



Python Programming

Part 1

v0.3 (2024)



Yihsiang Liow

Preface

I hope one day I have notes (open books) for all my classes so that students won't have to spend 100's of dollars on books. This is one of them. It's still under heavy construction.

Goal

Learn Python programming, learn some CS stuff, and have fun. You will learn how to do basic graphics and animation. Woo hoo

Audience

Anyone who can read, has access to a computer and the Internet, and most importantly, wants to learn. I do not assume any knowledge of programming. However, I do assume you want to learn.

Disclaimer

While this set of notes is being constantly being revised, we do still miss some errors, so please make this a great book by reporting bugs to yliow@ccis.edu.

Acknowledgments

I want to thank the following people for using and testing out these set of notes.

- CCPC (Columbia College Programming Club) 2006
 - Josh Lory
 - Shahthureen Khan
 - Nick Myers
 - Brad Nullmeyer
 - Rosalie Purvis
 - Dan Zhao

- CISS145 Spring 2009

The following helped with revision:

- Spring 2007: Nick Myers
- Summer 2024: Ryan Harvey

Welcome

Welcome to Python Programming Part 1!!!

Well of course you want to know what you will learn right? You will learn how to write **Python** programs. We will be doing **graphics and animation**. Of course by doing all the above, you will have a glimpse into the world of **Computer Science**. Who knows? Maybe one day you will be a software engineer working for Microsoft or EA.

And most important of all, along the way, we will have lots of **fun**.

What you will need

You need the following to join the club:

- **PC**. I'll assume that you have a Windows desktop or laptop machine. Specifically you're using Windows 10/11 as your operating system (OS). Don't worry if you don't know what OS you're using. If you do not know what you're using, then you're most probably using Windows.
- **Internet access**. I assume you have fast internet access.

Resources

There are many ways to learn Python programming. Since you're reading this page, I'm assuming that you're using my notes. They are designed to be covered in about 10-14 weeks.

FOR CCCS: If you're learning this in our Columbia College Computer Science Club (CCCS), then besides the notes you'll have the following resources:

- During club meetings you will work in **small groups**. Learning from each other helps to speed up your learning process.
- There are also **mentors** in the club that will help you learn. These mentors are Computer Science students in Columbia College.
- The club maintains a **message board** where you can interact outside the club meetings.

FOR CISS145:

- You are part of the class. There are other students in the class. So ... get to know each other and learn together.
- Make full use of the class time or google group. Ask questions.

All the notes are free. Furthermore, I have selected reading material and references so that you may deepen your mastery of the Python programming language. The extra references are freely available on the web. All these can be accessed through our club's web site.

What else you need before the fun begins

OK. So you have the equipment, you have the notes, and you're a member of CCCS. You're all fired up to have fun. But wait. There's something else ...

I need to talk about **you**.

Programming is like building something. While a electrical or computer engineer builds hardware (and more!), a software engineer builds software. Both hardware and software “thingies” that they build are like machines.

There are two main obstacles on your way to become a software engineer (and this is somewhat similar to the hardware engineers). First you will need to learn to use the tools. Next you need to know how to design a solution. You can think of your tools as things to help you transform your design to a real machine.

For writing programs in Python, the main tool will be a programming language called Python.

Learning a programming language is no different from learning any other human language. Certain words have special meanings. And you basically use these words to control the PC. (There are details ... but we can ignore them for the time being).

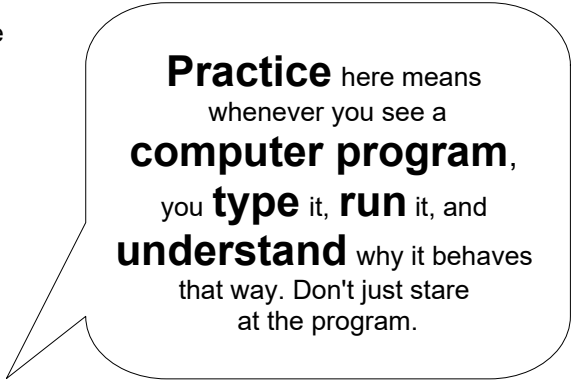
Designing a solution more or less means planning out what you want to do. This is much harder than learning the tools.

But back to learning the language of Python. Much of learning the language is actually very simple. The only thing you need to do in order to know the language well is to **practice** using the language. Like learning any other language, this takes time. Much of remembering the words or grammar of a language involves repetitive usage. Therefore you need to **persevere** and **be patient**. I guarantee you that your patience will be greatly rewarded.

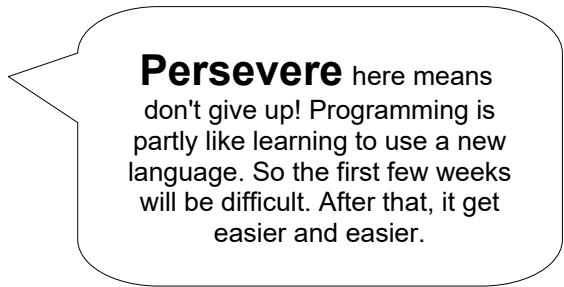
Remember to make full use of the resources available. **Read** the notes and if necessary the references. Practice using the Python language to **solve problems** given in the notes. **Ask** questions. If you need help outside the meetings, use the message board. Let's not only have fun, but have fun together by learning together. Other members will be really grateful if you can answer their questions on the message board.

If you are determined to learn to program, then you will. Someone will be around to help you.

OK. Enough talk. Let the fun begin ...



Practice here means whenever you see a **computer program**, you **type** it, **run** it, and **understand** why it behaves that way. Don't just stare at the program.



Persevere here means don't give up! Programming is partly like learning to use a new language. So the first few weeks will be difficult. After that, it get easier and easier.

Software Installation (DIY)

Before we begin, let's just say that I'm assuming you're using a 64-bit Windows machine. Most of what I'm going to say applies to for instance MacOS as well.

I'll talk about running Python

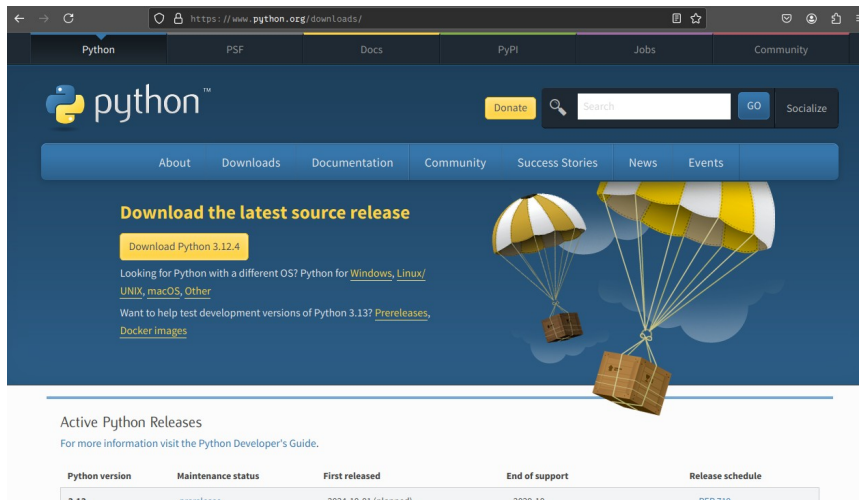
- In your platform (i.e. in Windows)
- In a linux virtual machines

Besides Python, I'll be using PyGame, a game writing library for Python.

Installation of Firefox

This section is optional.

My favorite browsers are Firefox (**FF**) and Chrome. You may use Microsoft Edge (**Edge**). The only reason why I'm bring this up is because if I do a screen shot of the browser you will very likely see FF:

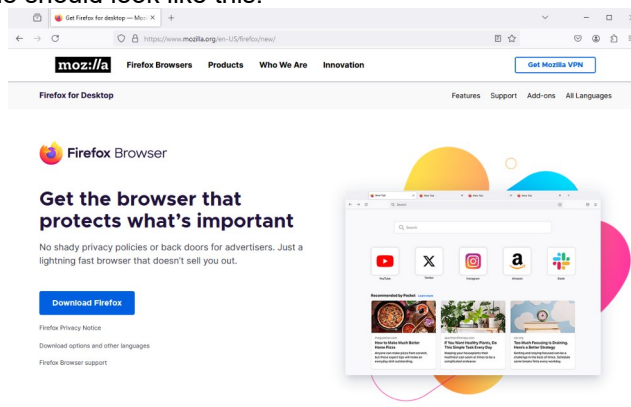


Why am I not telling you the URL of Firefox's home page? Because you should learn to hunt for things on your own.

FF, Chrome and Edge are probably instead installed in your laptop. If FF is not installed in your laptop and you wish to try out FF follow the given instructions ...

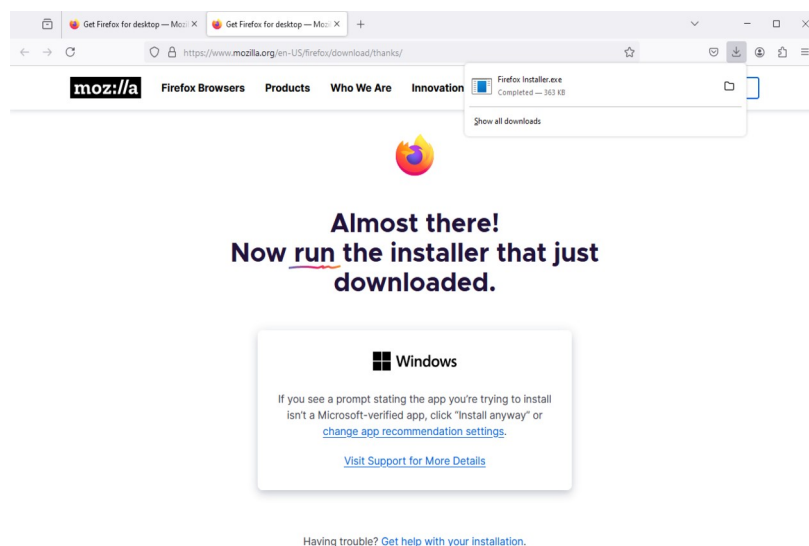
If you're comfortable looking for and installing software, all I need to tell you is to look for FF and install it. If you feel uncomfortable about that, please read on.

First open your IE. Go to "www.google.com". Search for "Firefox". Click on the link that gives you the home page of FF (it should be the first). FF's home should look like this:



Click on "Download Firefox" and you should see this:

Python Programming Part 1 v0.3



Run the installer program. Follow the instructions. To run the installation program manually, you need to double-click on this icon.

In general, during the installation process you accept the default choices. You usually do that by clicking on “Next”. You might need to agree with some license agreement. When you're done you probably have to click on “Finish”.

On your Desktop you will see



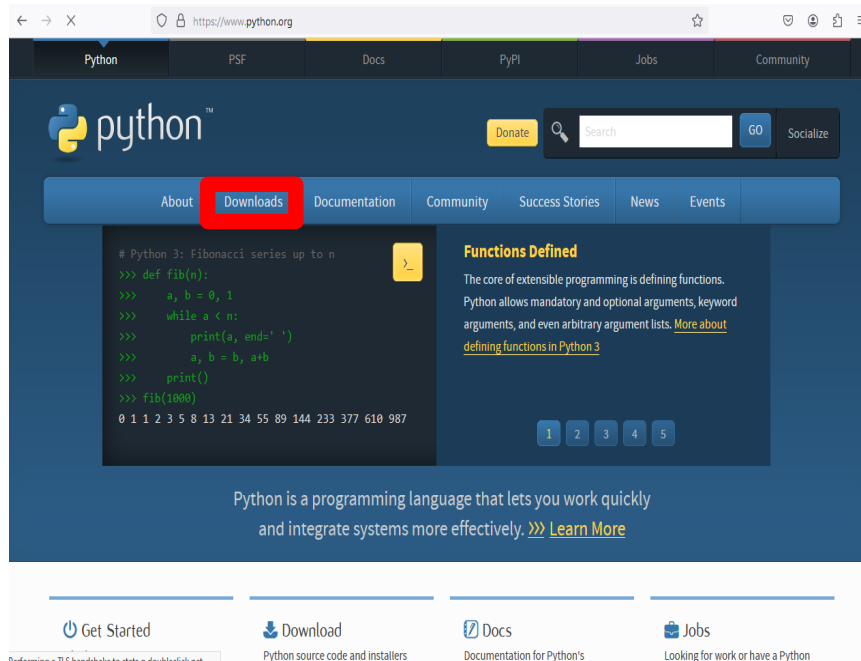
You double-click on this icon to run firefox. Of course after you're done installing FF, you can get rid of the installation program.

On Windows: installation of Python

The Python software does not come with your PC or notebook by default if you're using the Windows operating system. Here's how you make your PC understand Python.

First you need to find the home page of Python. Google for “python” and find it. It should look like this:

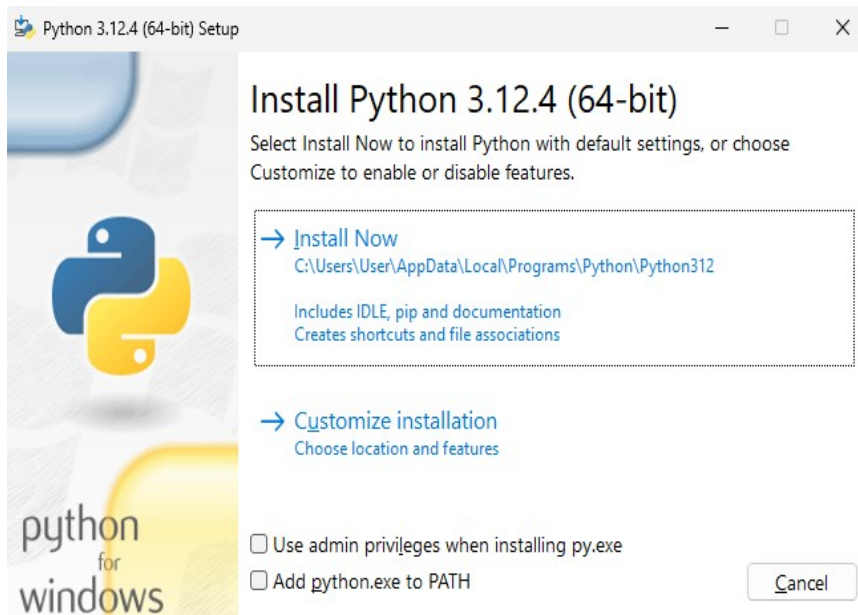
Google knows everything



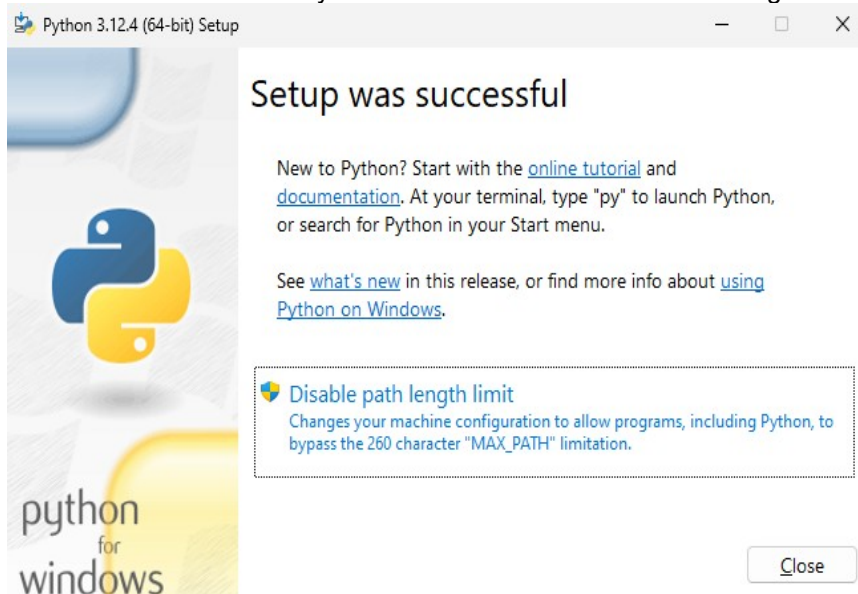
Hover mouse over Downloads, and click the “Python 3.12.4” link under “Download for Windows,” to install Python 3.12.4.

This will download the python installation setup, which, when opened, will look like this:

Python Programming Part 1 v0.3



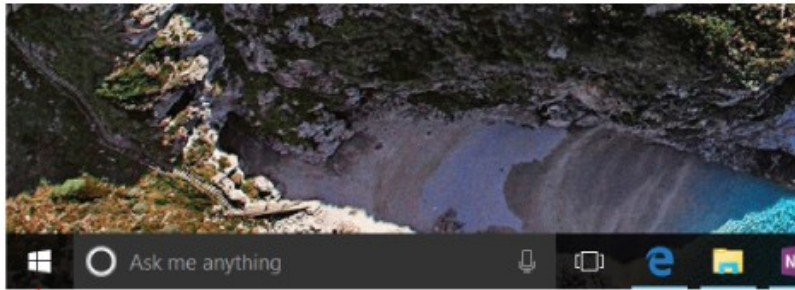
Click on “Install Now” and you will see this after it’s done installing:



Click on “Close” ... and you're done! You now have the Python software on your computer.

When you open the start menu:

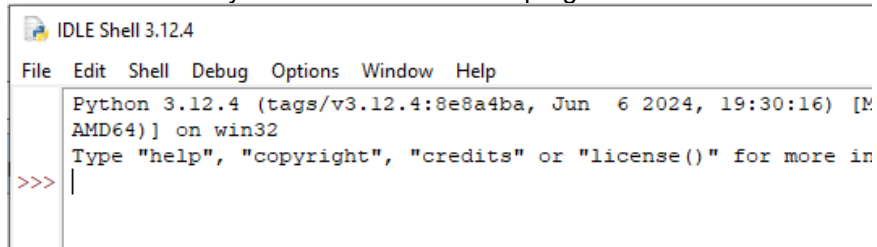
Python Programming Part 1 v0.3



you should be able to find your python:



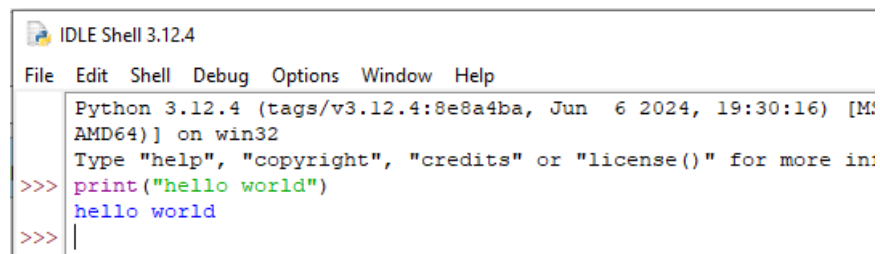
Click on IDLE and you'll see the IDLE Shell program:



Let's test your python software.

Enter `print("hello world")` into the shell and press the enter key. You'll see this:

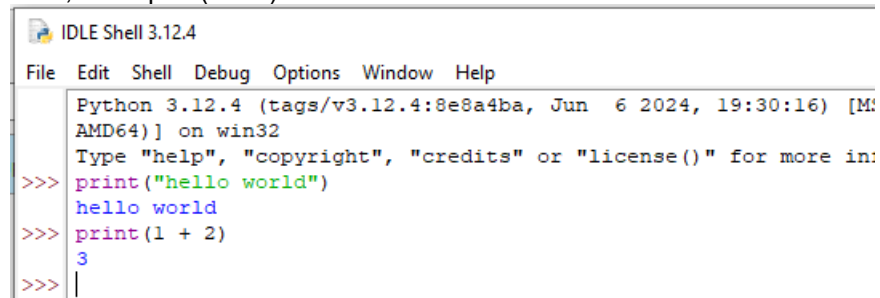
Python Programming Part 1 v0.3



```
IDLE Shell 3.12.4
File Edit Shell Debug Options Window Help
Python 3.12.4 (tags/v3.12.4:8e8a4ba, Jun 6 2024, 19:30:16) [MS
AMD64] on win32
Type "help", "copyright", "credits" or "license()" for more in:
>>> print("hello world")
hello world
>>> |
```

The shell says hello world to you.

Next, enter `print(1 + 2)` in the shell:



```
IDLE Shell 3.12.4
File Edit Shell Debug Options Window Help
Python 3.12.4 (tags/v3.12.4:8e8a4ba, Jun 6 2024, 19:30:16) [MS
AMD64] on win32
Type "help", "copyright", "credits" or "license()" for more in:
>>> print("hello world")
hello world
>>> print(1 + 2)
3
>>> |
```

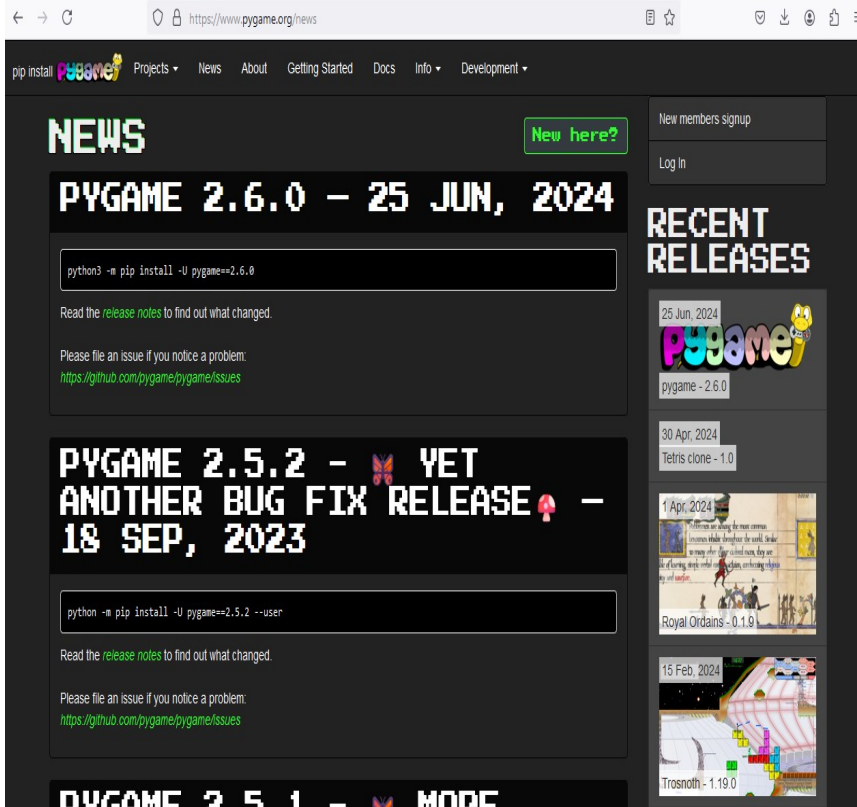
and the shell says 3 to you.

Congratulations! Your python software is working.

WARNING: There might be other versions of python already installed in your laptop. (Maybe you downloaded a program that uses Python.) Make sure you run the correct version that you want!

On Windows: installation of PyGame

PyGame is a python library for game development. First Google for “pygame” to find PyGame’s home page:



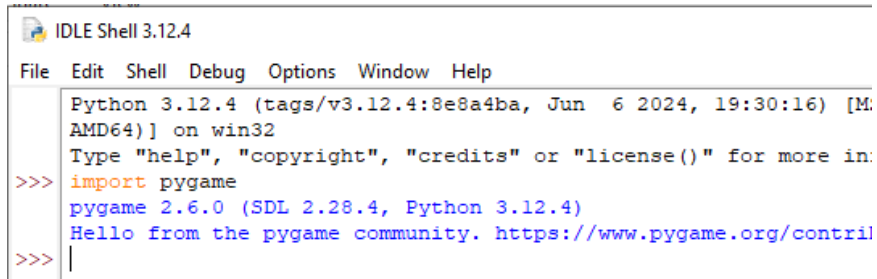
You can browse the website for more information on pygame and also on how to install pygame. I'll show you a different way of installing pygame ...

Open your IDLE shell again. Enter the following and then press the enter key:

```
import os
os.system("python pip -m install -U pygame --user")
```

Python should print 0. If you see the 0, it means that python has no problems installing pygame.

Let's test our pygame. Open IDLE shell again and enter `import pygame`:



```
IDLE Shell 3.12.4
File Edit Shell Debug Options Window Help
Python 3.12.4 (tags/v3.12.4:8e8a4ba, Jun 6 2024, 19:30:16) [AMD64] on win32
Type "help", "copyright", "credits" or "license()" for more in:
>>> import pygame
pygame 2.6.0 (SDL 2.28.4, Python 3.12.4)
Hello from the pygame community. https://www.pygame.org/contrib
>>> |
```

Python prints the pygame version. This means that pygame is successfully installed. If python has problems installing pygame, you'll get an error message when you try to import pygame.

To uninstall a library you do

```
import os
os.system("python pip -m uninstall pygame --user")
```

By the way you only need to do import os once. For instance

```
import os
os.system("python pip -m install -U pygame -user")
os.system("python pip -m install -U pyperclip -user")
```

Exercise. Install the emoji library. Then run this example in IDLE:

```
from emoji import emojize
print(emojize(":thumbs_up:"))
```

There's a similar library called emojis. Install it and then run this example:

```
import emojis
emojified = emojis.encode(":snake: in my boot!")
print(emojified)
```

Finally, uninstall the emoji and emojis libraries.

Exercises. Depending on the version of your Microsoft Windows, the python libraries are usually installed in one of the following folders:

C:/Users/yliow/AppData/Roaming/Python/Python312/site-packages
C:/Users/yliow/AppData/Local/Python/Python312/site-packages

(with my user name replaced by yours and the python version 312 might be different depending on which version you downloaded.) Open these folders and then install the emoji library and check where emoji is installed. Then, uninstall emoji and check that it's removed.

On Windows: how to use IDLE

This section is just to show you how to use the Python shell and IDLE software to run python commands. Don't worry about the concepts. That will come in a later chapter. There are two ways to run python commands using IDLE:

- Interactive mode using the Python shell
- Batch mode by running a Python program

Interactive mode

Ready? Remember to learn actively: Don't just stare at the notes.

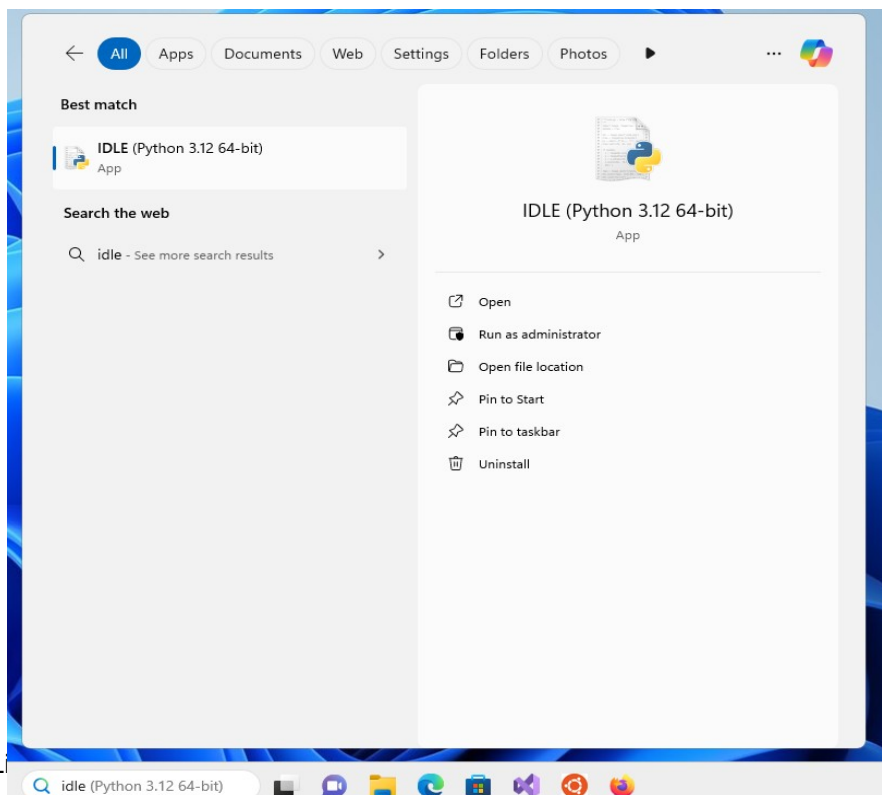
Follow the notes by typing, running and understanding all programs.

Before I begin showing your programs, I'll just give you an advice to minimize the amount of pain. **Type in whatever is**

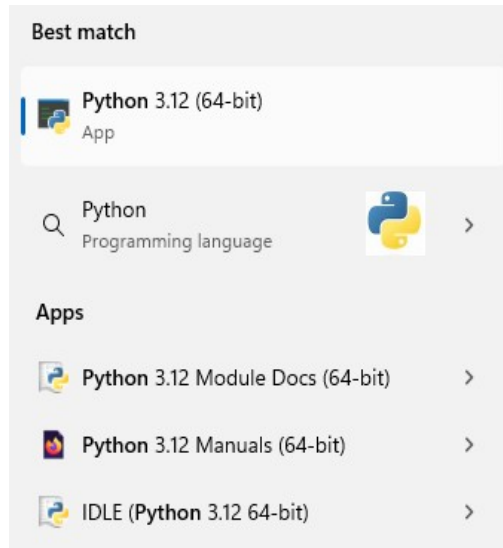
shown as is. Do not change the case (uppercase to lowercase or lowercase to upper) and do not insert or remove spaces.

OK. Here we go. First I'll show you how to run IDLE. This is a program that will allow you to execute Python commands and to write Python programs.

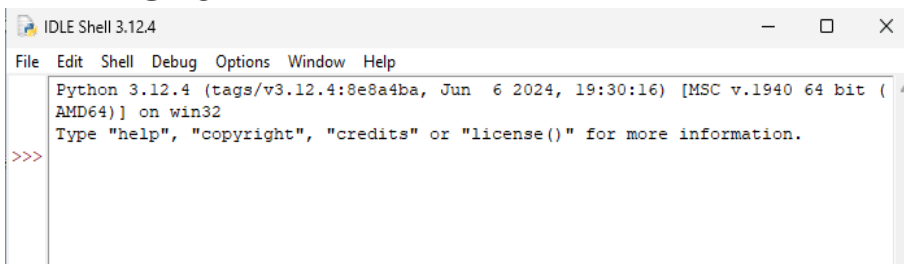
Click on the “Start” on the bottom left of your screen, select “Programs”:



(Your Windows desktop might look slightly different). Look for “Python 3.12” and click on it and you should see this:



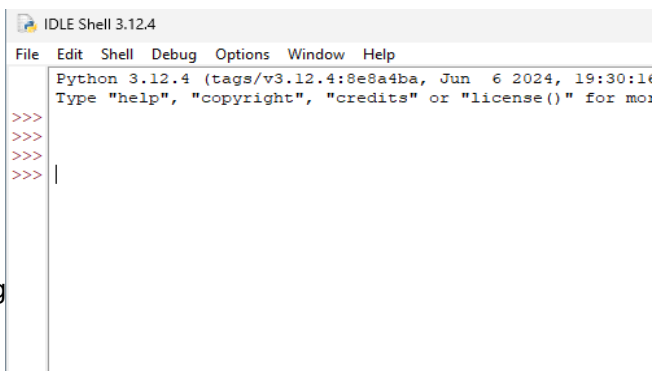
Click on “IDLE (Python 3.12 64-bit)” and you will get this window titled **IDLE Shell**:



Before I continue, in the future, instead of saying all the above to run IDLE I might say “Start > Programs > Python 3.12 > IDLE (Python GUI)”.

The program you just ran is called IDLE. At the top of the window you see “IDLE Shell”. This means that IDLE allows you to “talk” to Python by give it one command at a time. A Python “command” is called a **statement**.

First press your “enter” key several times:



Now type in `print(2)` and press the “enter” key. You should get this:

```
>>>
>>>
>>>
>>> print(2)
2
>>>
>>>
```

Woo hoo ... !!! You've just commanded the Python shell to print the number 2.

Nowadays, most people interact with programs by clicking on icons with the mouse. The Python shell interacts with us through a

command line interface. You don't see pictures or buttons. You enter text and the program responds with text (or graphics or sound ...)

The “>>>” is called a **prompt**:

```
>>>
>>>
>>>
>>> print(2)
2
>>>
>>>
```

Prompt

When you see the prompt, it means the Python shell is ready to receive a statement to execute.

Note also that when you pressed enter three times, the shell doesn't do anything. So it's OK to issue an “empty” statement.

Instead of saying “Python shell prints 2” or “IDLE prints 2”, from now on I might just say “Python prints 2”.

Actually to get Python to print 2 you can also type “2” and then press the “enter” key:

```
>>>
>>> print(2)
2
>>> 2
2
>>> |
```

Exercise. You have 10 seconds to close IDLE, open it again, and write a Python statement that prints the number 42.

Before we go on, I must let you run the following program:

```
>>> print("hello world")
hello world
>>> |
```

It's a tradition among programmers to run a hello world program the first time they learn a new language. The practice has been around since 1974.

Let's compare the statements:

```
print(2)
```

and

```
print("hello world")
```

The print statement allows you to print values to the shell's window.

While **2** is a whole **number**, **"hello world"** is a **string**. Think of a string as textual data for the time being.

The important thing to remember is that you **can't throw away the quotes in a string**.

Exercise. Can you mix strings and integers in a print statement? Try this:

```
print(1, 2, "buckle my shoes")
```

LISTEN VERY CAREFULLY! The above "experiments" based on minor modifications of the print statement

```
print(1, 2, 3, 4)
```

is what we call **active learning**. By experimenting with what you know, you are actively involved with the new knowledge and so it's easier to remember what you learned. Even more importantly, active learning allows you to go beyond what you know. In fact the skill of **learning to learn** through active learning is just as important as the knowledge you receive in your classroom or from books, important as it may be. So remember to experiment and to question what you learn from books, from classrooms, etc.

See what I mean by this advice:

A quick advice to ease the pain of learning your first

*programming language: Type the program **exactly** as given.
Even the spaces and blank lines must match my programs.*

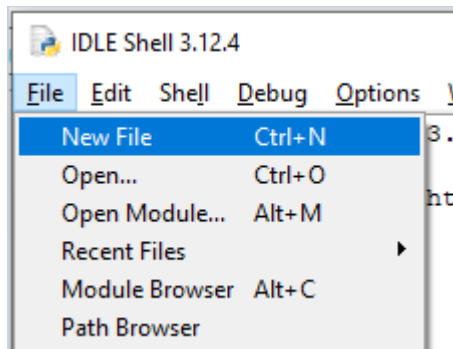
Computers are dumb (and picky about details.) We are the smart ones.
So ... when you communicate with your computer, you have to be exact
and explicit in your programs.

OK. Let's go on ...

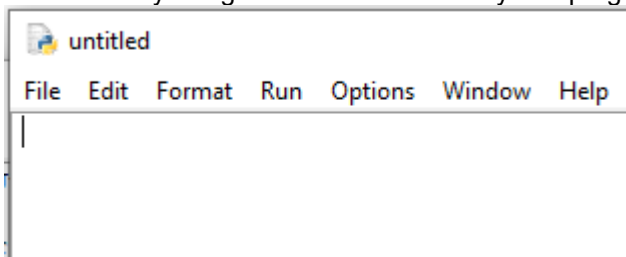
Exercise. Now write a program that prints any message (example: "do
you want green eggs and ham?") Close your notes first. Peek at your
notes only when you have problems.

Batch mode

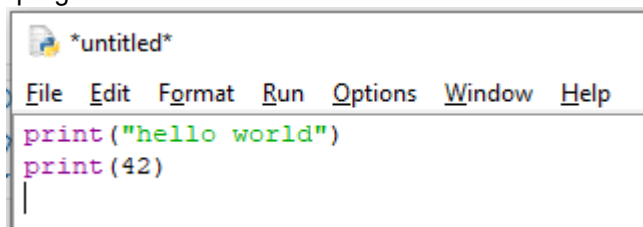
In your IDLE program, click on File and then click on New File:



Click on New File and you'll get an area to write a Python program:

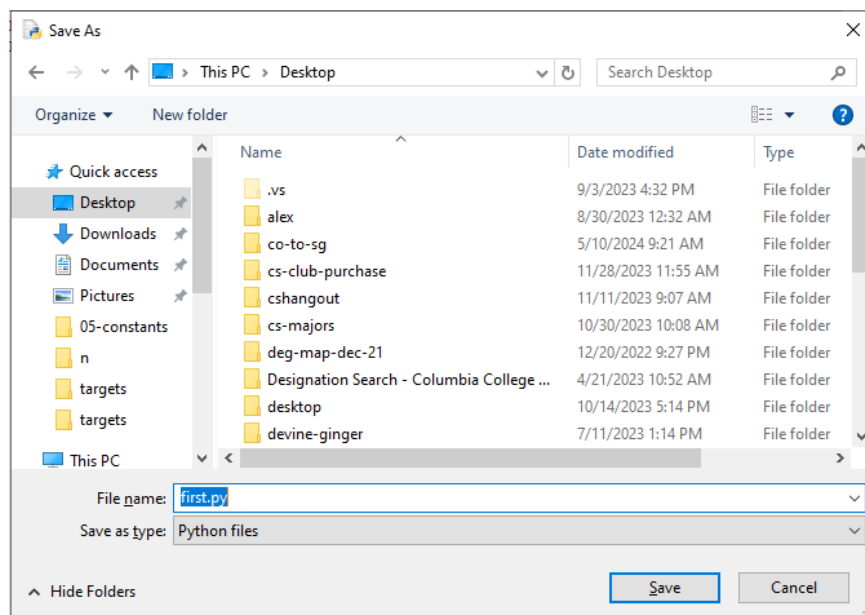


Enter this program:

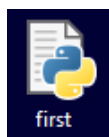


Click on File and then click on Save and you'll be prompted to save your
program:

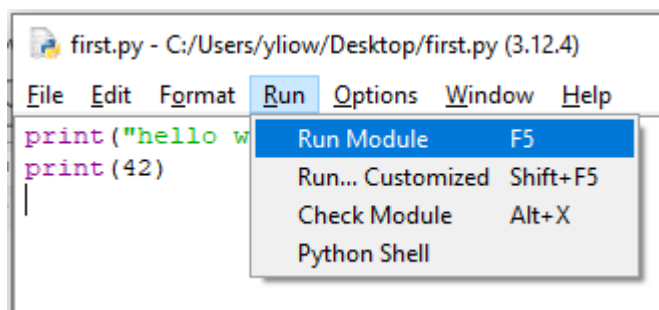
Python Programming Part 1 v0.3



You can click on “Desktop” to save your program on your Desktop. Enter a filename for your program. For instance you can use “first.py” as filename. Click on Save. Go to your Desktop and you’ll see your program’s icon:



To run the program, click on Run and then click on Run Module:



(Notice the short cut: press F5). And you’ll see and you’ll see IDLE shell pop up and and the output of your program is shown:



```
Python 3.12.4 (tags/v3.12.4:8e8a4ba, Jun 6  
AMD64) on win32  
Type "help", "copyright", "credits" or "license()">>>  
= RESTART: C:/Users/yliow/Desktop/first.py  
hello world  
42  
>>> |
```

Notice that both Python statements are executed.

If you need to re-open first.py, right-click on



and click on Edit with IDLE.

On Linux

Programming Python on Microsoft Windows is fine. But it's much better to write programs in linux. I'll show you how to using a linux virtual machine and how to write Python programs in the virtual machine. You only need to download the virtual machine and learn to use it – Python and Pygame are both installed in the virtual machine.

- Go to my website <http://yliow.github.io>.
- Scroll down and look for the Software section on the web page (or just search for software on the web page).
- Look for “Software downloads for classes” and click on the README and follow the instructions on running a linux platform. You will need to (1) download and install some software and you'll need to (2) go over some tutorials on how to use the linux platform. Specifically the instructions will direct you to:
 - Install VMware workstation if you are using a Microsoft Windows machine (or install VMware Fusion if you have a MacOS laptop).
 - Install 7zip (or any good decompression software that can decompress 7z files).
 - Download our Fedora virtual machine onto your laptop.
 - After the above, you are ready to go over the following tutorials:
 - [vmwareplayer.pdf](#): how to use VMware workstation (or fusion) to run a linux platform
 - [unix1.pdf](#): how to use linux commands
 - [emacs.pdf](#): how to use the emacs text editor to write programs

After the above, you can run Python commands in the following ways.

Interactive mode: To run Python interactively, open your bash shell and run `enter python` to run the python shell:

```
[student@localhost ~]$ python
Python 3.7.9 (default, Aug 19 2020, 17:05:11)
[GCC 9.3.1 20200408 (Red Hat 9.3.1-2)] on linux
Type "help", "copyright", "credits" or "license" for
more information.
>>>
```

and then enter python statements:

```
[student@localhost ~]$ python
Python 3.7.9 (default, Aug 19 2020, 17:05:11)
[GCC 9.3.1 20200408 (Red Hat 9.3.1-2)] on linux
Type "help", "copyright", "credits" or "license" for
more information.
>>> print("hello world")
hello world
>>> print(1 + 2)
3
>>>
```

Python Programming Part 1 v0.3

Notice that in the python shell, every time you enter a python command (the technical term is python statement), the python shell runs that command immediately.

To end the python shell and go back to the bash shell, enter Ctrl-d (i.e., hold the control key and press d).

Batch mode: To run a python program, use your text editor to write and save this program with filename `first.py`:

```
print("hello world")
print(1 + 2)
```

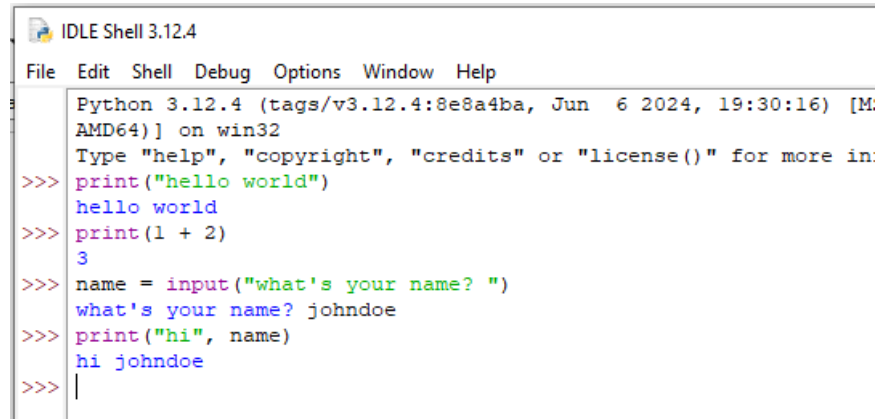
Next, run your `first.py` program in the bash shell by entering `python first.py`:

```
[student@localhost ~]$ python first.py
hello world
3
[student@localhost ~]$
```

Notice that in batch mode, you store all the python commands in a file and then run all of them in the bash shell or in IDLE.

On reading the notes

In the notes, to save on ink, when I run the following 4 python commands interactively in a python shell (such as IDLE), instead of showing you a screenshot:

A screenshot of the IDLE Shell 3.12.4 window. The title bar says "IDLE Shell 3.12.4". The menu bar includes "File", "Edit", "Shell", "Debug", "Options", "Window", and "Help". The status bar at the bottom says "Python 3.12.4 (tags/v3.12.4:8e8a4ba, Jun 6 2024, 19:30:16) [AMD64] on win32". The main text area shows the following interaction:

```
>>> print("hello world")
hello world
>>> print(1 + 2)
3
>>> name = input("what's your name? ")
what's your name? johndoe
>>> print("hi", name)
hi johndoe
>>> |
```

I might write

```
>>> print("hello world")
hello world
>>> print(1 + 2)
3
>>> what's your name? johndoe
hi johndoe
```

Notice that I've underlined

johndoe

to indicate that it's a user input to a python command.