Overview

This is a setup guide for Advanced Python for JPMC (rev. 1.1) on ***Windows***

Python is open source (*i.e.*, free). Each student needs a computer (or remote account) with Python, some extra modules, an IDE (Integrated Development Environment) such as PyCharm, and the student files installed.

NOTE	This setup guide should work in most environments, but is not guaranteed to work in all possible situations. Please call or email your contact with any further questions.

Steps for installation

Setup for this class requires 4 separate installation steps:

1. Installing the lab files specific to this course
2. Installing **Python**
3. Installing some extra Python packages (modules) that are not part of the standard library
4. Installing and setting up **PyCharm**, an Integrated Development Environment for **Python**

Anaconda vs. python.org installation

There are two approaches you can use for the **Python** installation. You can install the **Anaconda Individual Edition** bundle from **anaconda.com** *OR* install basic **Python**, and then add any needed extra packages.

The easiest approach is to install the Anaconda bundle. This is a free (community) bundle that installs basic Python and many additional libraries in a single step.

IMPORTANT	Follow *either* **Step 2-A** *or* **Step 2-B**, but not both.

Student files

The student files contain examples, data, and answers to labs. They will be provided prior to class.

IDE/Editor

We recommend **PyCharm** as a Python IDE (Integrated Development Environment) and it is part of the installation specifications below. However, some programmers already have a favorite IDE or editor. We do not *require* students to use **PyCharm**. If students are already using **Spyder**, **Visual Studio Code**, **Eclipse**, **Notepad++**, **emacs** or other tool, that will not cause a problem.

NOTE Links sometimes change outside our control. Be sure to download *Windows* versions of all specified software.

Detailed Setup

Step 1: Installing the student (lab) files

The lab file archives contain setup, example, data, and answer files for use in the labs.

The file name is **py3jpmcadv_1.1.zip**

The zip file should be extracted to the user's desktop. It will create a folder named **py3jpmcadv**.

The easy way to extract the files is just to double-click on the zip file, and then drag the extracted folder to the desktop.

If you want to use the **Extract** menu, be sure the target folder is

```
C:\users\USERNAME\desktop
```

NOT

```
C:\users\USERNAME\desktop\py3jpmcadv
```

The **Extract** menu defaults to the second form, which adds a confusing extra **py3jpmcadv** folder.

2-A: Installing Python from Anaconda

1. Download the latest **Anaconda Individual Edition** installer from https://www.anaconda.com/download/. Install, using default responses.

IMPORTANT Download and install Python 3, not Python 2

2-B: Installing Python from python.org

1. Download the latest Python 3 installer from http://www.python.org/download/ . Be sure to download the 64-bit Windows installer.
2. Once downloaded, double-click the `.exe` file to start installing.

Choose "Install Python 3.*x* for all users", and select the "Add python .exe to Path" option in the installer.

TIP If the installation seems to be hanging, check to see if there’s a Windows dialog asking for permission to proceed.

Step 3 Installing extra packages

This class may require some Python packages (libraries) that are not part of the standard Python installation. To be sure everything needed gets installed we provide a **requirements.txt** file that lists the extra packages.

If Ananconda was installed, open a new Anaconda prompt after basic Anaconda installation is complete by searching the main Windows menu for "Anaconda Prompt".

Use the **pip** command, which comes with Python, to install these additional packages. This step is to make sure all required packages get installed.

```
pip install -r requirements.txt
```

NOTE Some or all of the files in `requirements.txt` may already be installed, especially if you have installed the **Anaconda Personal Edition**.

Troubleshooting pip

If pip does not run from the command line, try this:

```
python -m pip install -r requirements.txt
```

Step 4: Installing and configuring PyCharm

Part A: Installing PyCharm

Install the latest version of **PyCharm Community Edition** from

```
http://www.jetbrains.com/pycharm/download
```

If the installer stops and asks for admin rights, just click **no** and keep going.

At end of install:

- don’t import settings from other versions
- don’t customize PyCharm

- don’t create .py association
- do create a desktop shortcut (if desired)
- don’t open folder as project

IMPORTANT Do not install the Professional or Educational Edition

Part B: Setting up the project

The next step is to configure a Python interpreter and open the student files as a PyCharm project.

Start up PyCharm. It will create a sample project. Go to the **File** menu and select **Close Project**. Now use **File**→**Open** open the **py3jpmcadv** folder on your desktop.

PyCharm will open the project and auto-configure the Python interpreter. You are ready to go!

NOTE If you have an older version of PyCharm, it will not autoconfigure the interpreter, and you will have to set it up manually. Continue with the following steps:

Older PyCharm versions (before 2020.1)

You will need to know the location of the actual interpreter (an **.exe** file). If you have installed the standard interpreter, or any other non-Anaconda interpreter, you will need to figure out its path before continuing

Step 1. Get the interpreter path

- Open a command prompt
  - If using Anaconda
    - Use the Windows menu to open the Anaconda Prompt application. Otherwise, open a normal command prompt (Do not open Anaconda *Navigator*).
  - If NOT using Anaconda
    - Open a normal command window (hint: search for "cmd")
- In the terminal window, type `python`. This brings up the Python interactive prompt (`>>>`
- At the >>> prompt, enter the following two commands (don’t type the ">>>")

```
>>> import sys
>>> sys.executable
```

  - Copy the output of `sys.executable` to the clipboard. This is the path to your Python interpreter.
  - Exit the Python interactive prompt by typing `quit()`

IMPORTANT Be sure to exit the Python prompt (last step above)

TIP You can open Notepad or another editor and paste the path into a document so you'll have it.

Step 2. Configure the interpreter

- Start Pycharm via the Windows menu or the desktop shortcut
- Using the **File** menu, close any open projects. This will leave only the PyCharm welcome screen.
- On the welcome screen, click Configure
- Select **Settings**
- On the **Settings** menu (on the left), select **Project Interpreter**

NOTE Even though it says *Project* Interpreter, you are selecting the default *Python* interpreter for future projects.

- Try the dropdown next to "interpreter:".
- If your interpreter is there, select it and go to part C
- It is more likely that the interpreter is *not* there.
  - Add the interpreter by choosing its path
    - Click the "gear" icon to the right of the dropdown
    - Select "Add…". This brings up the *Add Python Interpreter* window
    - Select **System Interpreter** from the menu on the left
    - Select the interpreter configured in the the beginning of this section
- Close the **Settings** window; you should be back to the welcome screen

**Step 3: Create a PyCharm project with the lab files

Once the Python interpreter is configured, create a PyCharm project with the class files.

- From the welcome screen, click on *Open*
- On the **New Project** screen
  - Use the button with a file folder icon to select the folder which is the top level of the student files. This will normally be on your desktop, unless you put it somewhere else. It should be something like
    C:\Users\YOURNAME\Desktop\py3jpmcadv
  - Click on Project Interpreter to open it up
  - Click *Existing Interpreter*

    - Select the interpreter you configured in Part B
  - Click *Create*
  - Answer Y to the question about creating a project from existing files

This completes the installation. You are ready to go.

Overview

This is a setup guide for Advanced Python for JPMC (rev. 1.1) on ***Linux***

Python is open source (*i.e.*, free). Each student needs a computer (or remote account) with Python, some extra modules, an IDE (Integrated Development Environment) such as PyCharm, and the student files installed.

NOTE This setup guide should work in most environments, but is not guaranteed to work in all possible situations. Please call or email your contact with any further questions.

Steps for installation

Setup for this class requires 4 separate installation steps:

1. Installing the lab files specific to this course
2. Installing **Python**
3. Installing some extra Python packages (modules) that are not part of the standard library
4. Installing and setting up **PyCharm**, an Integrated Development Environment for **Python**

Anaconda vs. python.org installation

There are two approaches you can use for the **Python** installation. You can install the **Anaconda Individual Edition** bundle from **anaconda.com** *OR* install basic **Python**, and then add any needed extra packages.

The easiest approach is to install the Anaconda bundle. This is a free (community) bundle that installs basic Python and many additional libraries in a single step.

IMPORTANT Follow *either* **Step 2-A** *or* **Step 2-B**, but not both.

Student files

The student files contain examples, data, and answers to labs. They will be provided prior to class.

IDE/Editor

We recommend **PyCharm** as a Python IDE (Integrated Development Environment) and it is part of the installation specifications below. However, some programmers already have a favorite IDE or editor. We do not *require* students to use **PyCharm**. If students are already using **Spyder**, **Visual Studio Code**, **Eclipse**, **Notepad++**, **emacs** or other tool, that will not cause a problem.

NOTE Links sometimes change outside our control. Be sure to download *Linux* versions of all specified software.

Detailed Setup

Step 1: Installing the student (lab) files

The lab file archives contain setup, example, data, and answer files for use in the labs.

The file name is **py3jpmcadv_1.1.tar.gz**

Download or copy **py3jpmcadv_1.1.tar.gz** to the user's desktop. Extract to the user's desktop or home folder. It will create a directory named **py3jpmcadv**.

Sample tar extraction command (execute as the user, not as root):

```
cd
tar xzvf py3jpmcadv_1.1.tar.gz
```

2-A: Installing Python from Anaconda

1. Download the latest **Anaconda Individual Edition** installer from https://www.anaconda.com/download/. Install, using default responses.

IMPORTANT Download and install Python 3, not Python 2

2-B: Installing Python from python.org

1. Python may already be installed. If not, install Python 3 from http://www.python.org/download/.

Step 3 Installing extra packages

This class may require some Python packages (libraries) that are not part of the standard Python installation. To be sure everything needed gets installed we provide a **requirements.txt** file that lists the extra packages.

If Ananconda was installed, open a new terminal window (shell prompt) prompt after basic Anaconda installation is complete.

Use the **pip** command, which comes with Python, to install these additional packages. This step is to make sure all required packages get installed.

```
pip install -r requirements.txt
```

NOTE Some or all of the files in `requirements.txt` may already be installed, especially if you have installed the **Anaconda Personal Edition**.

Troubleshooting pip

If pip does not run from the command line, try this:

```
python -m pip install -r requirements.txt
```

Step 4: Installing and configuring PyCharm

Part A: Installing PyCharm

Install the latest version of **PyCharm Community Edition** from

```
http://www.jetbrains.com/pycharm/download
```

At end of install:

- don’t import settings from other versions
- don’t customize PyCharm
- don’t create .py association
- do create a desktop shortcut (if desired)
- don’t open folder as project

IMPORTANT Do not install the Professional or Educational Edition

Part B: Setting up the project

The next step is to configure a Python interpreter and open the student files as a PyCharm project.

Start up PyCharm. It will create a sample project. Go to the **File** menu and select **Close Project**. Now use **File→Open** open the **py3jpmcadv** folder on your desktop.

PyCharm will open the project and auto-configure the Python interpreter. You are ready to go!

NOTE If you have an older version of PyCharm, it will not autoconfigure the interpreter, and you will have to set it up manually. Continue with the following steps:

Older PyCharm versions (before 2020.1)

You will need to know the location of the actual interpreter (an **.exe** file). If you have installed the standard interpreter, or any other non-Anaconda interpreter, you will need to figure out its path before continuing

Step 1. Get the interpreter path

- Open a command prompt
- In the terminal window, enter the command `type python` and press Enter/Return.
- Copy the resulting path to the clipboard (just the path, not the "python is" part).
- Exit the Python interactive prompt by typing `quit()`

TIP You can open **nano**, **vi**, or some other editor and paste the path into a document so you'll have it for later.

Step 2. Configure the interpreter

- Start Pycharm via the app menu or the desktop shortcut
- Using the **File** menu, close any open projects. This will leave only the PyCharm welcome screen.
- On the welcome screen, click Configure
- Select **Settings**
- On the **Settings** menu (on the left), select **Project Interpreter**

NOTE Even though it says *Project* Interpreter, you are selecting the default *Python* interpreter for future projects.

- Try the dropdown next to "interpreter:".
- If your interpreter is there, select it and go to part C
- It is more likely that the interpreter is *not* there.
    - Add the interpreter by choosing its path
        - Click the "gear" icon to the right of the dropdown
        - Select "Add…". This brings up the *Add Python Interpreter* window
        - Select **System Interpreter** from the menu on the left
        - Select the interpreter configured in the the beginning of this section
- Close the **Settings** window; you should be back to the welcome screen

**Step 3: Create a PyCharm project with the lab files

Once the Python interpreter is configured, create a PyCharm project with the class files.

- From the welcome screen, click on *Open*
- On the **New Project** screen
  - Use the button with a file folder icon to select the folder which is the top level of the student files. This will normally be on your desktop, unless you put it somewhere else. It should be something like
    /Users/YOURNAME/Desktop/py3jpmcadv
  - Click on Project Interpreter to open it up
  - Click *Existing Interpreter*
    - Select the interpreter you configured in Part B
  - Click *Create*
  - Answer Y to the question about creating a project from existing files

This completes the installation. You are ready to go.

Overview

This is a setup guide for Advanced Python for JPMC (rev. 1.1) on ***Mac***

Python is open source (*i.e.*, free). Each student needs a computer (or remote account) with Python, some extra modules, an IDE (Integrated Development Environment) such as PyCharm, and the student files installed.

NOTE This setup guide should work in most environments, but is not guaranteed to work in all possible situations. Please call or email your contact with any further questions.

Steps for installation

Setup for this class requires 4 separate installation steps:

1. Installing the lab files specific to this course
2. Installing **Python**
3. Installing some extra Python packages (modules) that are not part of the standard library
4. Installing and setting up **PyCharm**, an Integrated Development Environment for **Python**

Anaconda vs. python.org installation

There are two approaches you can use for the **Python** installation. You can install the **Anaconda Individual Edition** bundle from **anaconda.com** *OR* install basic **Python**, and then add any needed extra packages.

The easiest approach is to install the Anaconda bundle. This is a free (community) bundle that installs basic Python and many additional libraries in a single step.

IMPORTANT Follow *either* **Step 2-A** *or* **Step 2-B**, but not both.

Student files

The student files contain examples, data, and answers to labs. They will be provided prior to class.

IDE/Editor

We recommend **PyCharm** as a Python IDE (Integrated Development Environment) and it is part of the installation specifications below. However, some programmers already have a favorite IDE or editor. We do not *require* students to use **PyCharm**. If students are already using **Spyder**, **Visual Studio Code**, **Eclipse**, **Notepad++**, **emacs** or other tool, that will not cause a problem.

NOTE Links sometimes change outside our control. Be sure to download *Mac* versions of all specified software.

Detailed Setup

Step 1: Installing the student (lab) files

The lab file archives contain setup, example, data, and answer files for use in the labs.

The file name is **py3jpmcadv_1.1.tar.gz**

Download or copy **py3jpmcadv_1.1.tar.gz** to the user’s desktop. Extract to the user’s desktop or home folder. It will create a directory named **py3jpmcadv**.

Sample tar extraction command (execute as the user, not as root):

```
cd
tar xzvf py3jpmcadv_1.1.tar.gz
```

2-A: Installing Python from Anaconda

1. Download the latest **Anaconda Individual Edition** installer from https://www.anaconda.com/download/. Install, using default responses.

IMPORTANT Download and install Python 3, not Python 2

2-B: Installing Python from python.org

1. Install Python 3 for OS X from http://www.python.org/download/. Choose the latest version.

Step 3 Installing extra packages

This class may require some Python packages (libraries) that are not part of the standard Python installation. To be sure everything needed gets installed we provide a **requirements.txt** file that lists the extra packages.

If Ananconda was installed, open a new terminal window (shell prompt) prompt after basic Anaconda installation is complete.

Use the **pip** command, which comes with Python, to install these additional packages. This step is to make sure all required packages get installed.

```
pip install -r requirements.txt
```

NOTE Some or all of the files in `requirements.txt` may already be installed, especially if you have installed the **Anaconda Personal Edition**.

Troubleshooting pip

If pip does not run from the command line, try this:

```
python -m pip install -r requirements.txt
```

Step 4: Installing and configuring PyCharm

Part A: Installing PyCharm

Install the latest version of **PyCharm Community Edition** from

```
http://www.jetbrains.com/pycharm/download
```

At end of install:

- don’t import settings from other versions
- don’t customize PyCharm
- don’t create .py association
- do create a desktop shortcut (if desired)
- don’t open folder as project

IMPORTANT Do not install the Professional or Educational Edition

Part B: Setting up the project

The next step is to configure a Python interpreter and open the student files as a PyCharm project.

Start up PyCharm. It will create a sample project. Go to the **File** menu and select **Close Project**. Now use **File**→**Open** open the **py3jpmcadv** folder on your desktop.

PyCharm will open the project and auto-configure the Python interpreter. You are ready to go!

NOTE If you have an older version of PyCharm, it will not autoconfigure the interpreter, and you will have to set it up manually. Continue with the following steps:

Older PyCharm versions (before 2020.1)

You will need to know the location of the actual interpreter (an **.exe** file). If you have installed the standard interpreter, or any other non-Anaconda interpreter, you will need to figure out its path before continuing

Step 1. Get the interpreter path

- Open a command prompt
  - Using the Finder, you can open the **Terminal** application from **Applications/Utilities**
- In the terminal window, enter the command `type python` and press Enter/Return.
- Copy the resulting path to the clipboard (just the path, not the "python is" part).
- Exit the Python interactive prompt by typing `quit()`

Step 2. Configure the interpreter

- Start Pycharm via Applications or the desktop shortcut
- Using the **File** menu, close any open projects. This will leave only the PyCharm welcome screen.
- On the welcome screen, click Configure
- Select **Preferences**
- On the **Preferences** menu (on the left), select **Project Interpreter**

NOTE Even though it says *Project* Interpreter, you are selecting the default *Python* interpreter for future projects.

- Try the dropdown next to "interpreter:".
- If your interpreter is there, select it and go to part C
- It is more likely that the interpreter is *not* there.
  - Add the interpreter by choosing its path
    - Click the "gear" icon to the right of the dropdown
    - Select "Add…". This brings up the *Add Python Interpreter* window
    - Select **System Interpreter** from the menu on the left
    - Select the interpreter configured in the the beginning of this section
- Close the **Preferences** window; you should be back to the welcome screen

**Step 3: Create a PyCharm project with the lab files

Once the Python interpreter is configured, create a PyCharm project with the class files.

- From the welcome screen, click on *Open*
- On the **New Project** screen
    - Use the button with a file folder icon to select the folder which is the top level of the student files. This will normally be on your desktop, unless you put it somewhere else. It should be something like
      /Users/YOURNAME/Desktop/py3jpmcadv
    - Click on Project Interpreter to open it up
    - Click *Existing Interpreter*
        - Select the interpreter you configured in Part B
    - Click *Create*
    - Answer Y to the question about creating a project from existing files

This completes the installation. You are ready to go.