Users\user\AppData\Local\Programs\Python\Python310\

Now that you have finished self-troubleshooting your installation, you may proceed to *the next part*.

Mac Users:

Step 1: Install Python for Mac

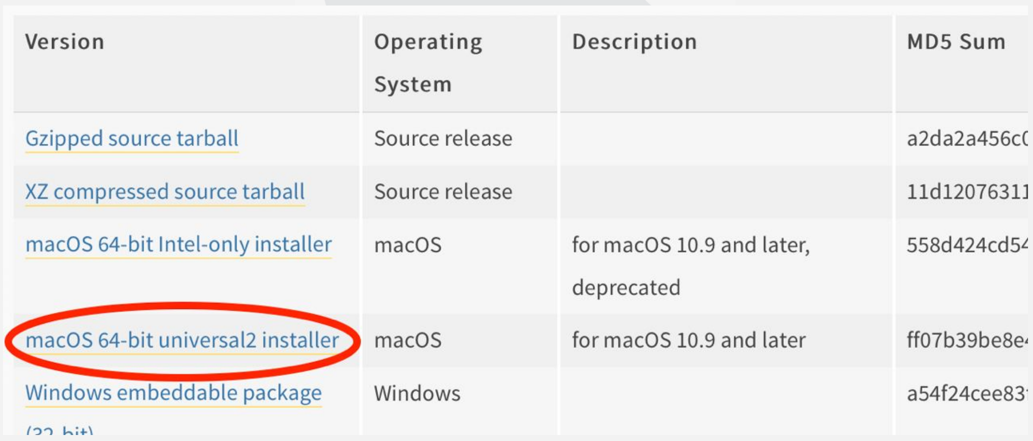
Download Python 3.10.6 **macOS 64-bit universal2 installer** from `https://www.python.org/downloads` and run the pkg installer.



Looking for a specific release?

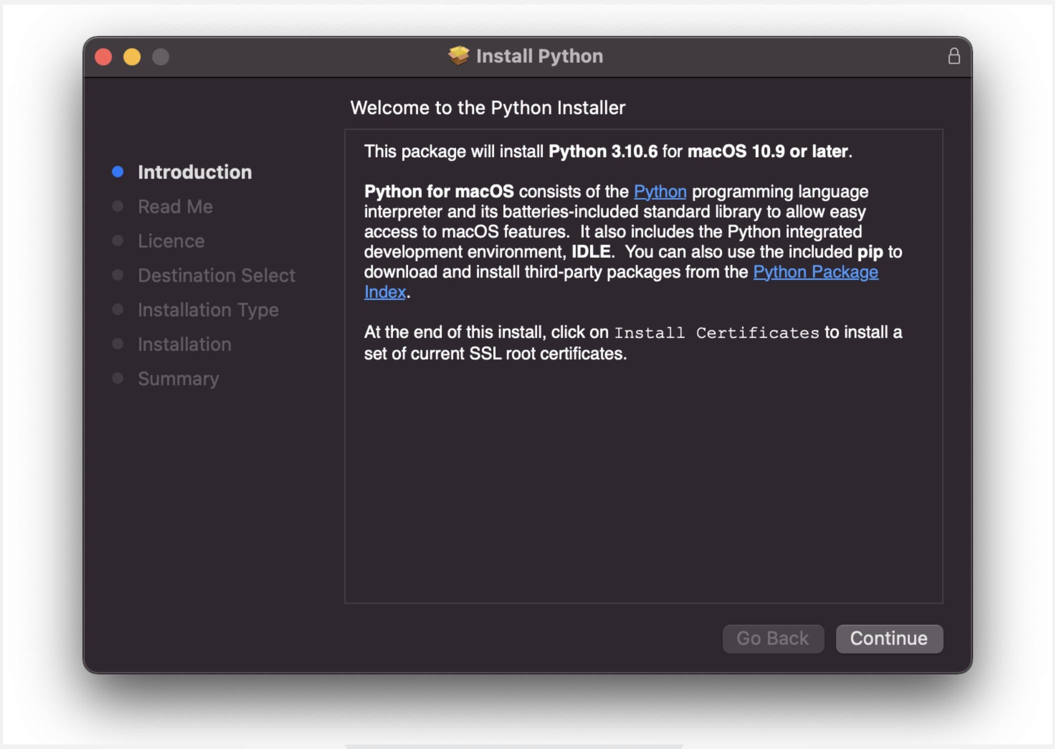
Python releases by version number:

Release version	Release date	
Python 3.10.6	Aug. 2, 2022	Download
Python 3.10.5	June 6, 2022	Download
Python 3.9.13	May 17, 2022	Download



Version	Operating System	Description	MD5 Sum
Gzipped source tarball	Source release		a2da2a456c(
XZ compressed source tarball	Source release		11d12076311
macOS 64-bit Intel-only installer	macOS	for macOS 10.9 and later, deprecated	558d424cd54
macOS 64-bit universal2 installer	macOS	for macOS 10.9 and later	ff07b39be8e
Windows embeddable package (32-bit)	Windows		a54f24cee83

Once installation is completed, you should see that **IDLE** is available from your finder.



Welcome to the Python Installer

This package will install **Python 3.10.6** for **macOS 10.9 or later**.

Python for macOS consists of the Python programming language interpreter and its batteries-included standard library to allow easy access to macOS features. It also includes the Python integrated development environment, **IDLE**. You can also use the included **pip** to download and install third-party packages from the Python Package Index.

At the end of this install, click on `Install Certificates` to install a set of current SSL root certificates.

Step 2: Installing the packages

Run the following commands in your terminal. (You can find the terminal by clicking on Finder on the dock, Go > Utilities > Terminal.)

```
pip3 install pillow seaborn cocos2d --no-cache-dir
```

If you see a bunch of red text, the packages might not be installed properly. Please take a look at the section next page. As a last resort, take a screenshot and check the Coursemology forums for assistance.

Otherwise, you may skip the extra troubleshooting part and proceed to *the next part*.

More Troubleshooting

Here are some commonly found issues that might help you. **Please read them carefully as your problem might be included below.**

1. **SciPy or Pandas cannot compile correctly**
   Attempting to run the installation command might sometimes result in compilation errors for M1 devices.
   Please follow these steps for an alternative installation method.
   (a) Install Homebrew by running

   ```
   /bin/bash -c "$(curl -fsSL
   https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"
   ```

   as one line.

   (b) Read and follow any additional instructions on the Terminal window.

   (c) Run

   ```
   brew install openblas gfortran
   ```

   (d) Run

   ```
   pip3.10 freeze | xargs pip uninstall -y
   ```

   If you receive a prompt to upgrade your pip version, please do so by running the command on screen.

   (e) Quit the Terminal application. **Quitting Terminal is different than merely closing the Terminal window. Make sure you don't have any Terminal window open.**

   (f) Locate Terminal in your Finder (User -> Applications -> Utilities). Select Terminal and press `Command + I`. **Check** the "Open with Rosetta" option.

   (g) Reopen Terminal.

   (h) Run

   ```
   export OPENBLAS=/opt/homebrew/opt/openblas/lib/
   ```

   (i) Run

   ```
   pip3.10 install pybind11 cython pythran pillow seaborn cocos2d
   --no-cache-dir
   ```

   as one line.

   (j) Locate Terminal in your Finder (User -> Applications -> Utilities). Select Terminal and press `Command + I`. **Uncheck** the "Open with Rosetta" option.

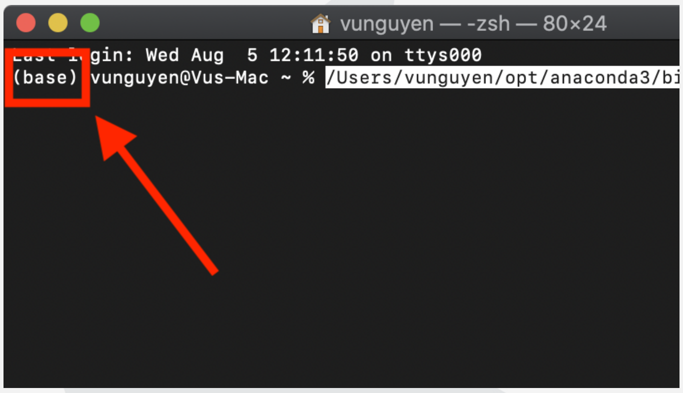
   (k) Quit the IDLE application if it is open.

   (l) Locate IDLE in your Finder (User -> Applications -> Python3.10). Select IDLE and press `Command + I`. **Check** the "Open with Rosetta" option.

   (m) Try and run the import statements from Mission 0 again.

2. **Packages installed correctly but unable to import**
   or
   Previously installed Anaconda/Conda
   Sometimes due to multiple versions of Python installation present, IDLE will not import packages from the correct version. This issue is especially pertinent to users who have previously installed Anaconda.

   If you are unsure about whether you have Anaconda installed, open a new Terminal window and look out for (base) on the left.



```
Last login: Wed Aug  5 12:11:50 on ttys000
(base) vunguyen@Vus-Mac ~ % /Users/vunguyen/opt/anaconda3/bi
```

   To remove Anaconda, follow the guide at
   https://docs.anaconda.com/anaconda/install/uninstall/.

   Beware of the rm -rf command. Incorrect usage might result in PERMANENT and IRRECOVERABLE loss of data.

   Additionally, open `.zshrc` or `.bashrc` and remove all traces of Anaconda. Check if the removal process is under control by closing all Terminal instances and reopening them.

   However, for any reason you are/want/like to keep Anaconda around, you do not have to remove them. Simply run `deactivate` to exit from the Anaconda base environment every time you launch Terminal. You can then follow the installation instructions in Mission 0 again.

Now that you have finished self-troubleshooting your installation, you may proceed to *the next part*.

Linux Users:

If you are *really* on Linux, then you are `1337` and will know how to install Python :)

Just kidding. You can approach the TA for help or any tutor familiar with Linux.

Editing Python Files

The default behaviour of double clicking on a Python file **executes** the content of the Python file. You should see the a command line window briefly open, and close when Python finishes executing the file.

In order to **edit** a Python file, you will need to use the **IDLE** application.

- Windows Users: Right click on the Python file > **Edit with IDLE**
- Mac Users: Right click on the Python file > **Open With** > **IDLE**

The content of the Python file should now appear in **IDLE**. You can then make changes to the file, and execute it.

To **execute** the Python file, go to **Run** > **Run Module**. The output of your Python file should then appear in the Shell Window.

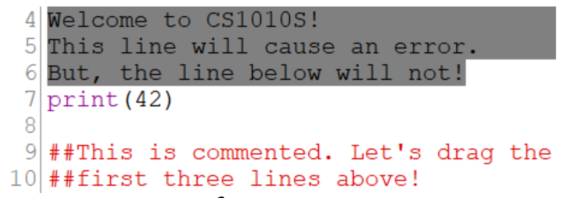
For Windows users, if these instructions still fail, you may read `Fix 'Edit with IDLE' missing from right-click (Windows).pdf` provided along with this PDF file. Change the versions in the instructions into 3.10.6 as well.

Part 2: The Task

Consider the following Python expressions. Your job is to predict the output when each expression is evaluated in IDLE. The first three has been done for you as an example.

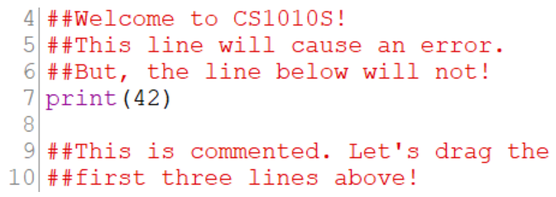
Before checking your answers with IDLE, write down briefly your guess of what the interpreter would **display** when the expressions are evaluated **sequentially** as listed. If you do not expect any output, you may write "no output" and if you expect an error, you may write "error".

Now, run the code by removing the # in the front of the respective lines in the template file. You should leave error-causing lines in their commented out state by adding # to the front of such lines to allow IDLE to skip processing them. (Or by selecting the area that you would like to comment out and then pressing IDLE's hot-key `alt+3` for Windows, `control+3` for MacOS)



```
Welcome to CS1010S!
This line will cause an error.
But, the line below will not!
print(42)

##This is commented. Let's drag the
##first three lines above!
```

Before commenting



```
##Welcome to CS1010S!
##This line will cause an error.
##But, the line below will not!
print(42)

##This is commented. Let's drag the
##first three lines above!
```

After commenting

Commenting with IDLE's hot-key

After you have checked your answers, if any expression has evaluated differently from your expected answer, write down what it evaluated to **below** your expected answer. **Please note that you will be graded only on your final answer (If you have no corrections to make, your initial answer will be your final answer).**

However, if any expression would generate an error message after running, please **specify the type of error** (such as `TypeError`) together with the **error message** in your final answer. You do not need to include the full error output. The required answer is usually only the *last line* of the error output.

Please use the template file provided in **mission00-template.py** instead of copying the code snippet from this PDF file.

Reminder: Lines that begin with a # are comments (text that do not affect the Python execution). To execute a particular expression, ensure it does not begin with a #.

```
# Example 1:
# I expect the result to be zero but upon printing the result is 0.
print(0)
# expected answer: zero
# final answer: 0

# Example 2:
# I expect the result to be 1 which is correct,
# so I don't need to do anything for the final answer.
print(1)
# expected answer: 1
# final answer:
```

```
# Example 3:
# This print statement results in an error,
# so I need to comment it and put the specific error as shown.
#print(a)
# expected answer: NameError: name 'a' is not defined
# final answer:

print(42)
# expected answer:
# final answer:

print(0000)
# expected answer:
# final answer:

print("the force!")
# expected answer:
# final answer:

print("Hello World")
# expected answer:
# final answer:

print(6 * 9)
# expected answer:
# final answer:

print(2 + 3)
# expected answer:
# final answer:

print(2 ** 4)
# expected answer:
# final answer:

print(2.1**2.0)
# expected answer:
# final answer:

print(15 > 9.7)
# expected answer:
# final answer:

print((5 + 3) ** (5 - 3))
# expected answer:
# final answer:

print(--4)
# expected answer:
# final answer:
```

```
print(1 / 2)
# expected answer:
# final answer:

print(1 / 3)
# expected answer:
# final answer:

print(1 / 0)
# expected answer:
# final answer:

print(7 / 3 == 7 / 3.0)
# expected answer:
# final answer:

print(3 * 6 == 6.0 * 3.0)
# expected answer:
# final answer:

print(11 % 3)
# expected answer:
# final answer:

print(2 > 5 or (1 < 2 and 9 >= 11))
# expected answer:
# final answer:

print(3 > 4 or (2 < 3 and 9 > 10))
# expected answer:
# final answer:

print("2" + "3")
# expected answer:
# final answer:

print("2" + "3" == "5")
# expected answer:
# final answer:

print("2" <= "5")
# expected answer:
# final answer:

print("2 + 3")
# expected answer:
# final answer:
```

```
print("May the force" + " be " + "with you")
# expected answer:
# final answer:

print("force"*2)
# expected answer:
# final answer:

print('daw' in 'padawan')
# expected answer:
# final answer:

a, b = 3, 4 # Do not comment this line

print(a)
# expected answer:
# final answer:

print(b)
# expected answer:
# final answer:

a, b = b, a # Do not comment this line

print(a)
# expected answer:
# final answer:

print(b)
# expected answer:
# final answer:

print(red == 44)
# expected answer:
# final answer:

red, green = 44, 43 # Do not comment this line

print(red == 44)
# expected answer:
# final answer:

print(red = 44)
# expected answer:
# final answer:

print("red is 1") if red == 1 else print("red is not 1")
# expected answer:
# final answer:
```

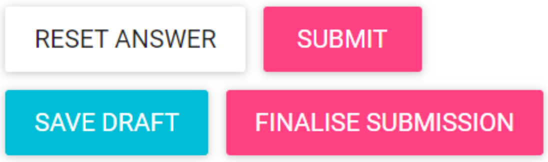
```
print(red - green)
# expected answer:
# final answer:

purple = red + green # Do not comment this line

print("purple")
# expected answer:
# final answer:

print("purple"[7])
# expected answer:
# final answer:

print(red + green != purple + purple / purple - red % green)
# expected answer:
# final answer:

print(green > red)
# expected answer:
# final answer:

print("green bigger") if green > red else print("red equal or bigger")
# expected answer:
# final answer:

print(green + 5)
# expected answer:
# final answer:

print(round(3.8))
# expected answer:
# final answer:

print(int(3.8))
# expected answer:
# final answer:

print(int("3.8"))
# expected answer:
# final answer:

# Do not comment these lines
def f(n):
    if n == 1: return 1
    return n + f(n - 1)

print(f(4))
# expected answer:
# final answer:
```

```
print(f(f(2)))
# expected answer:
# final answer:

print(f(0))
# expected answer:
# final answer:

d = {1: 2} # Do not comment this line

print(d[1])
# expected answer:
# final answer:

print(d[2])
# expected answer:
# final answer:

d[2] = "apple" # Do not comment this line

print(d[2])
# expected answer:
# final answer:

###########################################################
# The following 7 questions are to ensure that you have   #
# installed all the packages correctly:                   #
# - PILLOW                                                #
# - matplolib                                             #
# - scipy                                                 #
# - seaborn                                               #
# - numpy                                                 #
# - pyglet                                                #
# - cocos                                                 #
# Just uncomment the line "from <package> import *"       #
# and observe the output.                                 #
#                                                         #
# If there is no output, the packages have been installed #
# correctly! Answer "Nothing" to let us know that it's    #
# working.                                                #
###########################################################

from PIL import *
# expected answer:
# final answer:

from matplotlib import *
# expected answer:
# final answer:
```

```
from scipy import *
# expected answer:
# final answer:

from seaborn import *
# expected answer:
# final answer:

from numpy import *
# expected answer:
# final answer:

from pyglet import *
# expected answer:
# final answer:

from cocos import *
# expected answer:
# final answer:
```

To submit your work to Coursemology, complete **mission00-template.py** and **copy all the contents from the file** into the box on the mission page, and click "Save Draft." You can continue to make changes to your submission if you wish.

If you want to check whether your submission passes the test cases provided, click "Submit". The autograder provided will evaluate your answer and tell whether you fail some test cases or not. **By doing this, you can still continue to amend your submission.**



Once you are satisfied with your solution, click "Finalize Submission." **Note that once your submission is finalized, it is considered to be submitted and cannot be changed.** If you need to undo this action, you will have to email your tutor or the lecturer. You will not be allowed to undo your *finalized* submission, or a penalty might be imposed. **Please do not finalize your submission until you are sure that you want to submit your solutions for grading**.