



# Lecture 1-2: Programming Environment

Dr. James Chen, Animal Data Scientist, School of Animal Sciences

2023 APSC-5984 SS: Agriculture Data Science

# What Is a Programming Environment?

**I. Why Python?**

**II. Jupyter Notebook**

**III. Git and GitHub**

**IV. What is Unix shell?**

**V. Integrated Development Environment (IDE)**

## I. Why Python?

### 1. Interpreted vs. Compiled language



#### Interpreted language:

- **Examples: Python, Perl, R, Matlab**
- The interpreter executes the codes **line by line**.
- Faster **development process** as you can validate the results without compiling the entire program.
- Suitable for exploring new ideas or developing lightweight applications

## I. Why Python?

### 1. Interpreted vs. Compiled language



#### Interpreted language:

- **Examples: Python, Perl, R, Matlab**
- The interpreter executes the codes **line by line**.
- Faster **development process** as you can validate the results without compiling the entire program.
- Suitable for exploring new ideas or developing lightweight applications

#### Compiled language:

- **Examples: C/C++, Java, Fortran, Visual Basic**
- The **entire program** will be compiled and built before the execution.
- Faster **execution time**.
- Suitable for developing large and complicated applications

## I. Why Python?

### 2. Dynamic vs. Static typing

**Python** is a **dynamically-typed language**: A Python variable DOES NOT have a static type. A data type refers to an integer, a string (text), or a floating number.



## I. Why Python?

### 2. Dynamic vs. Static typing

**Python** is a **dynamically-typed language**: A Python variable DOES NOT have a static type. A data type refers to an integer, a string (text), or a floating number.



#### Pros:

Higher flexibility for development

#### Cons:

Slower and less efficient (memory-wise) as the interpreter has to determine the type of a variable at runtime

## I. Why Python?

### 2. Dynamic vs. Static typing

**Python** is a **dynamically-typed language**: A Python variable DOES NOT have a static type. A data type refers to an integer, a string (text), or a floating number.

#### Pros:

Higher flexibility for development

#### Cons:

Slower and less efficient (memory-wise) as the interpreter has to determine the type of a variable at runtime

```
x = 3
print(x) # 3
x = "hello"
print(x) # hello
```

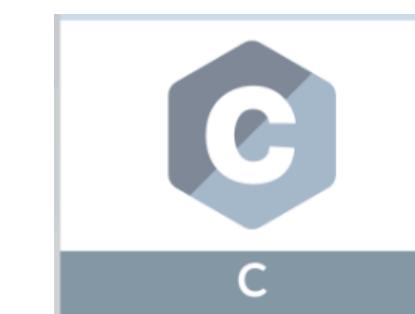
## I. Why Python?

### 2. Dynamic vs. Static typing



Python is a dynamically-typed language

```
1 < def main():
2     x = 5
3     print("x is %d" % x)
4     return 0
```



C is a statically-typed language



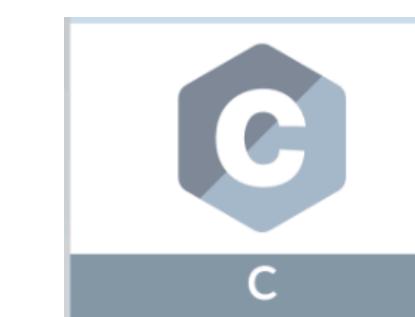
## I. Why Python?

### 2. Dynamic vs. Static typing



Python is a dynamically-typed language

```
1 ~ def main():
2     x = 5
3     print("x is %d" % x)
4     return 0
```



C is a statically-typed language

```
1 def main() -> int:
2     x: int = 5
3     print("x is %d" % x)
4     return 0
```

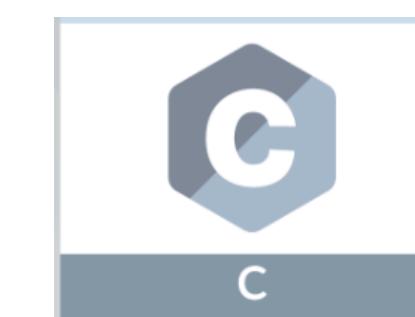


## I. Why Python?

### 2. Dynamic vs. Static typing



Python is a dynamically-typed language



C is a statically-typed language

```
1 ~ def main():
2     x = 5
3     print("x is %d" % x)
4     return 0
```

The two statements are equivalent. The typing information is only an annotation to humans

```
1  def main() → int: → Indicate that the function
2  x: int = 5           will return an integer
3  print("x is %d" % x)
4  return 0
```

## I. Why Python?

### 2. Dynamic vs. Static typing

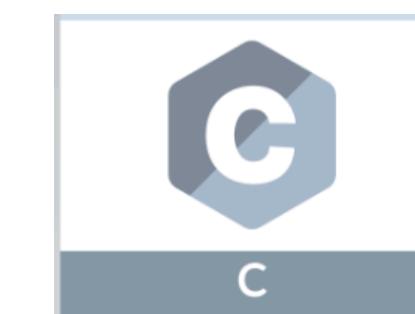


Python is a dynamically-typed language

```
1 ~ def main():
2     x = 5
3     print("x is %d" % x)
4     return 0
```

The two statements are equivalent. The typing information is only an annotation to humans

```
1  def main() → int: → Indicate that the function
2      x: int = 5    will return an integer
3      print("x is %d" % x)
4      return 0
```



C is a statically-typed language

```
1 #include <stdio.h>
2
3 int main() {
4     int x = 5;
5     printf("x = %d", x);
6
7 }
```

## I. Why Python?

### 2. Dynamic vs. Static typing

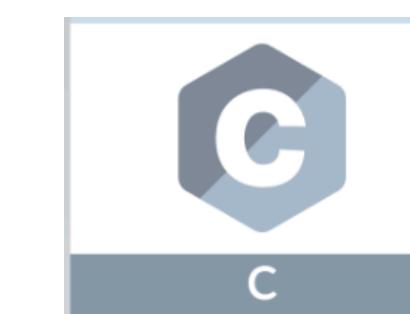


Python is a dynamically-typed language

```
1 def main():
2     x = 5
3     print("x is %d" % x)
4     return 0
```

The two statements are equivalent. The typing information is only an annotation to humans

```
1 def main() → int: → Indicate that the function
2     x: int = 5    will return an integer
3     print("x is %d" % x)
4     return 0
```



C is a statically-typed language

```
1 #include <stdio.h>
2
3 int main() {
4     int x = 5; → the main() function MUST return an integer value
5     printf("x = %d", x);
6
7 }
```

int: specify the variable type as an integer

### I. Why Python?

#### 3. Large and active community



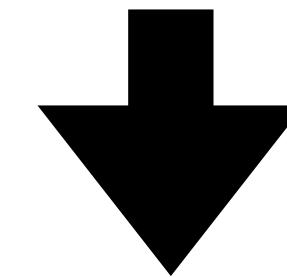
Rich resources are available to help you with your problems

### I. Why Python?

#### 3. Large and active community



Rich resources are available to help you with your problems



Wide range of libraries and frameworks

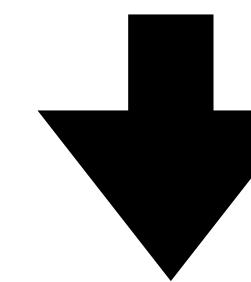
So that you don't need to code everything yourself

## I. Why Python?

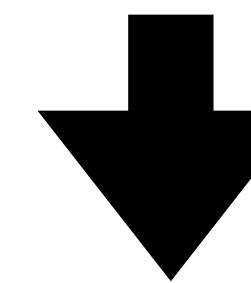
### 3. Large and active community



Rich resources are available to help you with your problems



Wide range of libraries and frameworks



So that you don't need to code everything yourself

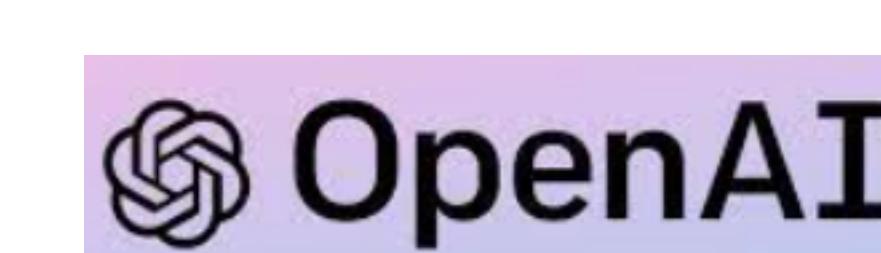
Scientific computation



Visualization



Machine learning

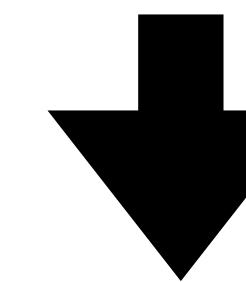


## I. Why Python?

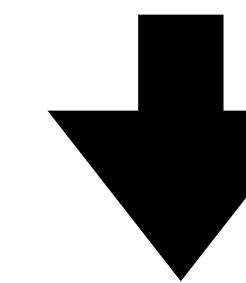
### 3. Large and active community



Rich resources are available to help you with your problems



Wide range of libraries and frameworks



So that you don't need to code everything yourself

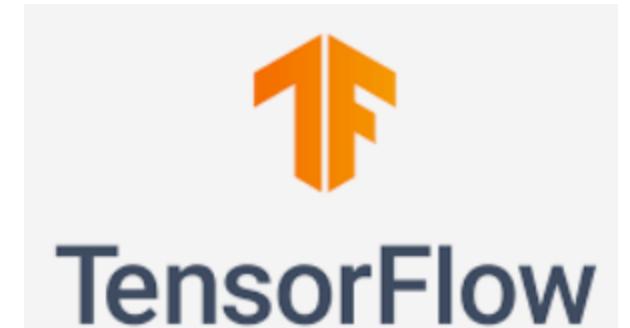
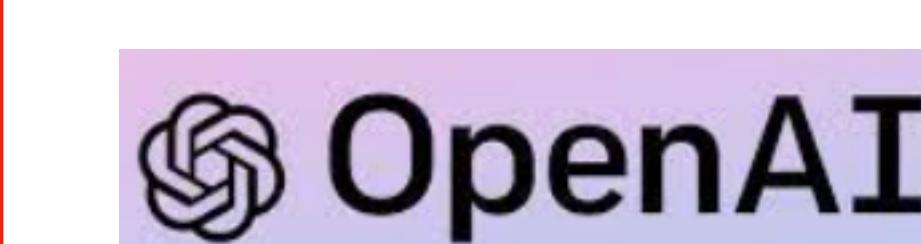
Scientific computation



Visualization



Machine learning



## I. Why Python?

### 3. Large and active community



<https://numpy.org/doc/stable/index.html>



<https://pandas.pydata.org/docs/index.html>



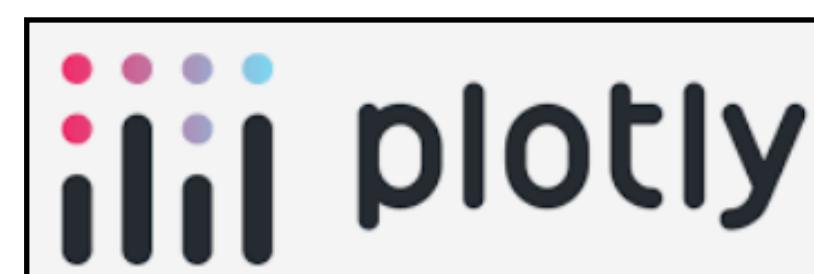
<https://docs.scipy.org/doc/scipy/index.html>



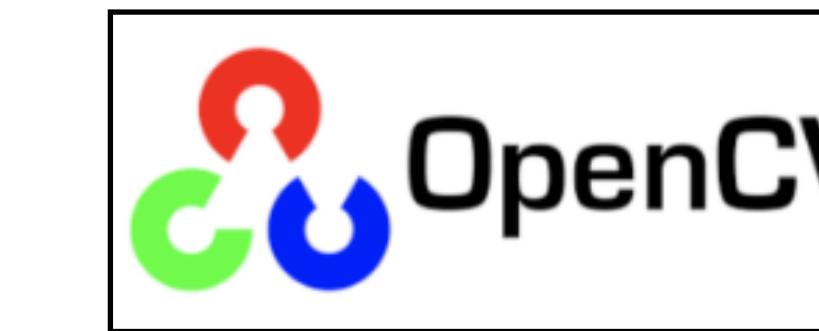
<https://scikit-learn.org/stable/index.html>



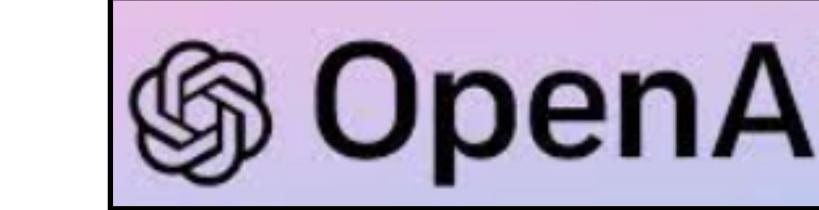
<https://bokeh.org/>



<https://plotly.com/>



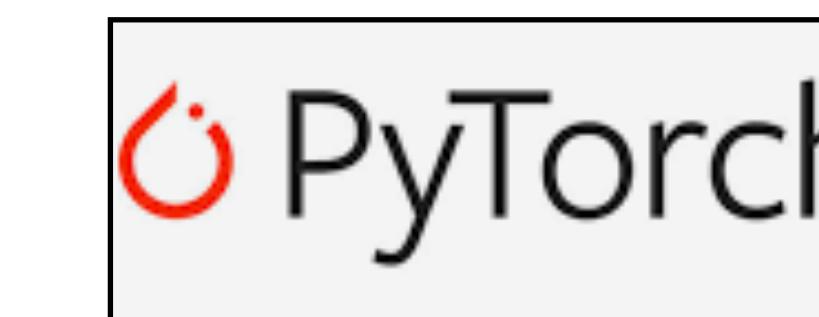
<https://opencv.org/>



<https://openai.com/>



<https://www.tensorflow.org/>



<https://pytorch.org/>

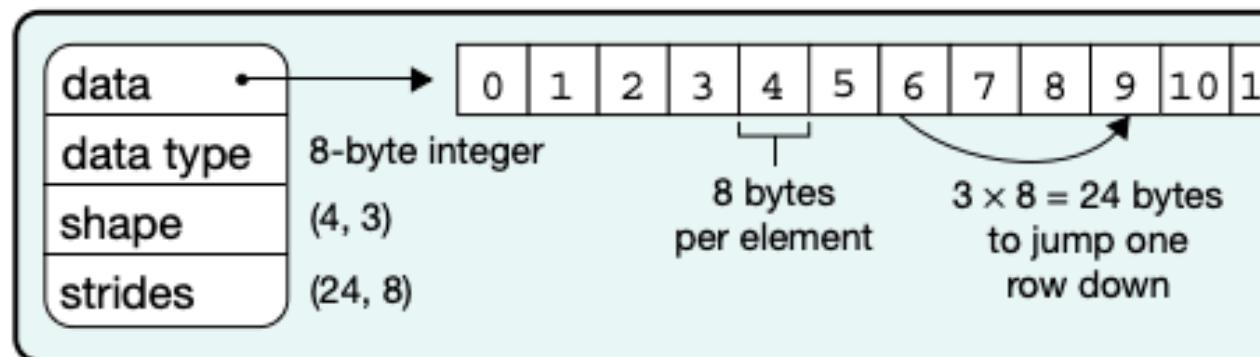
# Lecture 1-2: Programming environment

## I. Why Python?



### a Data structure

```
x = [[0, 1, 2],  
     [3, 4, 5],  
     [6, 7, 8],  
     [9, 10, 11]]
```



### b Indexing (view)

```
x[:, 1:] → [[0, 1, 2],  
             [3, 4, 5],  
             [6, 7, 8],  
             [9, 10, 11]]
```

with slices

```
x[:, :, ::2] → [[0, 1, 2],  
                 [3, 4, 5],  
                 [6, 7, 8],  
                 [9, 10, 11]]
```

Slices are **start:end:step**,  
any of which can be left blank

A diagram showing two 4x3 arrays. The first array has its second column highlighted. The second array has its second column and every second element in the third column highlighted. Arrows point from the slice notation to these highlighted areas.

with slices  
with steps

### c Indexing (copy)

```
x[1, 2] → 5 with scalars
```

```
x[[0, 1], [1, 2]] → [[x[0, 1], x[1, 2]]]
```

```
x[[1, 2], [1, 0]] → x[[1, 1], [1, 0], [2, 2], [1, 0]]
```

```
x[x > 9] → [10, 11] with masks
```

```
x[[1, 2], [1, 0]] → [1, 5] with arrays
```

```
x[[1, 2], [1, 0]] → x[[1, 1], [1, 0], [2, 2], [1, 0]] → [4, 3] with arrays  
with broadcasting
```

### d Vectorization

```
[[0, 1],  
 [3, 4],  
 [6, 7],  
 [9, 10]] + [[1, 1],  
              [1, 1],  
              [1, 1]] → [[1, 2],  
                          [4, 5],  
                          [7, 8],  
                          [10, 11]]
```

### e Broadcasting

```
[[0],  
 [3],  
 [6],  
 [9]] × [[1, 2],  
          [1, 2]] → [[0, 0],  
                      [3, 6],  
                      [6, 12],  
                      [9, 18]]
```

### f Reduction

```
[[0, 1, 2],  
 [3, 4, 5],  
 [6, 7, 8],  
 [9, 10, 11]] → sum axis 1 → [3, 12, 21, 30]  
 → sum axis 0 → 18, 22, 26 → sum axis (0,1) → 66
```

**Fig. 1 | The NumPy array incorporates several fundamental array concepts.**

**a**, The NumPy array data structure and its associated metadata fields.

**b**, Indexing an array with slices and steps. These operations return a ‘view’ of the original data. **c**, Indexing an array with masks, scalar coordinates or other arrays, so that it returns a ‘copy’ of the original data. In the bottom example, an array is indexed with other arrays; this broadcasts the indexing arguments

before performing the lookup. **d**, Vectorization efficiently applies operations to groups of elements. **e**, Broadcasting in the multiplication of two-dimensional arrays. **f**, Reduction operations act along one or more axes. In this example, an array is summed along select axes to produce a vector, or along two axes consecutively to produce a scalar. **g**, Example NumPy code, illustrating some of these concepts.

```
In [1]: import numpy as np
```

```
In [2]: x = np.arange(12)
```

```
In [3]: x = x.reshape(4, 3)
```

```
In [4]: x
```

```
Out[4]:
```

```
array([[ 0,  1,  2],  
      [ 3,  4,  5],  
      [ 6,  7,  8],  
      [ 9, 10, 11]])
```

```
In [5]: np.mean(x, axis=0)
```

```
Out[5]: array([4.5, 5.5, 6.5])
```

```
In [6]: x = x - np.mean(x, axis=0)
```

```
In [7]: x
```

```
Out[7]:
```

```
array([[-4.5, -4.5, -4.5],  
      [-1.5, -1.5, -1.5],  
      [ 1.5,  1.5,  1.5],  
      [ 4.5,  4.5,  4.5]])
```

# Lecture 1-2: Programming environment

## I. Why Python?



```
In [3]: titanic
Out[3]:
   PassengerId  Survived  Pclass  ...      Fare Cabin Embarked
0            1        0       3  ...    7.2500   NaN        S
1            2        1       1  ...  71.2833  C85        C
2            3        1       3  ...   7.9250   NaN        S
3            4        1       1  ...  53.1000  C123        S
4            5        0       3  ...   8.0500   NaN        S
..          ...
886         887        0       2  ...  13.0000   NaN        S
887         888        1       1  ...  30.0000  B42        S
888         889        0       3  ...  23.4500   NaN        S
889         890        1       1  ...  30.0000  C148        C
890         891        0       3  ...   7.7500   NaN        Q
[891 rows x 12 columns]
```

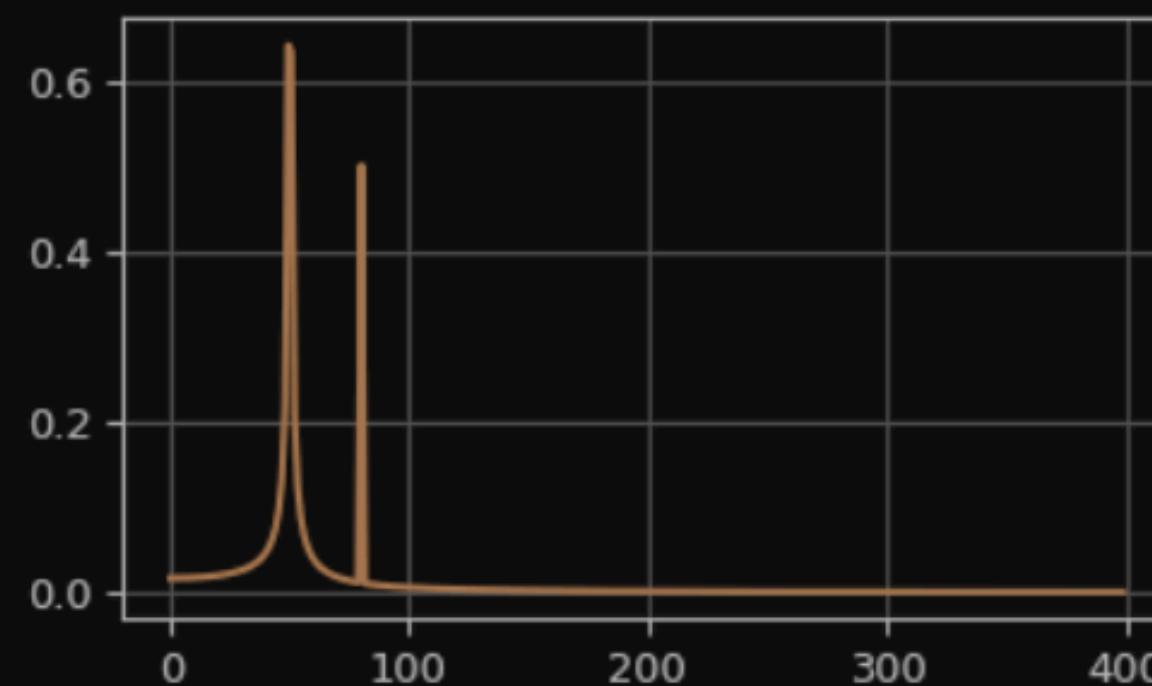
```
In [9]: titanic.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column      Non-Null Count  Dtype  
--- 
 0   PassengerId  891 non-null   int64  
 1   Survived     891 non-null   int64  
 2   Pclass       891 non-null   int64  
 3   Name         891 non-null   object  
 4   Sex          891 non-null   object  
 5   Age          714 non-null   float64 
 6   SibSp        891 non-null   int64  
 7   Parch        891 non-null   int64  
 8   Ticket       891 non-null   object  
 9   Fare          891 non-null   float64 
 10  Cabin        204 non-null   object  
 11  Embarked     889 non-null   object  
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
```

## I. Why Python?



### 1-D discrete Fourier transforms

```
>>> from scipy.fft import fft, fftfreq
>>> import numpy as np
>>> # Number of sample points
>>> N = 600
>>> # sample spacing
>>> T = 1.0 / 800.0
>>> x = np.linspace(0.0, N*T, N, endpoint=False)
>>> y = np.sin(50.0 * 2.0*np.pi*x) + 0.5*np.sin(80.0 * 2.0*np.pi*x)
>>> yf = fft(y)
>>> xf = fftfreq(N, T)[:N//2]
>>> import matplotlib.pyplot as plt
>>> plt.plot(xf, 2.0/N * np.abs(yf[0:N//2]))
>>> plt.grid()
>>> plt.show()
```



## Finding the inverse

The inverse of a matrix  $\mathbf{A}$  is the matrix  $\mathbf{B}$ , such that  $\mathbf{AB} = \mathbf{I}$ , where  $\mathbf{I}$  is the identity matrix consisting of ones down the main diagonal. Usually,  $\mathbf{B}$  is denoted  $\mathbf{B} = \mathbf{A}^{-1}$ . In SciPy, the matrix inverse of the NumPy array,  $\mathbf{A}$ , is obtained using `linalg.inv(A)`, or using `A.I` if  $\mathbf{A}$  is a Matrix. For example, let

$$\mathbf{A} = \begin{bmatrix} 1 & 3 & 5 \\ 2 & 5 & 1 \\ 2 & 3 & 8 \end{bmatrix},$$

then

$$\mathbf{A}^{-1} = \frac{1}{25} \begin{bmatrix} -37 & 9 & 22 \\ 14 & 2 & -9 \\ 4 & -3 & 1 \end{bmatrix} = \begin{bmatrix} -1.48 & 0.36 & 0.88 \\ 0.56 & 0.08 & -0.36 \\ 0.16 & -0.12 & 0.04 \end{bmatrix}.$$

The following example demonstrates this computation in SciPy

```
>>> import numpy as np
>>> from scipy import linalg
>>> A = np.array([[1,3,5],[2,5,1],[2,3,8]])
>>> A
array([[1, 3, 5],
       [2, 5, 1],
       [2, 3, 8]])
>>> linalg.inv(A)
array([[-1.48,  0.36,  0.88],
       [ 0.56,  0.08, -0.36],
       [ 0.16, -0.12,  0.04]])
>>> A.dot(linalg.inv(A)) #double check
array([[ 1.0000000e+00, -1.11022302e-16, -5.55111512e-17],
       [ 3.05311332e-16,  1.0000000e+00,  1.87350135e-16],
       [ 2.22044605e-16, -1.11022302e-16,  1.0000000e+00]])
```

## I. Why Python?

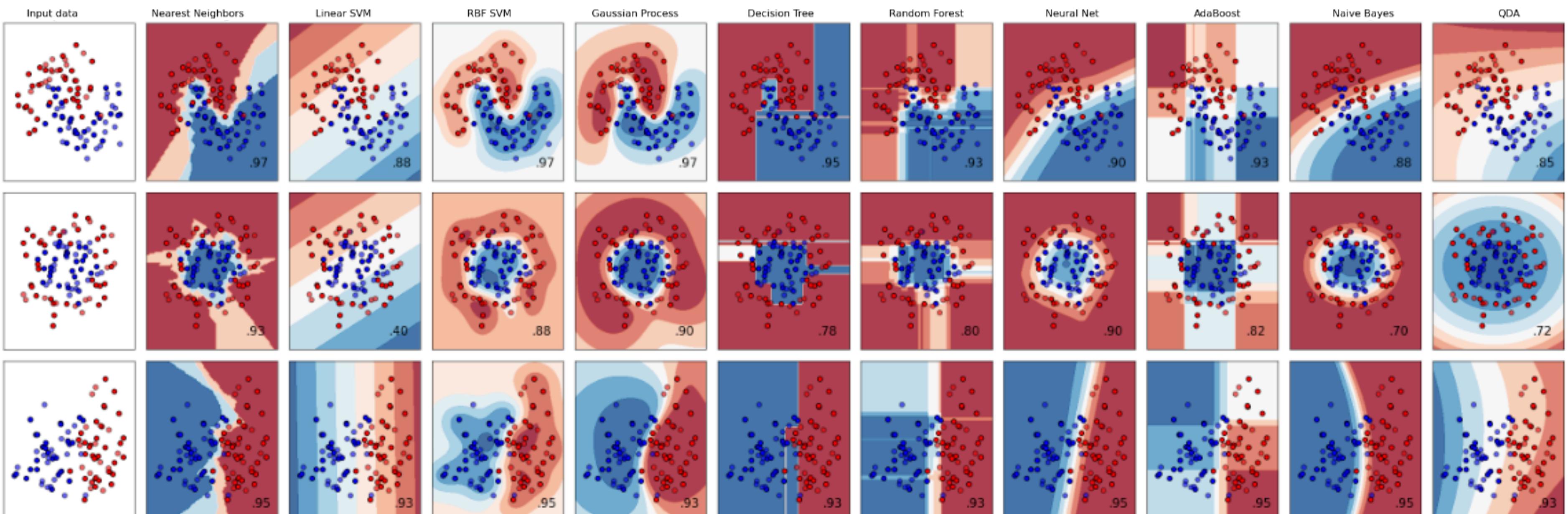


### Classifier comparison

A comparison of several classifiers in scikit-learn on synthetic datasets. The point of this example is to illustrate the nature of decision boundaries of different classifiers. This should be taken with a grain of salt, as the intuition conveyed by these examples does not necessarily carry over to real datasets.

Particularly in high-dimensional spaces, data can more easily be separated linearly and the simplicity of classifiers such as naive Bayes and linear SVMs might lead to better generalization than is achieved by other classifiers.

The plots show training points in solid colors and testing points semi-transparent. The lower right shows the classification accuracy on the test set.

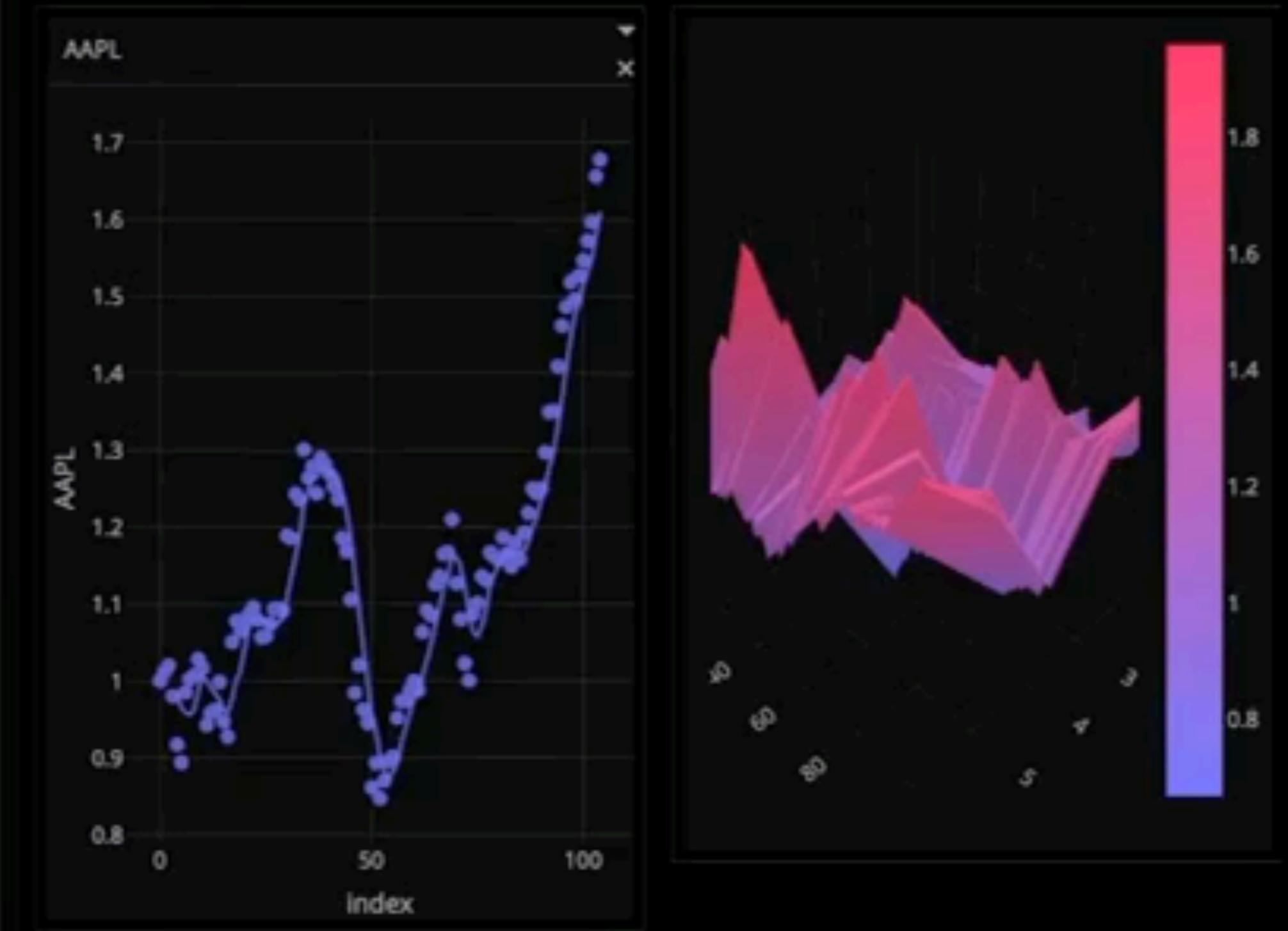


# Lecture 1-2: Programming environment

## I. Why Python?



```
1  from dash import Dash, dcc, callback, Input, Output
2  import plotly.express as px
3  import dash_design_kit as ddk # Dash Enterprise-only!
4  import plotly.graph_objs as go
5
6  df = px.data.stocks() # Fetch data from anywhere
7
8  app = Dash(__name__)
9
10 # Full control over the look and feel of your apps
11 app.layout = ddk.App([
12     ddk.Header(''),
13
14     ddk.Card([
15         dcc.Dropdown(df.columns, value='AAPL', id='dropdown'),
16         ddk.Graph(id='graph-1')
17     ], width=50),
18
19     ddk.Card(
20         ddk.Graph(
21             figure=go.Figure(data=go.Surface(z=df.values))),
22         width=50),
23
24     ddk.Card(
25         ddk.DataTable(data=df.round(2).to_dict('r')),
26         width=100),
27
28 ])
29
30 # Dash callback functions respond to user input,
31 # like selecting dropdowns or entering text
32 @callback(
33     Output('graph-1', 'figure'),
34     Input('dropdown', 'value'))
35 def update(stock_ticker):
36     return px.scatter(
37         df, y=stock_ticker,
38         trendline='rolling',
39         trendline_options=dict(window=5),
40     )
41 app.run(debug=True)
```



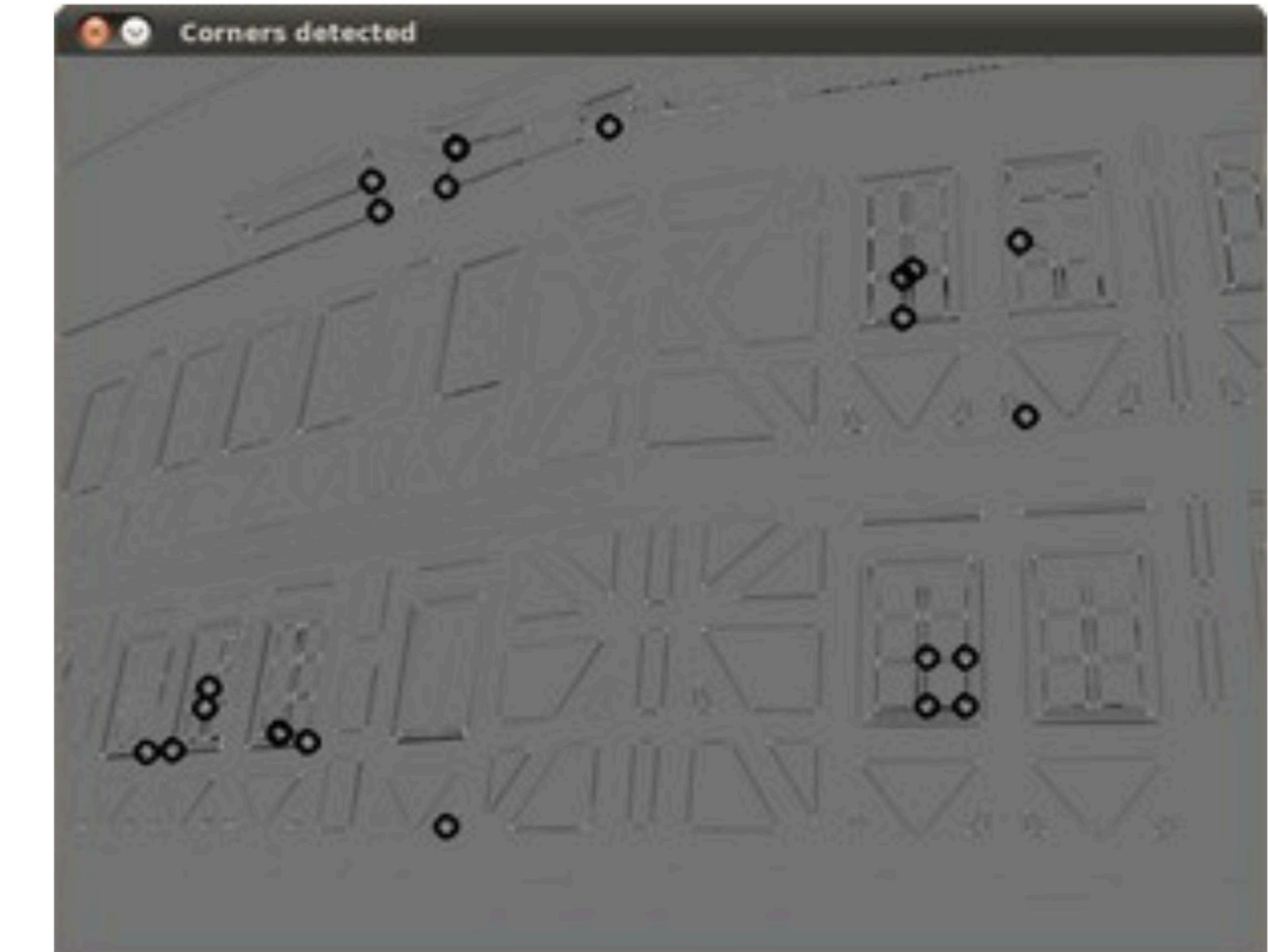
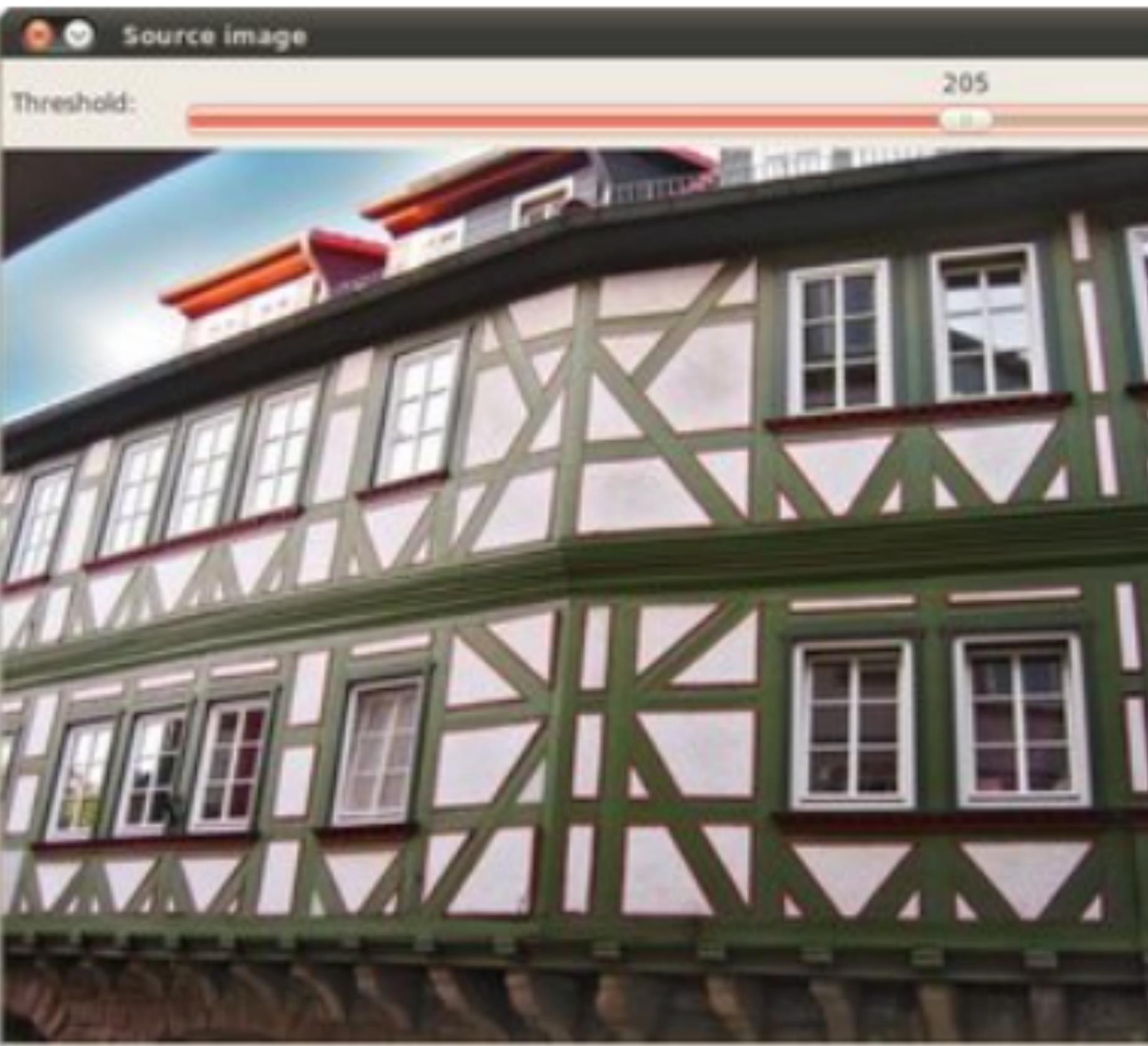
	AAPL	AMZN	FB	GOOG	MSFT	NFLX	date
1	1	1	1	1	1	1	2018-01-01
1.01	1.01	1.06	0.96	1.02	1.02	1.05	2018-01-08
1.02	1.02	1.05	0.97	1.03	1.02	1.05	2018-01-15
0.98	0.98	1.14	1.02	1.07	1.07	1.31	2018-01-22
0.92	0.92	1.16	1.02	1.01	1.04	1.27	2018-01-29
0.89	0.89	1.09	0.94	0.94	1	1.19	2018-02-05
0.99	0.99	1.18	0.95	0.99	1.04	1.33	2018-02-12



## I. Why Python?

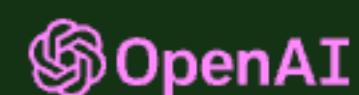
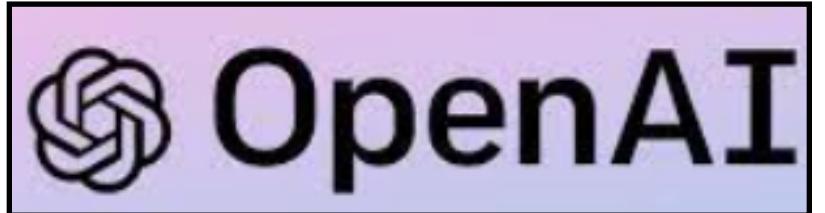


### Harris corner detector



# Lecture 1-2: Programming environment

## I. Why Python?



### ChatGPT: Optimizing Language Models for Dialogue

We've trained a model called ChatGPT which interacts in a conversational way. The dialogue format makes it possible for ChatGPT to answer followup questions, admit its mistakes, challenge incorrect premises, and reject inappropriate requests. ChatGPT is a sibling model to InstructGPT, which is trained to follow an instruction in a prompt and provide a detailed response.

[TRY CHATGPT ➔](#)

### Generations

The [image generations](#) endpoint allows you to create an original image given a text prompt. Generated images can have a size of 256x256, 512x512, or 1024x1024 pixels. Smaller sizes are faster to generate. You can request 1-10 images at a time using the [n](#) parameter.

Generate an image      python ▾    ⌂ Copy

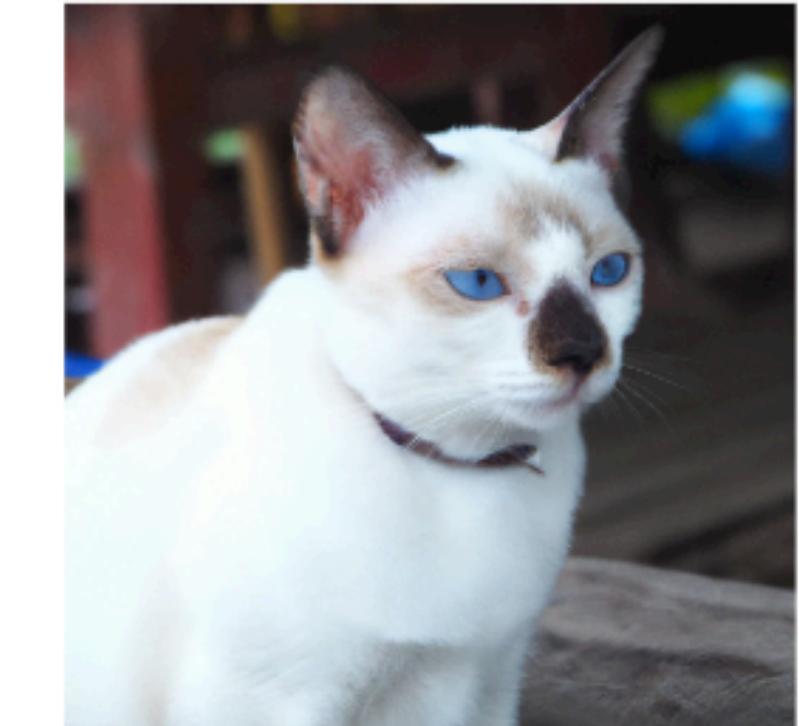
```
1 response = openai.Image.create(  
2     prompt="a white siamese cat",  
3     n=1,  
4     size="1024x1024"  
5 )  
6 image_url = response['data'][0]['url']
```

The more detailed the description, the more likely you are to get the result that you or your end user want. You can explore the examples in the [DALL-E preview app](#) for more prompting inspiration. Here's a quick example:

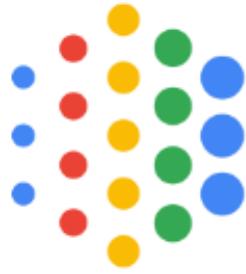
#### PROMPT

a white siamese cat

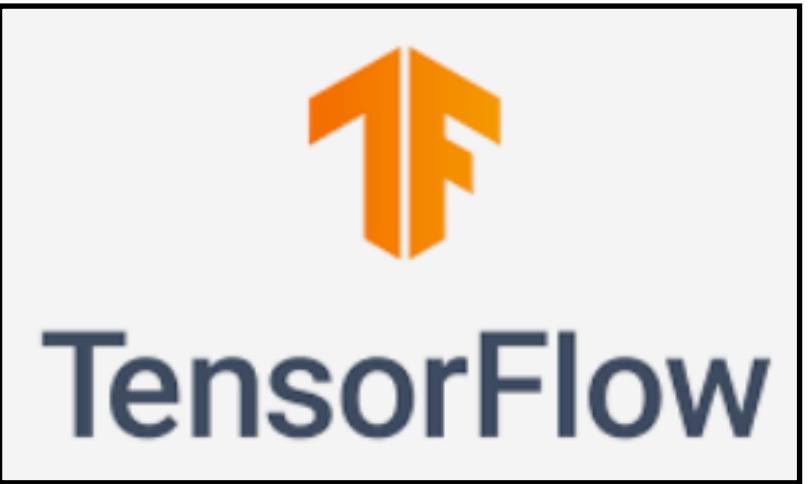
#### GENERATION



## I. Why Python?



Google AI



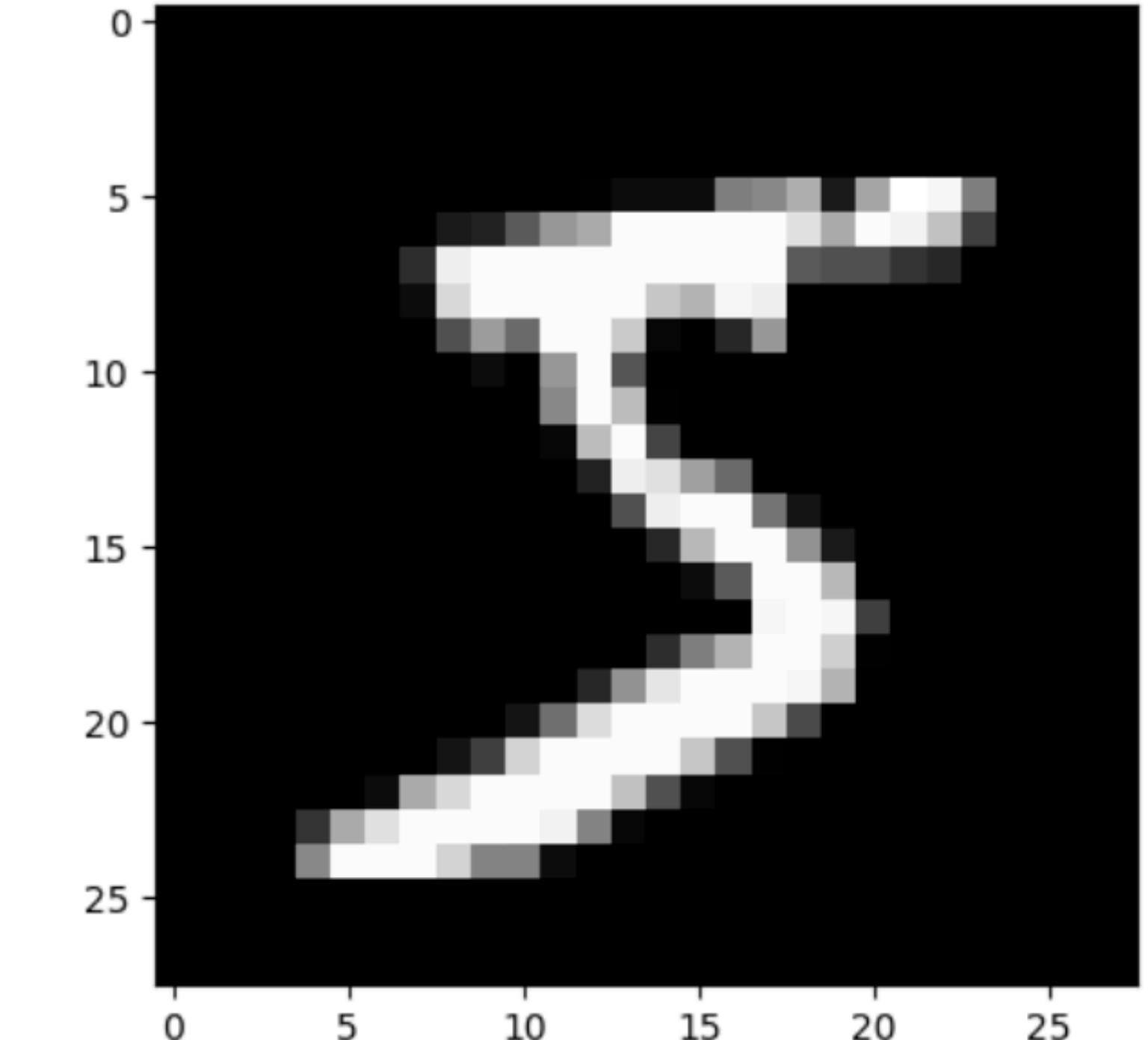
Meta AI



```
class Mnist_CNN(nn.Module):
    def __init__(self):
        super().__init__()
        self.conv1 = nn.Conv2d(1, 16, kernel_size=3, stride=2, padding=1)
        self.conv2 = nn.Conv2d(16, 16, kernel_size=3, stride=2, padding=1)
        self.conv3 = nn.Conv2d(16, 10, kernel_size=3, stride=2, padding=1)

    def forward(self, xb):
        xb = xb.view(-1, 1, 28, 28)
        xb = F.relu(self.conv1(xb))
        xb = F.relu(self.conv2(xb))
        xb = F.relu(self.conv3(xb))
        xb = F.avg_pool2d(xb, 4)
        return xb.view(-1, xb.size(1))

lr = 0.1
```



## II. Jupyter Notebook

It is a web-based interactive computing platform

- Good for prototyping and testing codes
- Easy to document and share your work

You can share the notebook through

a PDF file or an interactive webpage

with others

### Data

#### Multivariate normal distribution

First, we can simulate a structured data that is sampled from two different multivariate normal distributions.

```
In [194]: # number of clusters
k = 2
# number of data points in each cluster
n = 50
# total number of data points
N = n * k
# cluster properties: means and standard deviation
means = [[40, 40], [60, 70]]
stds = [[[80, 0],
          [0, 70]],
         [[60, 10],
          [10, 90]]]

# use np.random.multivariate_normal() to sample the datasets
data = []
for mean, std in zip(means, stds):
    data += [np.random.multivariate_normal(mean=mean, cov=std, size=n)]
data = np.concatenate(data)

# validation
print("Shape: ", data.shape)
print(data[:20])
```

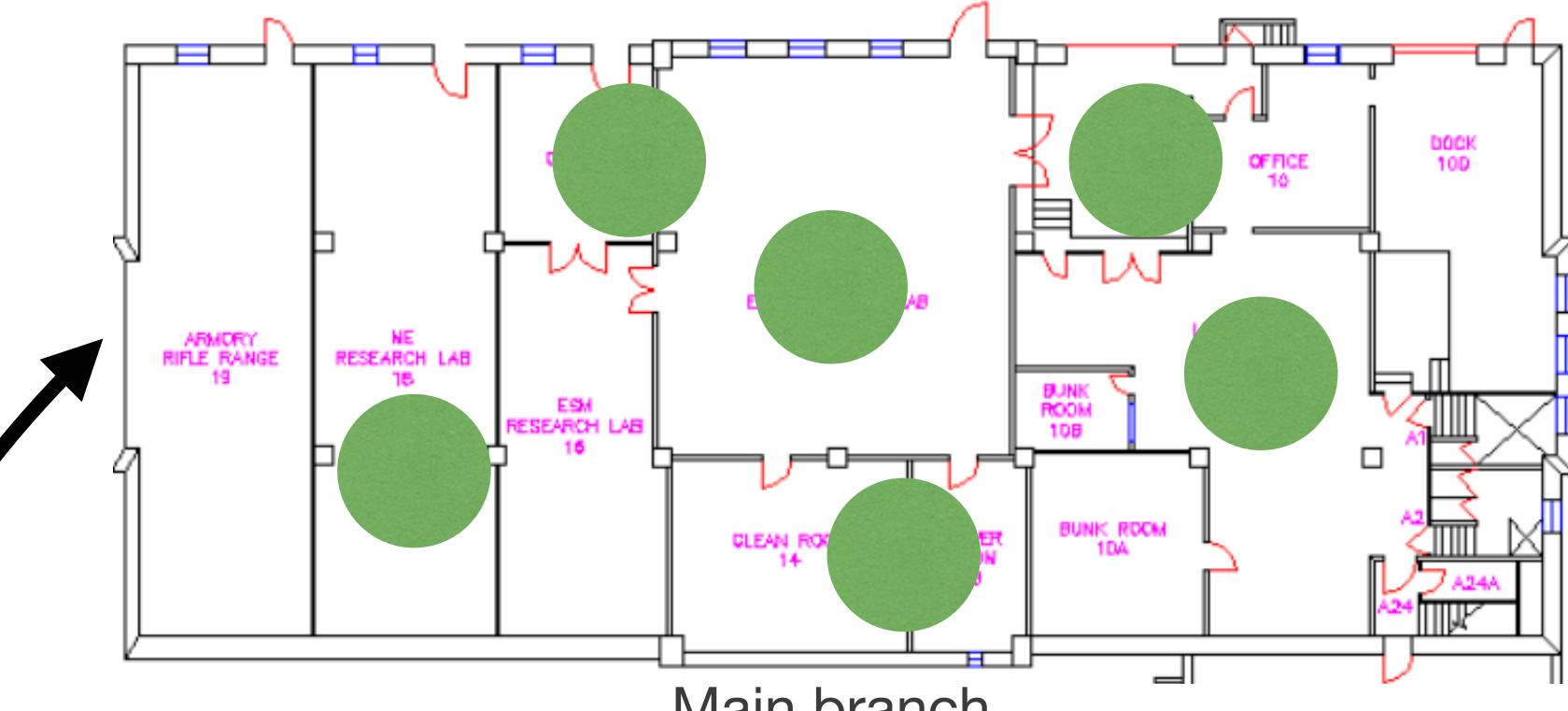
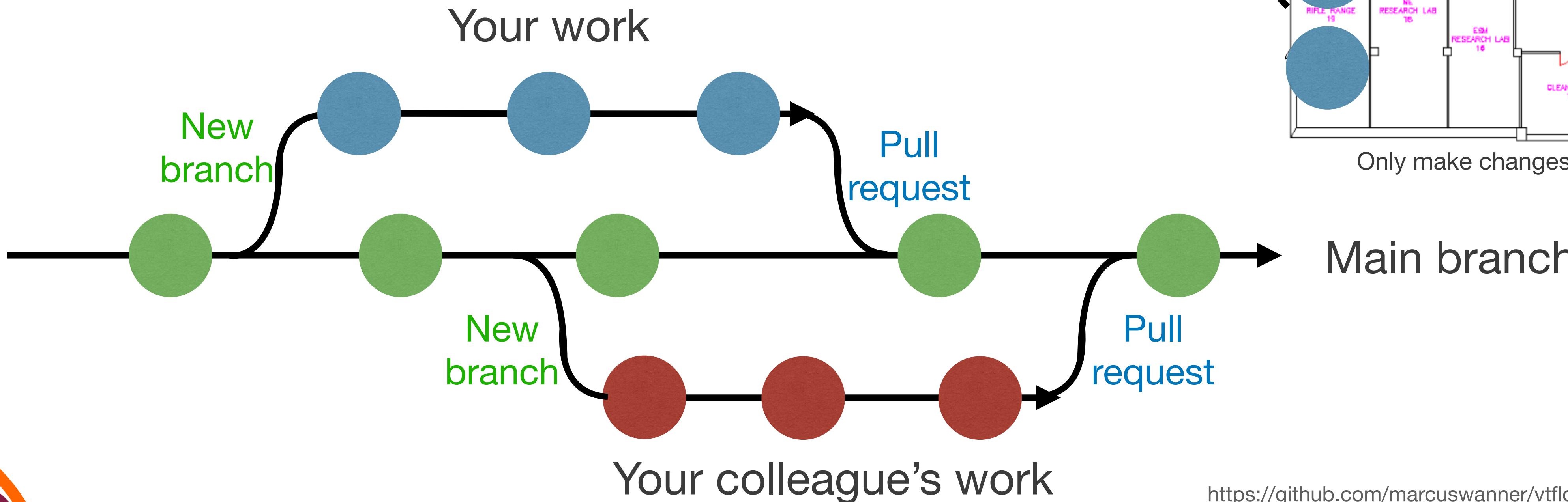
```
Shape: (100, 2)
[[51.66251802 47.84412914]
 [39.71618565 50.12966424]
 [35.30204586 30.11518838]
 [41.21290983 37.45870684]
 [33.50222942 27.44831926]
 [45.59088463 45.32844912]
 [42.53108047 46.10790442]
 [28.29073007 52.30456678]
 [38.31237176 34.84785694]
 [29.25183562 43.67950456]
 [40.58293457 25.99292839]
 [45.3154831 45.79598333]
 [22.58292304 39.54979532]
 [33.92459233 43.18324273]
 [38.06019806 40.15521412]
 [38.81693873 47.34977002]
 [43.93189465 42.68070032]
 [46.3391774 51.42474463]
 [42.13973106 38.61929858]
 [45.20335028 38.9488359 ]]
```

# Lecture 1-2: Programming environment

## III. Git and GitHub

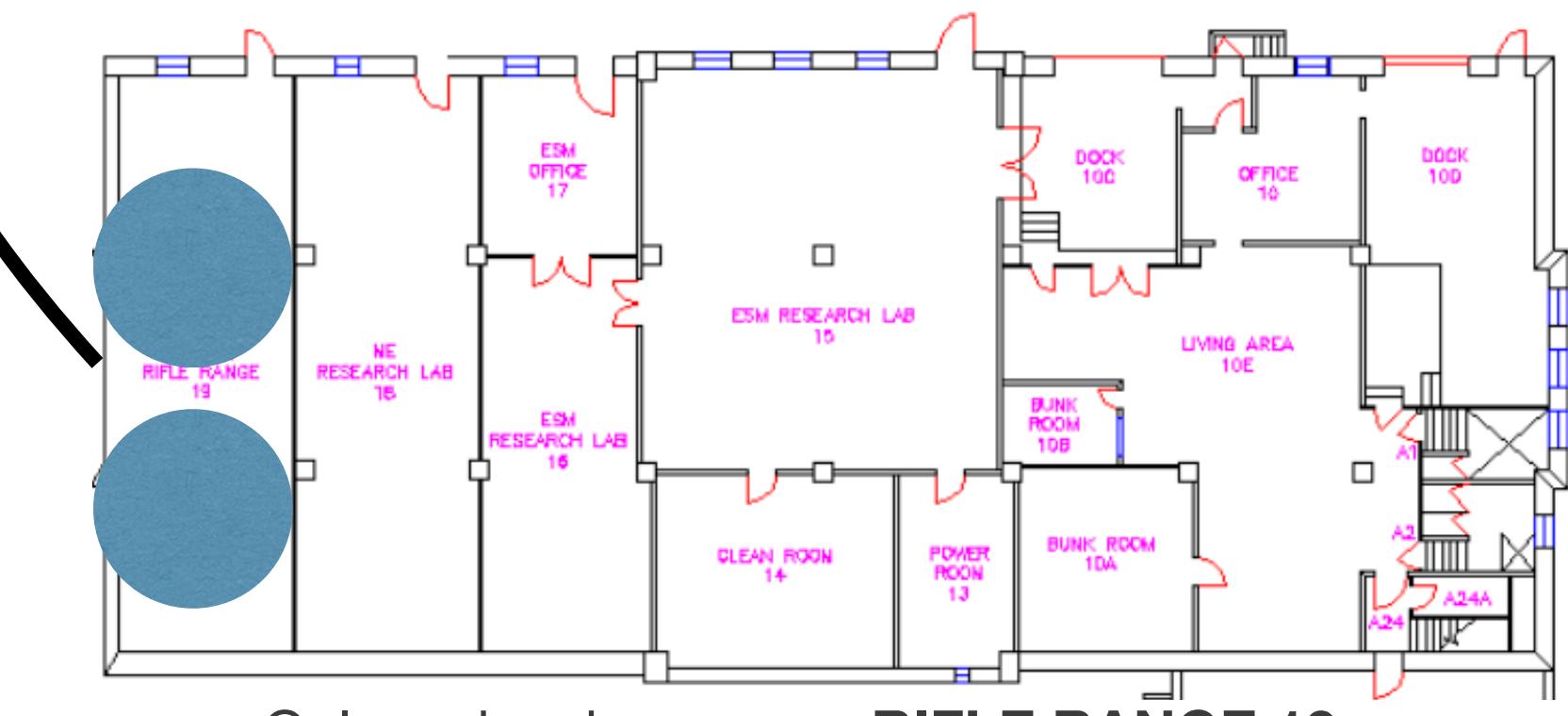
A time machine for your coding project!

- **Git** is a system that allows you to track and manage changes to your code over time
- It also allows you to merge changes from multiple sources (collaborators) into a single codebase.



Pull  
request

New  
branch

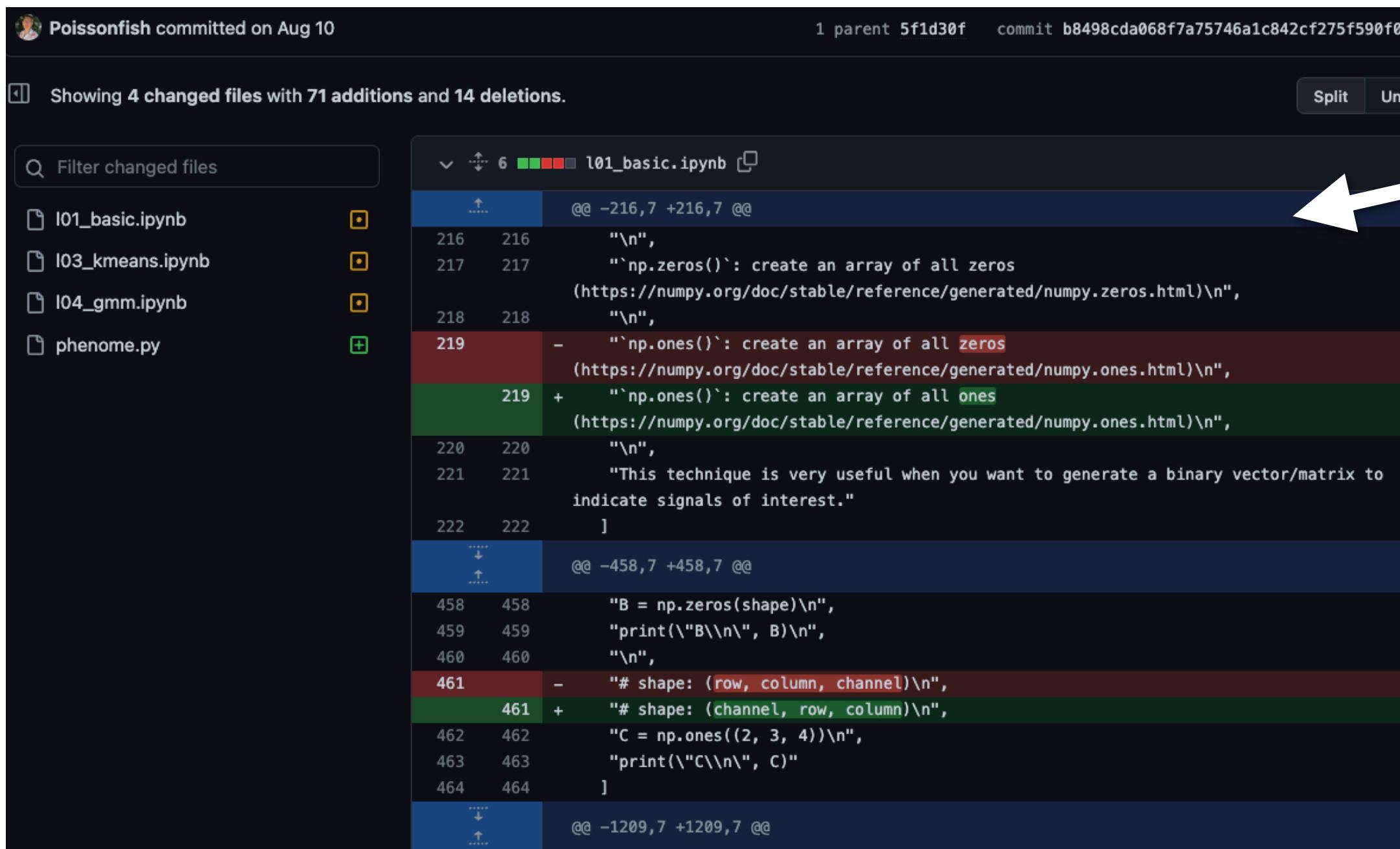


Main branch

## III. Git and GitHub

A time machine for your coding project!

- **Git** is a system that allows you to track and manage changes to your code over time
- It also allows you to merge changes from multiple sources (collaborators) into a single codebase.



Poissonfish committed on Aug 10

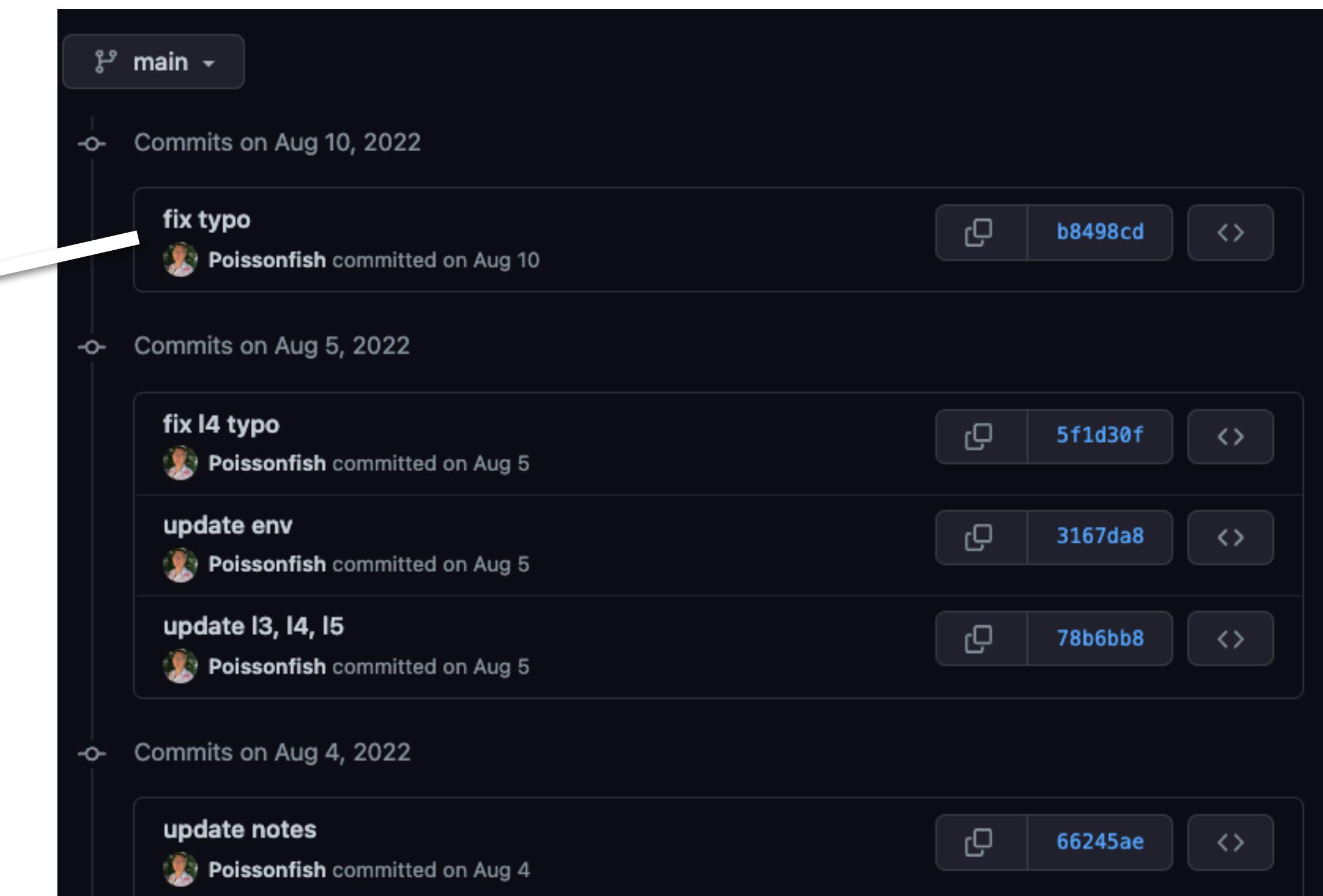
Showing 4 changed files with 71 additions and 14 deletions.

l01\_basic.ipynb

```
diff --git a/l01_basic.ipynb b/l01_basic.ipynb
--- a/l01_basic.ipynb
+++ b/l01_basic.ipynb
@@ -216,7 +216,7 @@ 
 216 216      "\n",
 217 217      ``np.zeros()``: create an array of all zeros
 218 218      (https://numpy.org/doc/stable/reference/generated/numpy.zeros.html)\n",
 219 219 -    ``np.ones()``: create an array of all zeros
 219 219 +    ``np.ones()``: create an array of all ones
 220 220      (https://numpy.org/doc/stable/reference/generated/numpy.ones.html)\n",
 221 221      "\n",
 221 221      "This technique is very useful when you want to generate a binary vector/matrix to
 222 222      indicate signals of interest."
 222 222 ]
```

1 parent 5f1d30f commit b8498cda068f7a75746a1c842cf275f590f0adaa

**GitHub** is a web-based, public platform that implements Git system.



Track every change you made

Historical records of your repository

# Lecture 1-2: Programming environment

## IV. Unix Shell

- **Unix shell** is a command-line interface that allows you to interact with the operating system.
- The major way to interact with a remote high-performance computer (e.g., VT ARC)
- You can use shell commands to:

Execute scripts

Copy files

Move / rename files

Delete files

Create folders

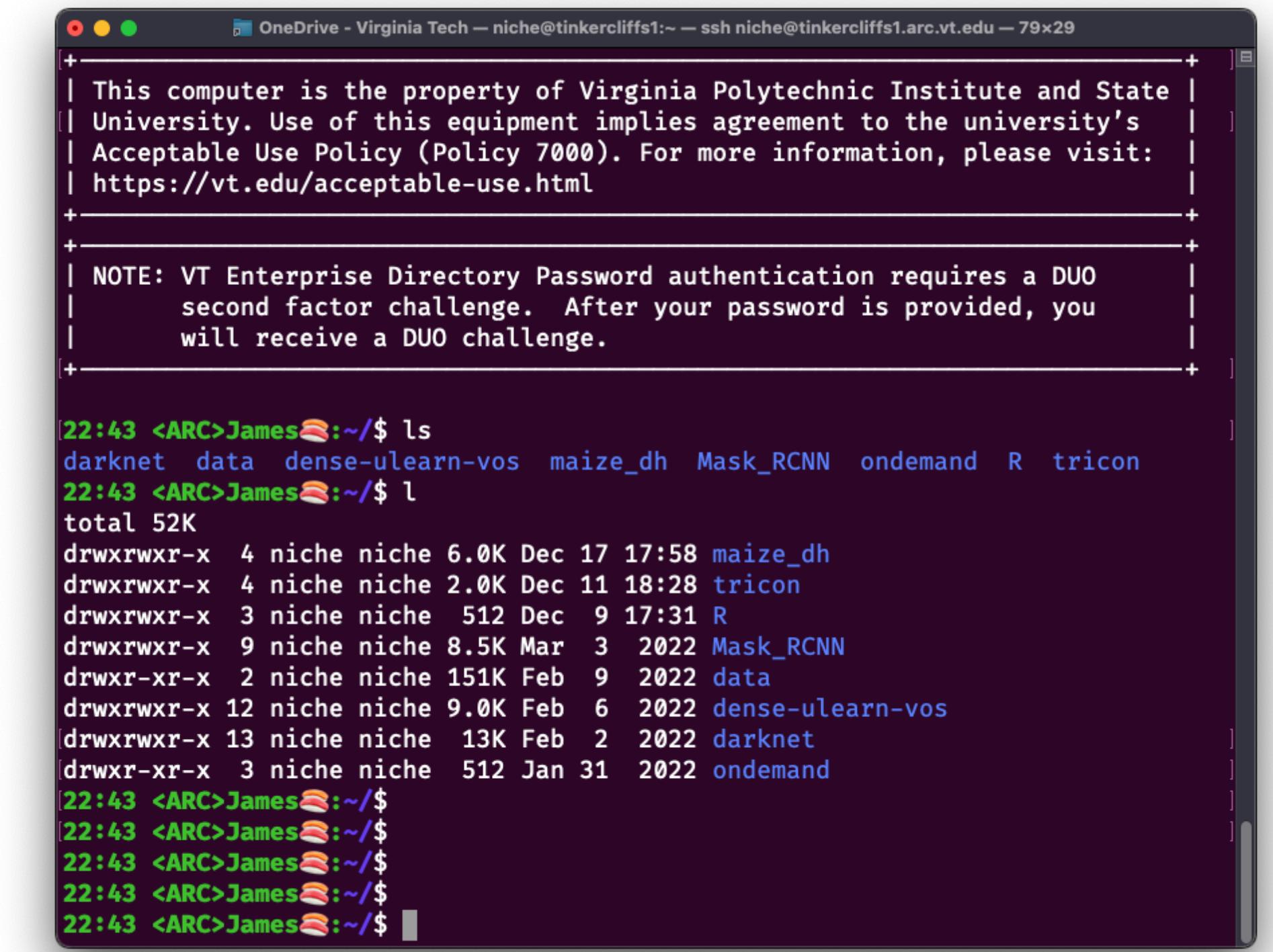
List files

Write texts to files

Display file contents

Change the working directory

Show the current working directory



This screenshot shows a terminal window titled "OneDrive - Virginia Tech - niche@tinkercliffs1:~ ssh niche@tinkercliffs1.arc.vt.edu - 79x29". It displays a message about university equipment usage and a note about VT Enterprise Directory Password authentication requiring a DUO second factor challenge. Below this, the user runs the 'ls' command, listing several files and directories including 'darknet', 'data', 'dense-ulearn-vos', 'maize\_dh', 'Mask\_RCNN', 'ondemand', 'R', and 'tricon'. The terminal then prompts for a password, indicating a DUO challenge has been received.

```
| This computer is the property of Virginia Polytechnic Institute and State |
| University. Use of this equipment implies agreement to the university's |
| Acceptable Use Policy (Policy 7000). For more information, please visit: |
| https://vt.edu/acceptable-use.html |
|
| NOTE: VT Enterprise Directory Password authentication requires a DUO |
| second factor challenge. After your password is provided, you |
| will receive a DUO challenge. |
|
[22:43 <ARC>James:~/]$ ls
darknet data dense-ulearn-vos maize_dh Mask_RCNN ondemand R tricon
[22:43 <ARC>James:~/]$ l
total 52K
drwxrwxr-x 4 niche niche 6.0K Dec 17 17:58 maize_dh
drwxrwxr-x 4 niche niche 2.0K Dec 11 18:28 tricon
drwxrwxr-x 3 niche niche 512 Dec 9 17:31 R
drwxrwxr-x 9 niche niche 8.5K Mar 3 2022 Mask_RCNN
drwxr-xr-x 2 niche niche 151K Feb 9 2022 data
drwxrwxr-x 12 niche niche 9.0K Feb 6 2022 dense-ulearn-vos
drwxrwxr-x 13 niche niche 13K Feb 2 2022 darknet
drwxr-xr-x 3 niche niche 512 Jan 31 2022 ondemand
[22:43 <ARC>James:~/]$ 
[22:43 <ARC>James:~/]$ 
[22:43 <ARC>James:~/]$ 
[22:43 <ARC>James:~/]$ 
[22:43 <ARC>James:~/]$ 
```

A screenshot of the VT ARC remote server

## V. Integrated Development Environment (IDE)

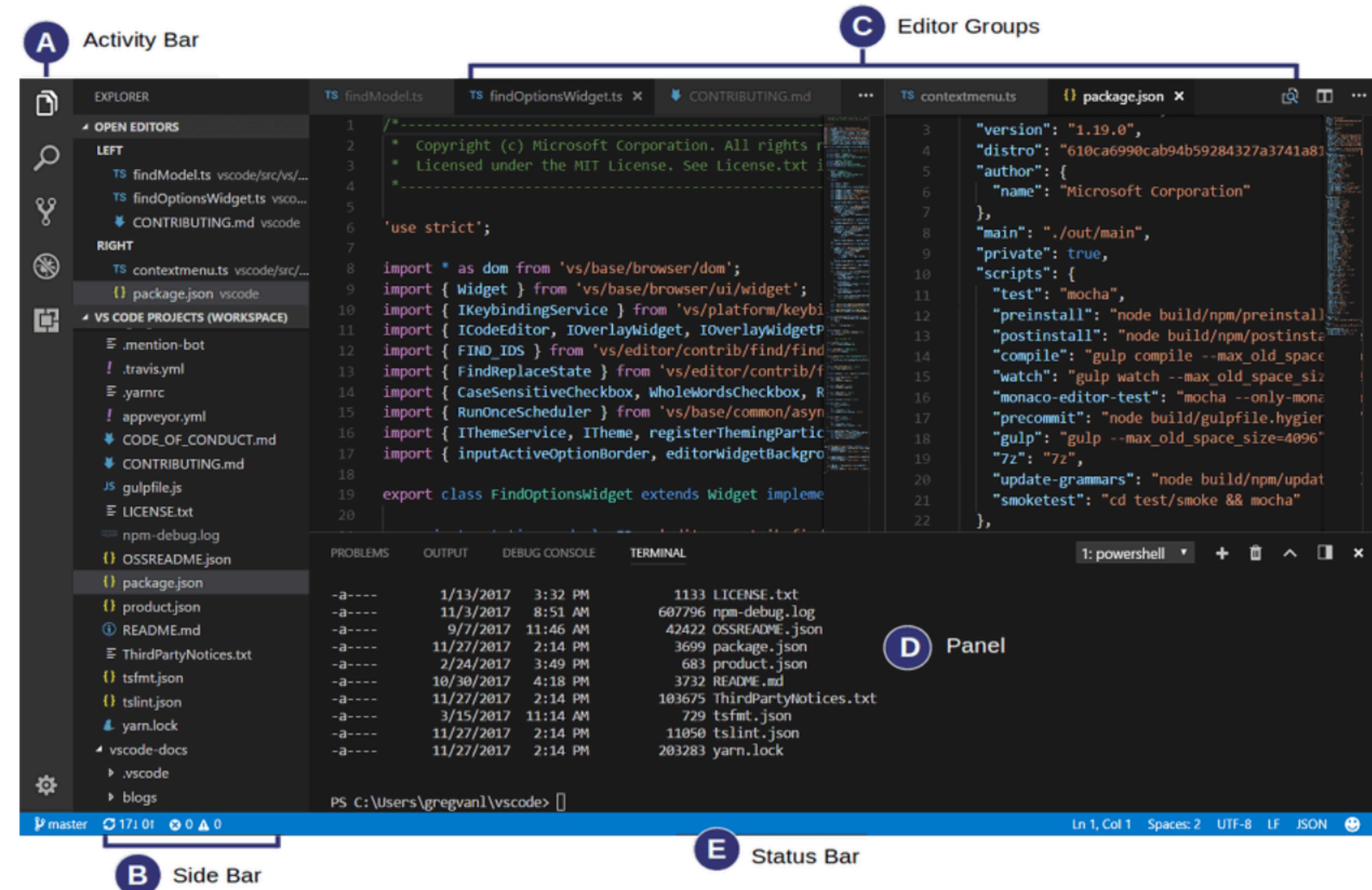
Integrated development environment (**IDE**) is a software application that provides a comprehensive set of tools for software development

Code editor

Automation tools

Warnings and suggestions

Code builder  
(e.g., LaTeX)



## V. Integrated Development Environment (IDE)

IDE put everything together!

The screenshot shows a Jupyter Notebook interface integrated into a larger development environment. The top navigation bar indicates the notebook is titled "Pi-Computer-Vision".

**Left Sidebar (File Explorer):**

- OPEN EDITORS: Shows multiple notebooks, including "221120\_segformer\_train.ipynb" which is currently active.
- PI-COMPUTER-VISION directory structure:
  - \_\_pycache\_\_
  - .ipynb\_checkpoints
  - .vscode
  - arc
  - data
  - James
  - models
  - node\_modules
  - notebooks (containing \_\_pycache\_\_ and .ipynb\_checkpoints)
  - out
  - packages
  - test (containing \_\_pycache\_\_, .ipynb\_checkpoints, .pytest\_cache, data, test\_file\_io.py, test\_models.py)
  - .env
- SOURCE CONTROL REPOSITORIES: Shows a commit message: "Message (%Enter to commit on "master")" and a list of changes in the "notebooks" folder.
- SOURCE CONTROL: Shows a "Commit" button.

**Central Area:**

**Code Editor:**

```
221120_segformer_train.ipynb
notebooks > 221120_segformer_train.ipynb > M+ Semantic Segmentation using nvidia/mit-b0 > M+ Insepct the 'scene_parse_150' dataset > M+ Image and Label > transformer (Python 3.9.13)

sturts like sky, road, grass, and discrete objects like person, car, bed.
Note that there are non-uniform distribution of objects occuring in the images,
mimicking a more natural object occurrence in daily scene.

Image and Label
```

**Output:**

```
plt.imshow(ds[5]["image"])
print(np.array(ds[5]["image"])[::2, ::2, :]) # first 2x2 pixels
print("Shape: ", np.array(ds[5]["image"]).shape)

[[[222 164 127]
 [222 164 127]]

 [[220 162 125]
 [220 162 125]]]
Shape: (512, 768, 3)
```

**Code Editor:**

```
221120_segformer_train.ipynb
notebooks > 221120_segformer_train.ipynb > M+ Semantic Segmentation using nvidia/mit-b0 > M+ Insepct the 'scene_parse_150' dataset > M+ Image and Label > transformer (Python 3.9.13)

plt.imshow(ds[5]["annotation"])
print(np.array(ds[5]["annotation"]))
print("Shape: ", np.array(ds[5]["annotation"]).shape)

[[[6 6 6 ... 6 0 0]
 [6 6 6 ... 6 0 0]
 [6 6 6 ... 6 0 0]
 ...
 [4 4 4 ... 4 4 0]
 [4 4 4 ... 4 4 0]
 [0 0 0 ... 0 0 0]]
Shape: (512, 768, 1)
```

**Terminal:**

```
bash - _03_Papers + ×
av
drwx----- 4 niche staff 128B Nov 13 20:54 talks
drwx----- 6 niche staff 192B Nov 11 11:31 _02_Grants_Docs
drwx----- 3 niche staff 96B Oct 6 00:40 _12_Reviews
drwx----- 5 niche staff 160B Oct 6 00:39 car
drwxr-xr-x+ 5 niche staff 160B Jan 28 2022 ..
o(transformer) 22:59 James:OneDrive - Virginia Tech/$
o(transformer) 22:59 James:OneDrive - Virginia Tech/$ ls
DC_MidJourney4_please_generate_a_landing_web_page_layout_for_a_
_57661d8e-c5fd-46e1-966a-25d235246c3c.png
DC_MidJourney4_please_generate_a_landing_web_page_layout_for_a_
_5992112d-e8d2-4791-a873-58fe6d1fb6ee.png
Icon?
Microsoft Teams Chat Files
_01_Lab
_02_Grants
_02_Grants_Docs
_03_Papers
_03_Papers_Docs
_04_Software
_05_Talks_Travels
_05_Teaching
_06_Mautushi
_06_Shihong
_06_Xiaohui
_07_Writing
_11_Admin
_12_Reviews
_99_Arc
_99_Random
car
eructation_01.wav
notebooks
talks
o(transformer) 22:59 James:OneDrive - Virginia Tech/$ cd _03_P
apers
o(transformer) 22:59 James:OneDrive - Virginia Tech/$ ls
DH_maize Tricon eigen_aug preg_detection
o(transformer) 22:59 James:OneDrive - Virginia Tech/$ ls -alht
ls: t: No such file or directory
o(transformer) 22:59 James:OneDrive - Virginia Tech/$ ls -alht
total 24
drwx-----@ 31 niche staff 992B Dec 25 23:51 ..
drwx----- 29 niche staff 928B Dec 21 18:47 DH_maize
-rw-r--r--@ 1 niche staff 8.0K Dec 16 17:08 .DS_Store
drwx----- 5 niche staff 160B Dec 16 14:00 eigen_aug
drwxr-xr-x 7 niche staff 224B Dec 16 13:58 .
drwxr-xr-x@ 23 niche staff 736B Dec 15 18:08 Tricon
drwx----- 14 niche staff 448B Dec 5 16:24 preg_detection
o(transformer) 22:59 James:OneDrive - Virginia Tech/$
```

## V. Integrated Development Environment (IDE)

IDE put everything together!

