Lecture 1 Introduction to Python

IDS 400
Programming for Data Science in Business

Tentative schedule

Date	Lecture Number	Topics
08/24	Lecture 1	Introduction
08/31	Lecture 2	Basic
09/07	Lecture 3	Condition
09/14	Lecture 4	Loop
09/21	Lecture 5	String + **Quiz 1** → Online
09/28	Lecture 6	Type
10/05	Lecture 7	Function
10/12	Lecture 8	File + **Quiz 2** → Online
10/19	Lecture 9	Pandas
10/26	Lecture 10	Numpy
11/02	Lecture 11	Machine Learning
11/09	Lecture 12	Visualization
11/16	Lecture 13	Web Scraping & Deep Learning
11/23	***Thanksgiving***	***No lecture***
11/30	**Final presentation**	In class presentation
12/05	**Project submission due**	

About me:

- My name is Aida (pronounced as "eye-duh.")
- Please call me by name or professor instead of Ms., Ma'am,..etc ☺
- I'm a 4th year PhD candidate in Management Information System
- Previously worked as a "system engineer"
- I love playing piano in my spare time

Introduce yourself:

- Your name
- Major
- Which year of the program
- The level of your programming skill
- Any hobby

Why Data Scientist ? [1]

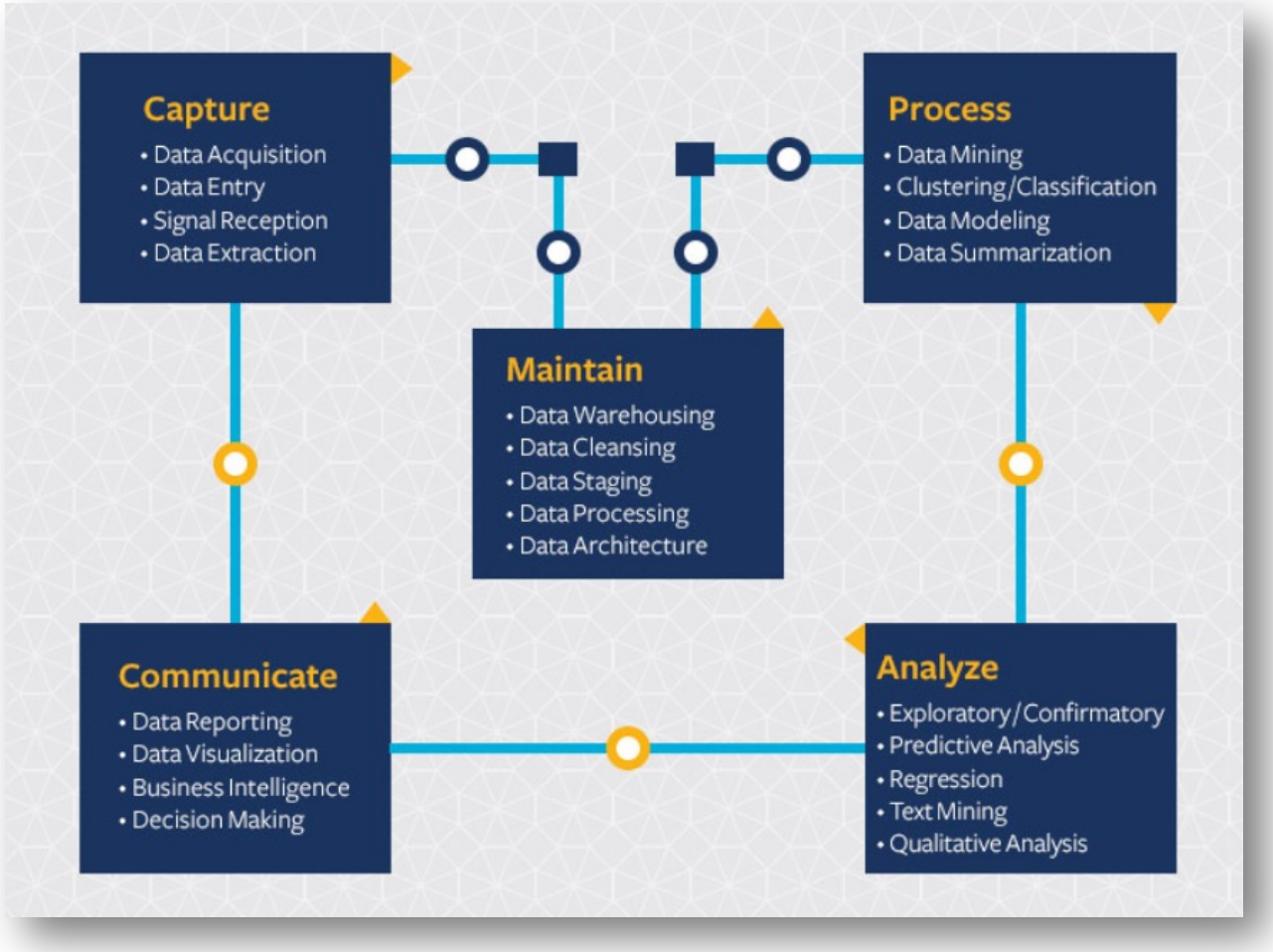
Glassdoor ranked data scientist as the #1 Best Job in America in 2018 for the third year in a row. The statistics listed below represent the significant and growing demand for data scientists.

[1]datascience@Berkeley https://datascience.berkeley.edu/about/what-is-data-science/

What is Data Analytics/Science?

- *Some key words?*
- *What are the necessary skills?*

The Data Science Life Cycle[1]



[1]datascience@Berkeley https://datascience.berkeley.edu/about/what-is-data-science/

Data Scientist [1]

Effective data scientists are able to

- Identify relevant questions.
- Collect data from a multitude of different data sources.
- Organize the information.
- Translate results into solutions.
- Communicate their findings in a way that positively affects business decisions.

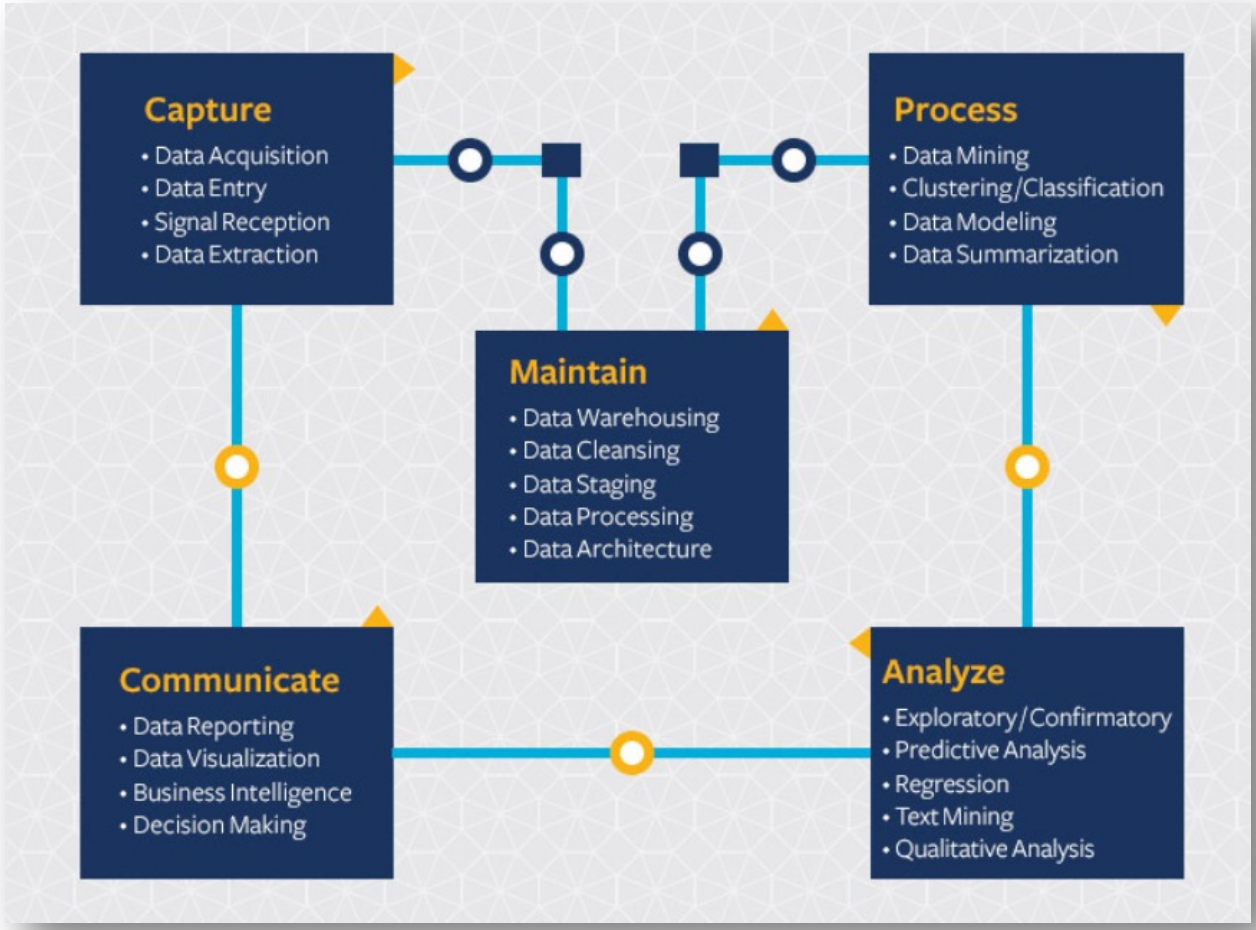
These skills are required in almost all industries, causing skilled data scientists to be increasingly valuable to companies.

[1]datascience@Berkeley https://datascience.berkeley.edu/about/what-is-data-science/

Why learning python for analytics?

- Reproducibility
- Size
- Automation
- …

This course



[1]datascience@Berkeley https://datascience.berkeley.edu/about/what-is-data-science/

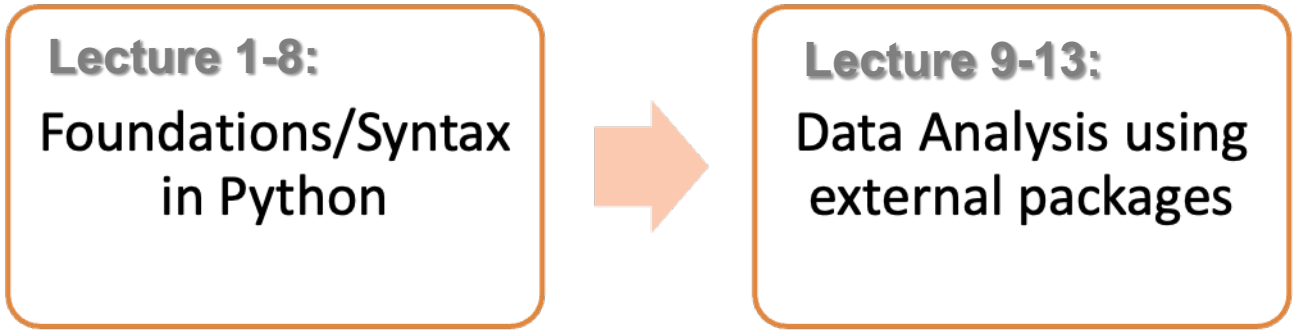
This course

Given the practical orientation of this course, we will develop data analytics skills in one of the leading software for data analytics nowadays: ***Python***

This course is a **fast-paced, intensive** class to help you learn the basics of programming for data analytics, and how to program in Python to solve real-world problems.

Course topics

Programming for Data Science in Business



This course

Instructor: Aida Sanatizadeh

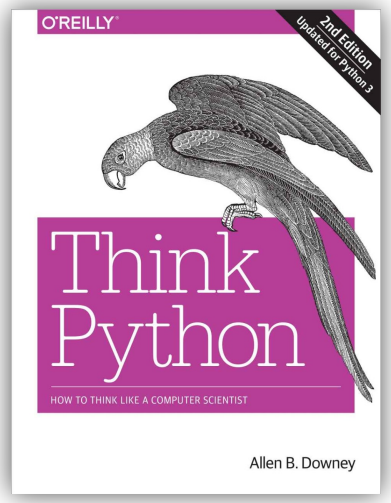
Meeting time: Thursday 6:30:pm-9:00pm

Teaching Assistant: Dharmamithra Tirunagari

Meeting time (labs & office hours): Monday 10 AM- 11AM

Recommended textbook

Think Python—How to Think Like a Computer Scientist, by Allen B. Downey, (1st edition for Python 2, 2nd edition for Python 3). You can download it here or read it in HTML.



This course

Who are recommended to take this course?

- Not familiar with Python and want to learn the foundations of Python programming and basic knowledge of data analyzing. **YES**
- Have rich experiences of using Python. **NO**
- Want to learn detailed/advanced machine learning models or algorithms. **NO**

This course

Prerequisites

- While programming experience is not necessary, entry-level experience will be helpful.
- **Tremendous after-class practice is needed (at least 8 hours per week), especially for those with limited prior programming experience.**

Piazza

We will use the Piazza (you can log into this system through Blackboard) for a discussion board and for posting updates. You are highly encouraged to post constructive, relevant, non-redundant posts. Your questions/answers could benefit other people.

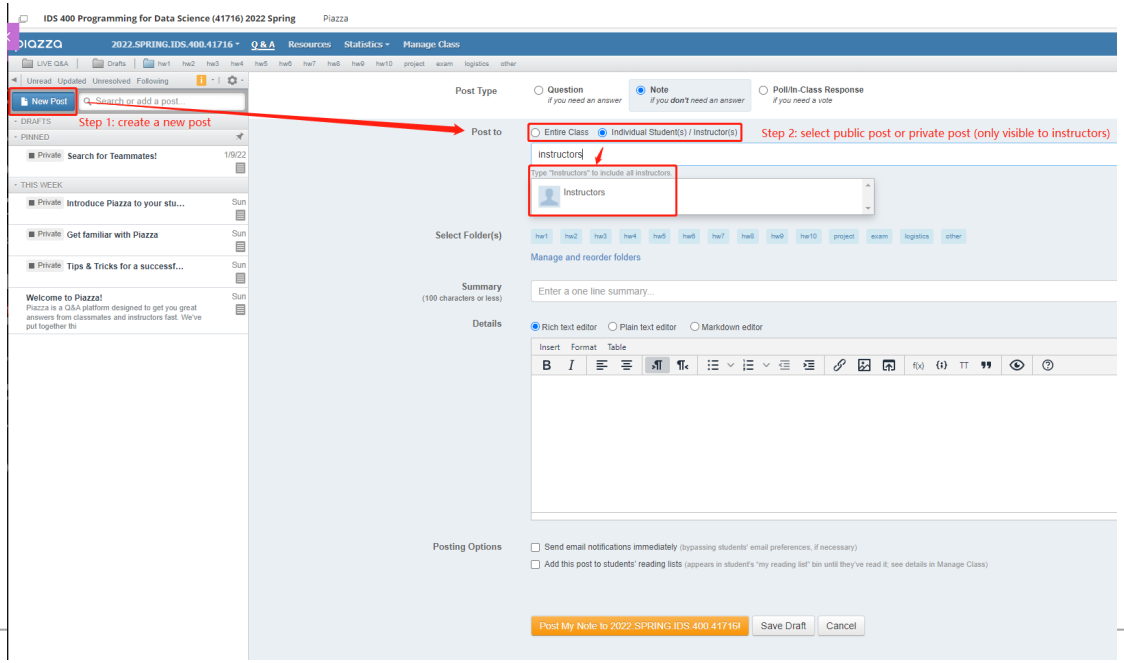
- Good question and elicit good answers
- Correct and substantive answer
- Correction to substantive things that I got wrong

Actively discussion on Piazza will be counted as ***bonus attendance score***. You can only earn Piazza points before the final project presentation. So, I strongly encourage you to start contributing as early as possible.

Q&A and contact Policy

- Everything related to class (Discussion/ Questions/Answers) → **Piazza only**
- Private concern relevant/irrelevant to class → **Piazza private post**
- **All questions will be answered by the teaching staff.**

How to create a public/private post:



Evaluation: Assignments

- 6/7 assignments, weekly
- All assignments are hands-on programming tasks. You are allowed to discuss with other students (up to three) or the teaching staff. *Please put all the names of students that you discussed with.* However individual students *must write their own solutions*. Copying a program, or letting someone else copy your program, is a form of academic dishonesty
- Each assignment will take about 2-6 hours on average.
- Late submission will receive credit deduction:

0-1 day	10%
1-2 days	20%
2-3 days	30%
3-5 days	50%
>5 days	will not accept

Lab sessions

Aim for:

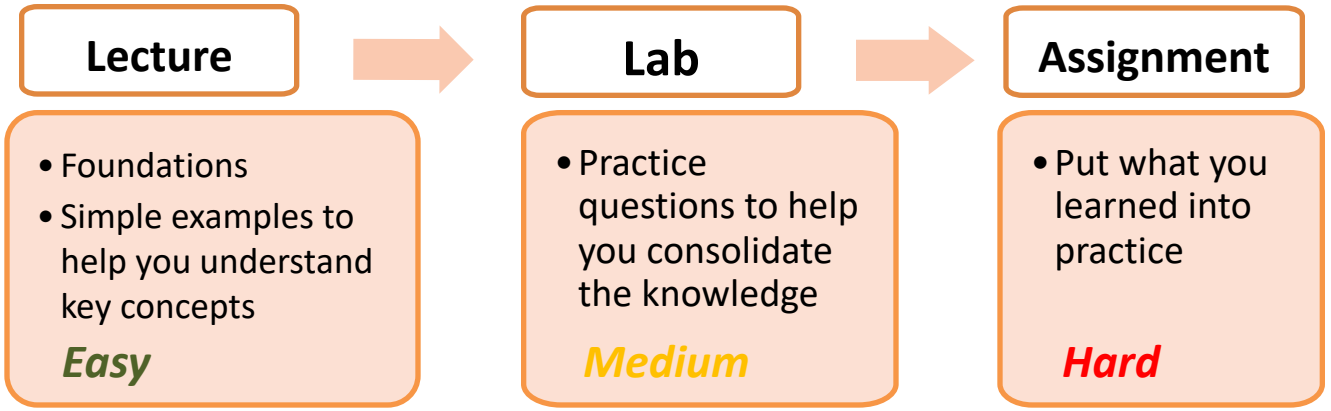
- Practice
- Consolidate the key knowledge points
- Find out the key knowledge points you missed/confused

Lab sessions

- 10 labs in total tentatively.
- Within each lab, you need to finish given tasks (usually posted immediately after lectures). Typical tasks are reviewing/practicing what we learned in the lectures for that week.
- You must submit at least 6 labs. Without submitting at least 6 labs, you will automatically fail this course.
- Lab submissions will not be graded.
- You can skip the most difficult task. But please make sure to post your questions/confusions on Piazza, so that you can have a discussion with your classmates, and I know what we need to recap.
- If you come up with a good solution, please DO post your good solutions on Piazza and share it with us!

FYI... You will find assignments challenging if you do not practice lab questions.

Practice chain



Evaluation: Quizzes

- ***Must attend***
- 2 quizzes
- Each quiz will take about 30-60 minutes.
- They will be closed book online tests via Blackboard.

Evaluation: Final Project

- A group project (***No more than 4 persons per group***). I will announce the group later.
- You customize your project. It has to be related to data analytics.
- There will be a **final presentation**.
- Every student is supposed to grade other groups' projects, and the final scores of projects are the combination of scores from your classmates and the teaching staff.
- You also need to submit a **project report** (due almost one week after the presentation)
- More information about project requirement will be provided

Attendance

- Lectures: ***must attend***.
- There might be some random attendance checks. 5 points deduction of attendance score if you are absent for all checks.
- Failing to attend quiz or final project presentation will receive an “F” for the course except extreme reasons.

Evaluation

Lecture attendance/Piazza	10%
Assignment	50%
Quiz	20%
Final project	20%

Points	Letter Grade
A	100-90
B	89.9-80
C	79.9-70
D	69.9-60
F	Below 60

Best way of learning coding

PRACTICE

Best way of learning coding

PRACTICE

PRACTICE

PRACTICE

.....

Let's start the Python journey...

Introduction to Python

Python is an interpreted, high-level, general-purpose programming language, created by Guido van Rossum and first released in 1991.[1]

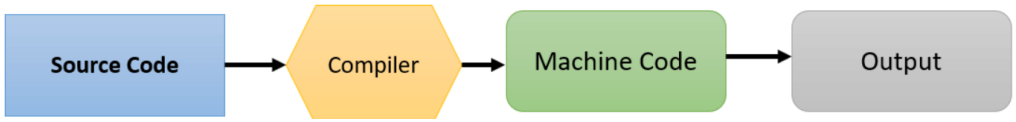
[1]https://en.wikipedia.org/wiki/Python_(programming_language)

Introduction to Python

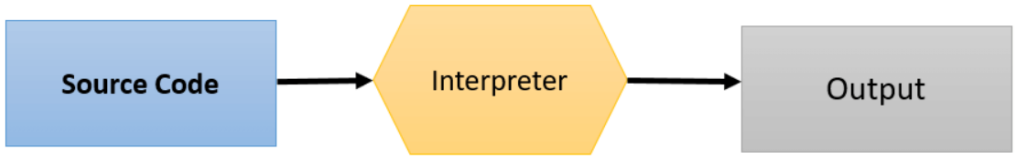
Interpreted -- Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.

- Not compiled like Java
- Code is written and then directly executed by an interpreter
- Type commands into interpreter and see immediate results

How Compiler Works:



How Interpreter Works:

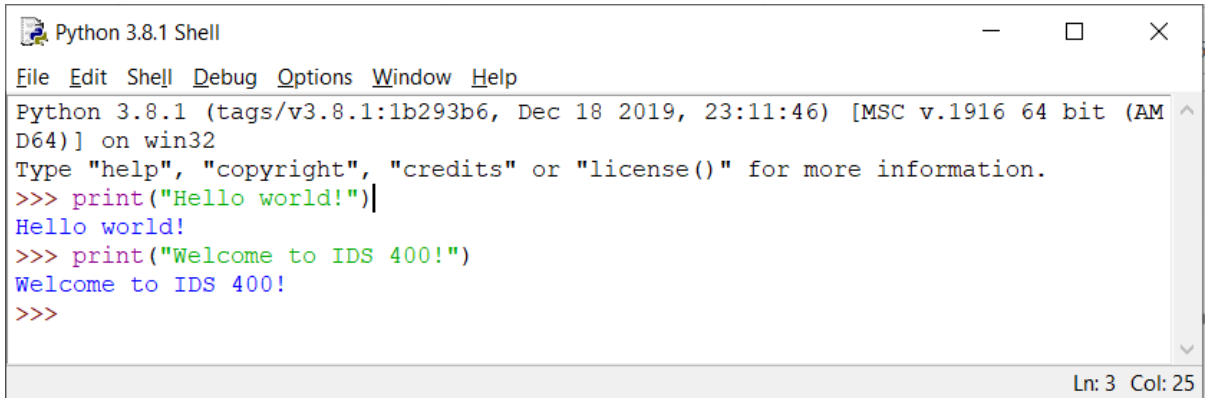


To learn more about the difference between Compiler and Interpreter: https://www.guru99.com/difference-compiler-vs-interpreter.html

Introduction to Python

Interactive -- You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

- Allows you to type commands one-at-a-time and see results
- A great way to explore Python's syntax



Introduction to Python

Object-Oriented -- Python supports Object- Oriented style or technique of programming that encapsulates code within objects. (We will NOT cover this in this course).

Beginner's Language -- Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing, numerical computing, web browsers to games.

To learn more about Object-Oriented Programming: https://www.youtube.com/watch?v=pTB0EiLXUC8

Why Python?

Easy-to-learn: Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.

Easy-to-read: Python code is more clearly defined and visible to the eyes.

Easy-to-maintain: Python's source code is fairly easy-to-maintain.

A broad standard library: Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.

Interactive mode: Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.

Portable: Python can run on a wide variety of hardware platforms and has the same interface on all platforms.

Extendable: You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.

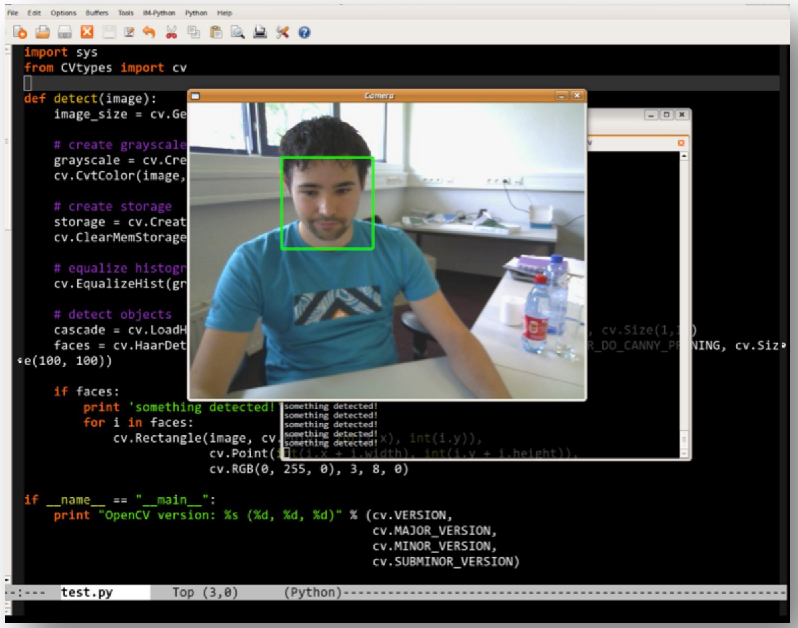
Databases: Python provides interfaces to all major commercial databases.

GUI Programming: Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems.

Scalable: Python provides a better structure and support for large programs than shell scripting.

What Python can do?

- With Python and OpenCV, we can do face detection.



What Python can do?

- With Python and OpenCV, we can do face detection.
- With BeautifulSoup and urllib, we can write a Python script to get the winners of the Food and Drink categories in the Chicago Readers' Best of 2017 list.

```python
from bs4 import BeautifulSoup
from urllib2 import urlopen
from time import sleep # be nice

BASE_URL = "http://www.chicagoreader.com"

def make_soup(url):
    html = urlopen(url).read()
    return BeautifulSoup(html, "lxml")

def get_category_links(section_url):
    soup = make_soup(section_url)
    boccat = soup.find("dl", "boccat")
    category_links = [BASE_URL + dd.a["href"] for dd in boccat.findAll("dd")]
    return category_links

def get_category_winner(category_url):
    soup = make_soup(category_url)
    category = soup.find("h1", "headline").string
    winner = [h2.string for h2 in soup.findAll("h2", "boc1")]
    runners_up = [h2.string for h2 in soup.findAll("h2", "boc2")]
    return {"category": category,
            "category_url": category_url,
            "winner": winner,
            "runners_up": runners_up}

if __name__ == '__main__':
    food_n_drink = ("http://www.chicagoreader.com/chicago/"
                    "best-of-chicago-2011-food-drink/BestOf?oid=4106228")

    categories = get_category_links(food_n_drink)

    data = [] # a list to store our dictionaries
    for category in categories:
        winner = get_category_winner(category)
        data.append(winner)
        sleep(1) # be nice

    print data
```

Best of Chicago 2017: Food & Drink

click to enlarge

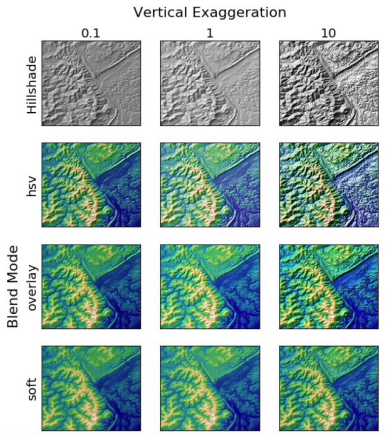
What Python can do?

- With Python and OpenCV, we can do face detection.
- With BeautifulSoup and urllib, we can write a Python script to get the winners of the Food and Drink categories in the Chicago Readers' Best of 2017 list.
- With MySQLdb, we can write a Python script to all records in the coworker database.

```
#!/usr/bin/python

import MySQLdb

db = MySQLdb.connect(host="localhost", port=3306, user="foo", passwd="bar", db="qoz")
cursor = db.cursor()
cursor.execute("SELECT name, phone_number FROM coworkers")
data = cursor.fetchall()
for row in data :
    print row[0],",",row[1]
```

What Python can do?

- With Python and OpenCV, we can do face detection.
- With BeautifulSoup and urllib, we can write a Python script to get the winners of the Food and Drink categories in the Chicago Readers' Best of 2017 list.
- With MySQLdb, we can write a Python script to all records in the coworker database.
- With matplotlib, we can make beautiful plots.

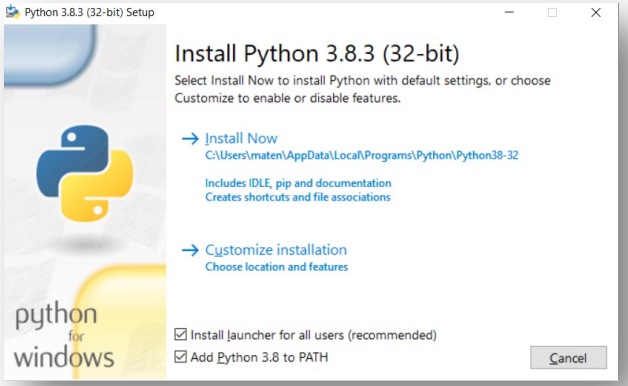



```
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.cbook import get_sample_data
from matplotlib.colors import LightSource

dem = np.load(get_sample_data('jacksboro_fault_dem.npz'))
z = dem['elevation']

#-- Optional dx and dy for accurate vertical exaggeration ----------------
# If you need topographically accurate vertical exaggeration, or you don't want
# to guess at what *vert_exag* should be, you'll need to specify the cellsize
# of the grid (i.e. the *dx* and *dy* parameters).  Otherwise, any *vert_exag*
# value you specify will be realitive to the grid spacing of your input data
# (in other words, *dx* and *dy* default to 1.0, and *vert_exag* is calculated
# relative to those parameters).  Similarly, *dx* and *dy* are assumed to be in
# the same units as your input z-values.  Therefore, we'll need to convert the
# given dx and dy from decimal degrees to meters.
dx, dy = dem['dx'], dem['dy']
dy = 111200 * dy
dx = 111200 * dx * np.cos(np.radians(dem['ymin']))
#------------------------------------------------------------------------

# Shade from the northwest, with the sun 45 degrees from horizontal
ls = LightSource(azdeg=315, altdeg=45)
cmap = plt.cm.gist_earth

fig, axes = plt.subplots(nrows=4, ncols=3, figsize=(8, 9))
plt.setp(axes.flat, xticks=[], yticks=[])

# Vary vertical exaggeration and blend mode and plot all combinations
for col, ve in zip(axes.T, [0.1, 1, 10]):
    # Show the hillshade intensity image in the first row
    col[0].imshow(ls.hillshade(z, vert_exag=ve, dx=dx, dy=dy), cmap='gray')

    # Place hillshaded plots with different blend modes in the rest of the rows
    for ax, mode in zip(col[1:], ['hsv', 'overlay', 'soft']):
        rgb = ls.shade(z, cmap=cmap, blend_mode=mode,
                       vert_exag=ve, dx=dx, dy=dy)
        ax.imshow(rgb)

# Label rows and columns
for ax, ve in zip(axes[0], [0.1, 1, 10]):
    ax.set_title('{0}'.format(ve), size=18)
for ax, mode in zip(axes[:, 0], ['Hillshade', 'hsv', 'overlay', 'soft']):
    ax.set_ylabel(mode, size=18)

# Group labels...
axes[0, 1].annotate('Vertical Exaggeration', (0.5, 1), xytext=(0, 30),
                    textcoords='offset points', xycoords='axes fraction',
                    ha='center', va='bottom', size=20)
axes[2, 0].annotate('Blend Mode', (0, 0.5), xytext=(-30, 0),
                    textcoords='offset points', xycoords='axes fraction',
                    ha='right', va='center', size=20, rotation=90)
fig.subplots_adjust(bottom=0.05, right=0.95)

plt.show()
```

What Python can do?

- With Python and OpenCV, we can do face detection.
- With BeautifulSoup and urllib, we can write a Python script to get the winners of the Food and Drink categories in the Chicago Readers' Best of 2017 list.
- With MySQLdb, we can write a Python script to all records in the coworker database.
- With matplotlib, we can make beautiful plots.
- With NLTK, we can do many language processing operations.
- ……

Install Python



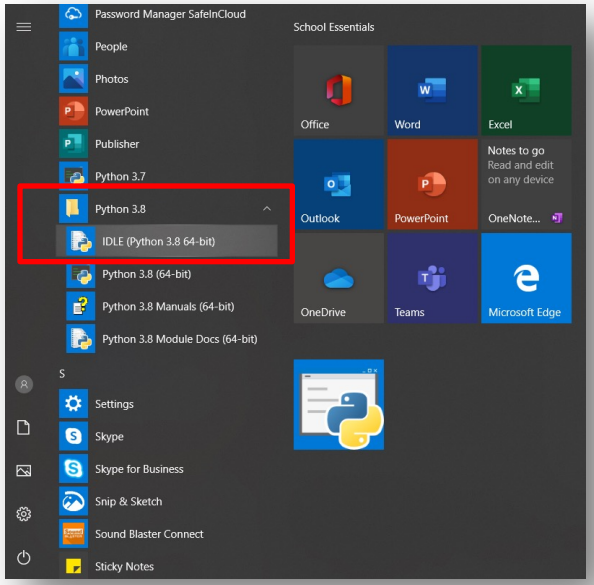
You can install Python from Python.org

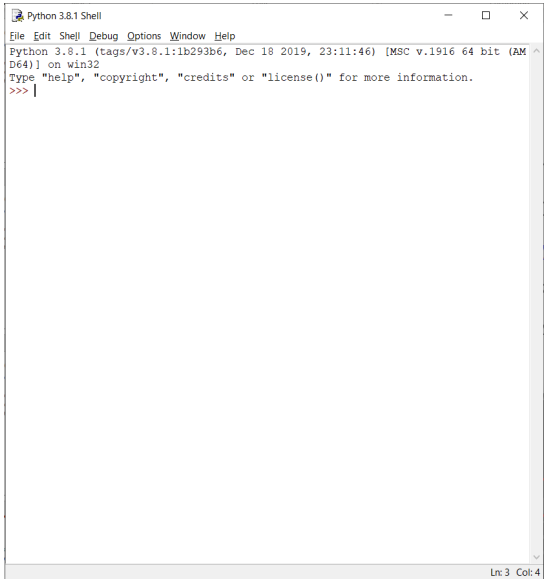
Step 1 Download the Python 3 Installer from

https://www.python.org/downloads/

Step 2 Run the Installer

Step 3 After the installation is complete, close the installer and select Start->Python 3.8->IDLE (Python GUI).





Jupyter Notebook through Anaconda[1]

We will use Jupyter Notebook for this course

Why Jupyter Notebook?

- Run in your browser.
- Evaluate code blocks and show the output right in the same document.
- Everything lives in memory – work with results from other code blocks.
- Integrates Markdown + code for real notebook-style documents.
- Easy to share online.
- Large community support and many extensions.

Installing Anaconda on Windows

- Visit Anaconda.com/downloads
- Select Windows/mac
- Download the .exe installer
- Open and run the .exe installer
- Open the Anaconda Prompt and run some Python code

[1] anaconda installation: https://problemsolvingwithpython.com/01-Orientation/01.03-Installing-Anaconda-on-Windows/

Install Python

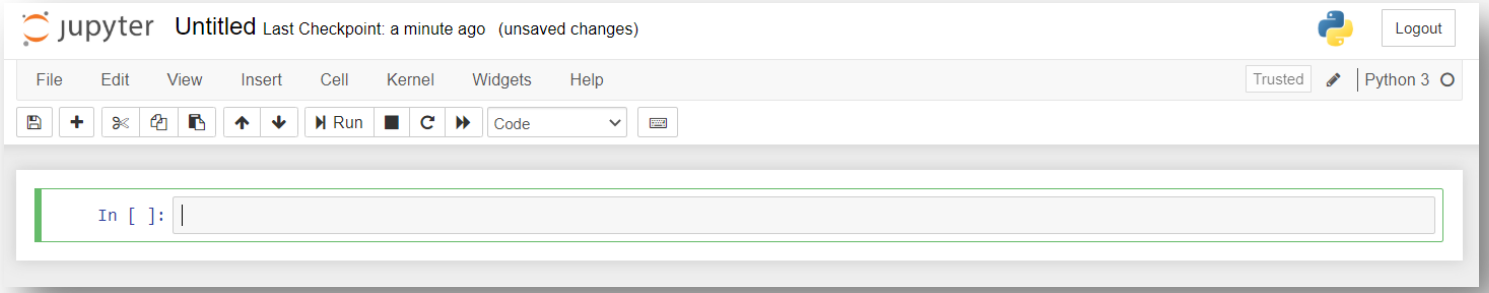
In addition, other Python IDEs:

- PyCharm
- Eclipse
- Spyder
- ….

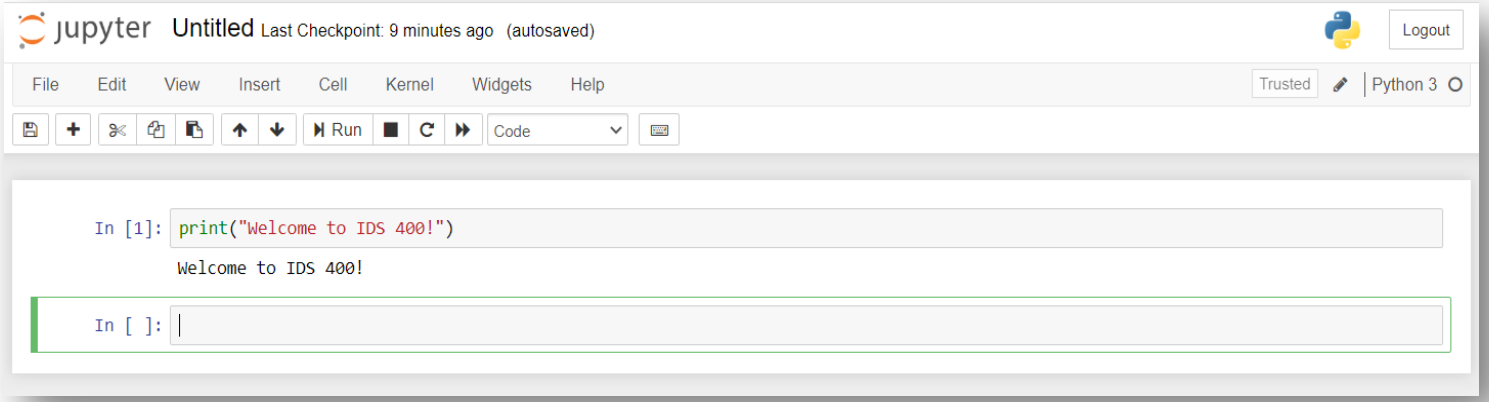
A convenient online “Jupyter”: Google Colab

First Python program

Jupyter interface

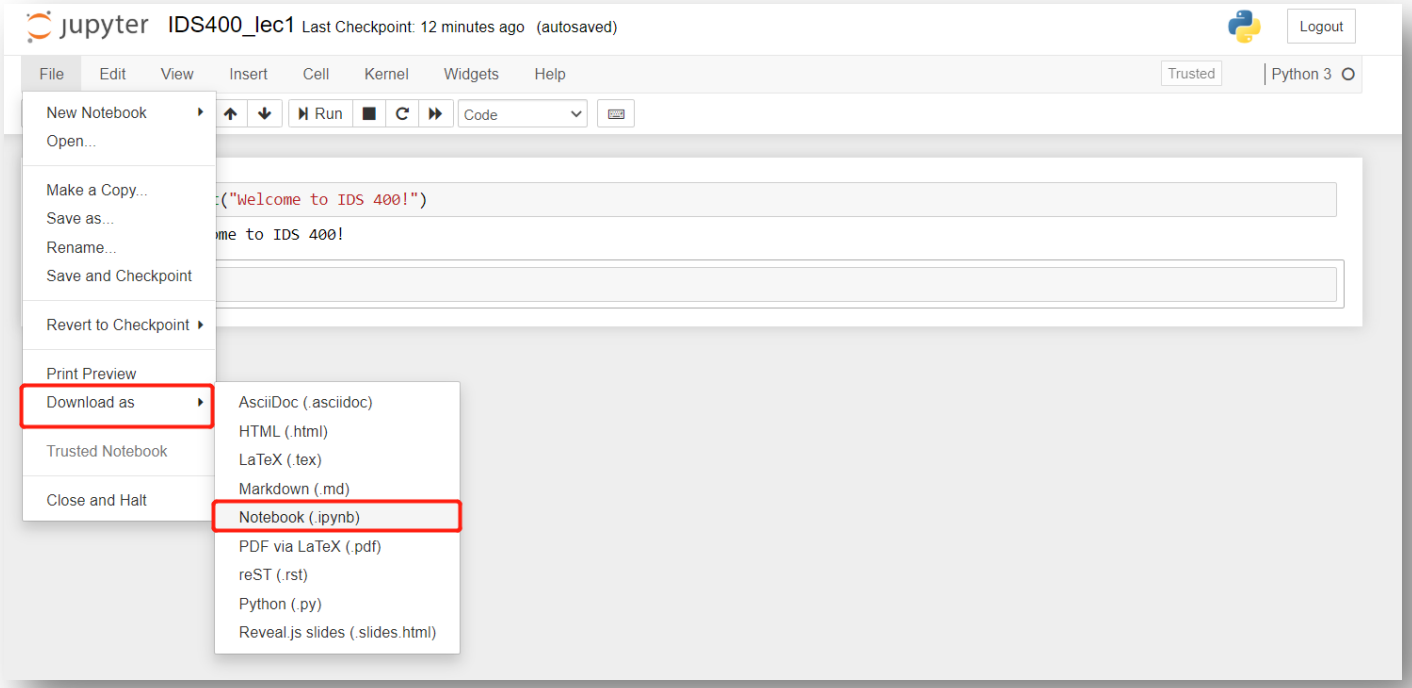


This is what you see after you click the Run button



First Python program

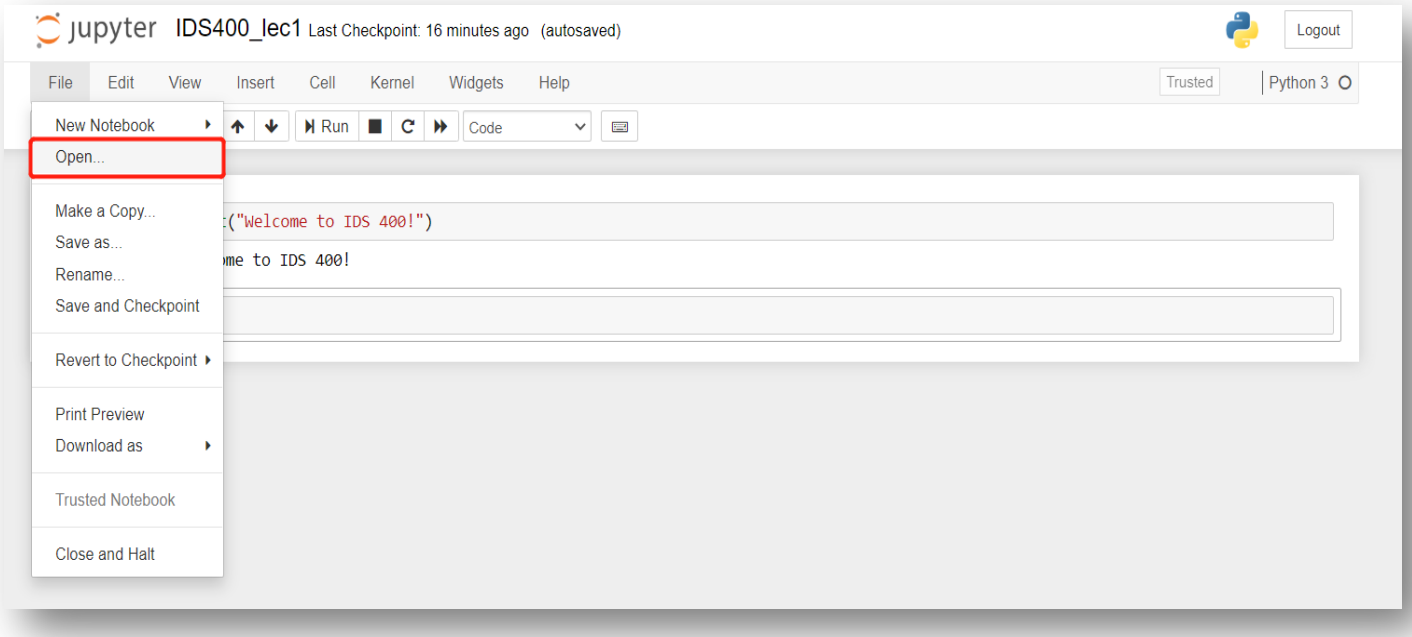
Save script



Note: All your code submissions should be in the format of .ipynb and .PDF
All submissions should show the output of your code

First Python program

If you want to open an ipynb file:



The print statement

The syntax of print() is:

```
print(*objects, sep=' ', end='\n')
```

print() Parameters:

- **objects** - object to the printed. * indicates that there may be more than one object.
- **sep** - objects are separated by sep. Default value: ' '
- **end** - end is printed at last.

The print statement

Print a string

- String needs to be surrounded by double “ or single ‘ quotes.
- As far as language syntax is concerned, there is no difference in single or double quoted string. Both representations can be used interchangeably.

```
print("text")
```

```
text
```

```
print('text')
```

```
text
```

```
a = 5
print("a =", a, sep='00000', end='\n\n\n')
print("a =", a, sep='0', end='')
```

```
a =000005


a =05
```

Question: What if we want to print **Don’t do this** ?

The print statement

Question: What if we want to print **Don’t do this** ?

A
```
print('Don't do this')
```

B
```
print("Don't do this")
```

C
```
None of them
```

The print statement

Question: What if we want to print **Don't do this** ?

A

```
print('Don't do this')
```

B

```
print("Don't do this")
```

√

C

```
None of them
```

```
print('Don't do this')
```

```
  File "<ipython-input-1-e7bfce1cac87>", line 1
    print('Don't do this')
                ^
SyntaxError: invalid syntax
```

```
print("Don't do this")
```

```
Don't do this
```

The print statement

- In Python strings, the backslash \ is a special character, also called the "escape" character. It is used in representing certain whitespace characters.

"\t" is a tab, "\n" is a newline, and "\r" is a carriage return.

```
print('apple\torange')
```

```
apple	orange
```

```
print('apple\norange')
```

```
apple
orange
```


What is a carriage return: https://www.pythonpool.com/carriage-return-python/

The print statement

- Conversely, prefixing a special character with "\" turns it into an ordinary character. This is called "escaping".

```
print('It\'s raining')
```

```
It's raining
```

```
print('\"It\'s raining\"')    # Same string specified differently
```

```
'It's raining'
```

```
print("\"hello\"")
```

```
"hello"
```

```
print('"\\" is the backslash')
```

```
"\" is the backslash
```

```
print('"//" is the backslash')
```

```
"//" is the backslash
```

Comments

The syntax of comments:

#comment text (one line)

- Comments in Python begin with a hash mark (#)
- Comments are in the source code for humans to read, NOT for computers to execute.

```
#IDS400, summer 2020
#This is a comment.
print("Hello,world!")
print("")  #blankline
print("IDS400, summer 2020")
```

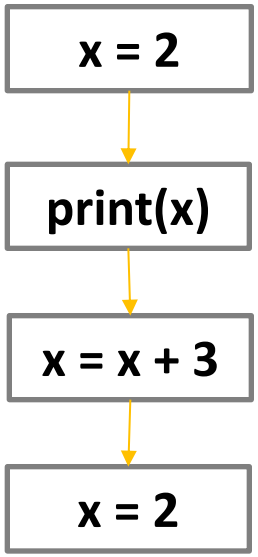
```
Hello,world!

IDS400, summer 2020
```

Program steps or flow

- Like a recipe or installation instructions, a program is a **sequence** of steps to be done in order.
- Some steps are **conditional** -they maybe skipped.
- Sometimes a step or group of steps are to be **repeated**.
- Sometimes we store a set of steps to be used over and over as needed several places throughout the program.

Sequential steps

```
In [19]: x = 2

In [20]: print(x)
         2

In [21]: x = x + 3

In [22]: print(x)
         5
```

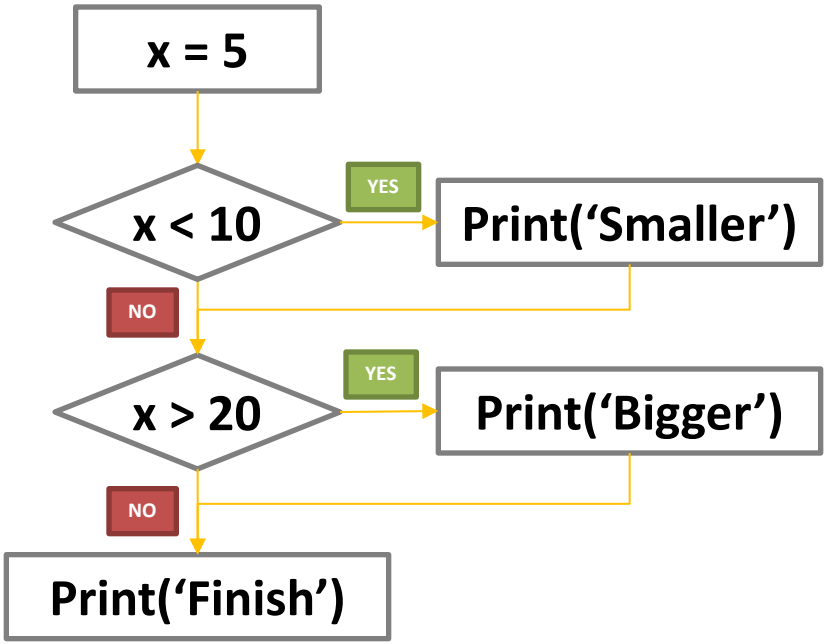


When a program is running, it flows from one step to the next. As programmers, we set up “paths” for the program to follow.

Conditional steps

```
x =5
if x <10:
    print('Smaller')
if x >20:
    print('Bigger')
print('Finish')
```

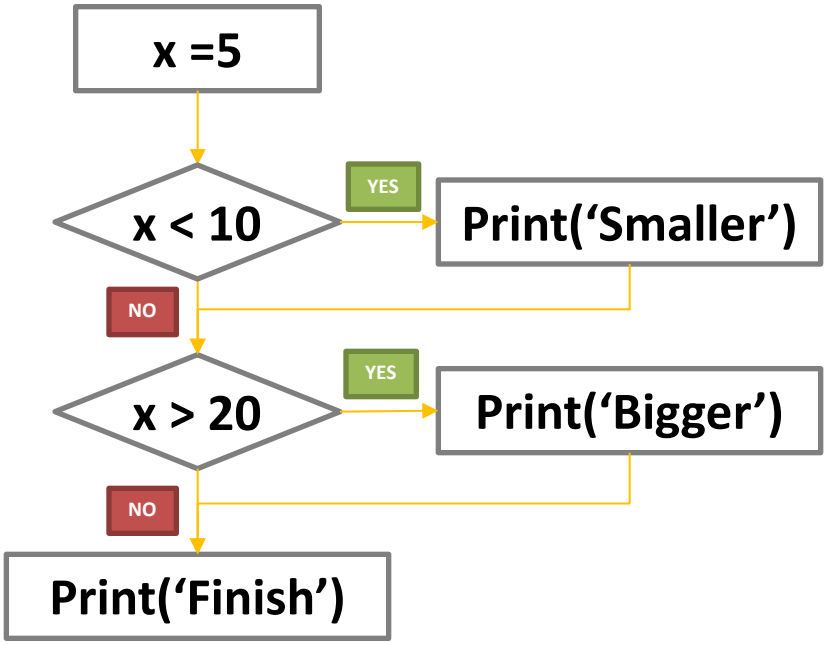
```
Smaller
Finish
```



Conditional steps

```
x =5
if x <10:
    print('Smaller')
if x >20:
    print('Bigger')
print('Finish')
```

```
Smaller
Finish
```

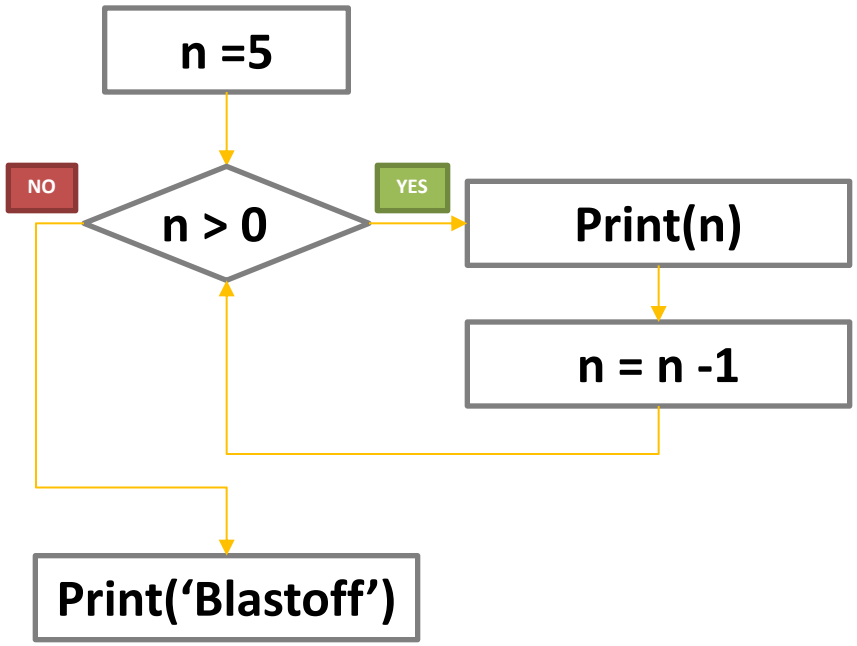


In a Python program, the ***if*** statement (conditional steps) is how you perform a decision-making. It allows for conditional execution of a statement or group of statements based on the value of an expression.

Repeated steps

```
n =5
while n > 0:
    print(n)
    n = n - 1
print('Blastoff!')
```

```
5
4
3
2
1
Blastoff!
```



Loops (repeated steps) have iteration variables that change each time through a loop.

Often these iteration variables go through a sequence of numbers.

A word count example

```
name = raw_input('Enterfile:')
handle = open(name,'r')
text = handle.read()
words = text.split()

counts = dict()
for counts word in words:
    counts[word] = counts.get(word,0) +1
bigcount = None
bigword = None

for word,count in counts.items():
    if bigcount is None or count > bigcount:
        bigword = word
        bigcount = count
print bigword, bigcount
```

Sequential

Repeated

Conditional

In practice, we usually combine different types of steps together to do specific tasks.

Environment Setup

- Install Jupyter Notebook, create your first Python program and get familiar with Jupyter coding environment.

Python installation

- Install Anaconda based on your computer OS and follow instructions in the link.

  https://problemsolvingwithpython.com/01-Orientation/01.03-Installing-Anaconda-on-Windows/
- Open Anaconda navigator and launch Jupyter Notebook.
- You are ready to run codes.

To do

- Practice the followings:
  - Install Anaconda and launch Jupyter Notebook.
  - Create/ open/ download a .ipynb file.
  - Change the name of the file.
  - Run cells.
  - Explore the menus, *e.g., File, Edit, View*…
  - Create headings for your cells (change cell types).
  - Practice print statement, escape characters and comments.
- Here is a helpful link: https://realpython.com/jupyter-notebook-introduction/