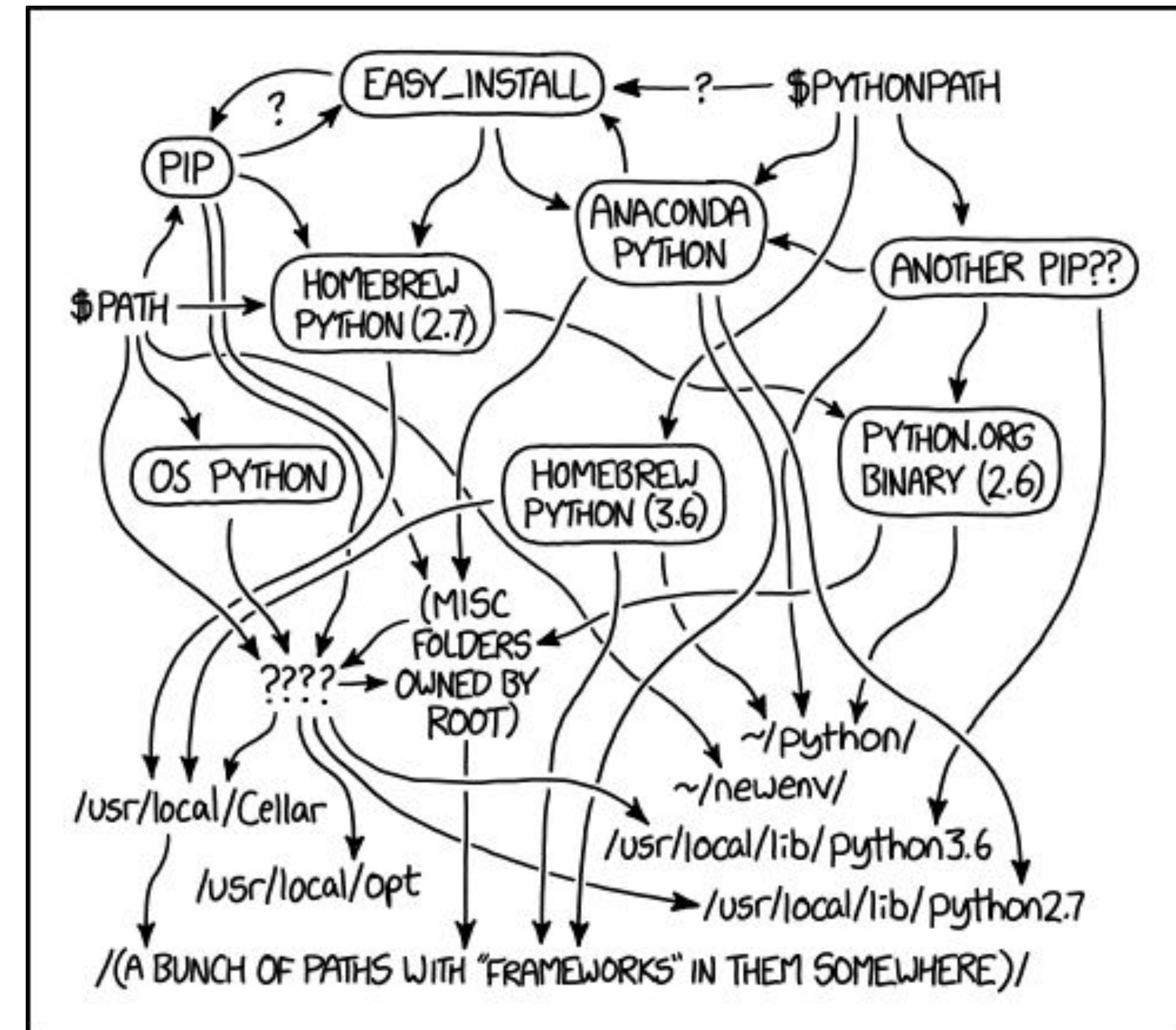


Python Part 4

Spring 2022
PCfB Class 7
February 25, 2022



light roast comics



MY PYTHON ENVIRONMENT HAS BECOME SO DEGRADED
THAT MY LAPTOP HAS BEEN DECLARED A SUPERFUND SITE.

Outline

- Stand-alone python scripts
- **argparse** module
- Functions
- Sets

Stand-alone python
scripts

To make a text file a Python script

argparse module

argparse generates help documentation

```
(base) jtl-macbook-pro:Demos jtladner$ ./calc.py -h
usage: calc.py [-h] [-o {+,-,*,/,**, %}] a b

positional arguments:
  a          First number to use in calculation
  b          Second number to use in calculation

optional arguments:
  -h, --help          show this help message and exit
  -o {+,-,*,/,**, %}, --operation {+,-,*,/,**, %}
                        Operation to perform
```



```
import argparse
```

```
parser = argparse.ArgumentParser()
```

```
args = parser.parse_args()
```

```
(base) jtl-macbook-pro:Demos jtladner$ ./test.py -h
usage: test.py [-h]

optional arguments:
  -h, --help  show this help message and exit
```

Positional arguments

Optional arguments

```
parser = argparse.ArgumentParser()
```

```
parser.add_argument("num",  
help="Number of hits to report")
```

```
args = parser.parse_args()
```

```
usage: test.py [-h]
```

```
optional arguments:
```

```
  -h, --help  show this help message and exit
```

```
usage: test.py [-h] num
```

```
positional arguments:
```

```
  num          Number of hits to report
```

```
optional arguments:
```

```
  -h, --help  show this help message and exit
```



```
parser = argparse.ArgumentParser()
```

```
parser.add_argument("-n", "--num",  
help="Number of hits to report")
```

```
args = parser.parse_args()
```

```
usage: test.py [-h]
```

```
optional arguments:
```

```
  -h, --help  show this help message and exit
```

```
usage: test.py [-h] [-n NUM]
```

```
optional arguments:
```

```
  -h, --help          show this help message and exit
```

```
  -n NUM, --num NUM  Number of hits to report
```

Functions

```
def sqrt(num):  
    squareroot = float(num)**(0.5)  
    return squareroot
```



```
def sqrt(num):  
    squareroot = float(num)**(0.5)  
    return squareroot
```

```
numstring="123456789"  
sqroots = [sqrt(x) for x in numstring]
```

```
def sqrt(num):  
    squareroot = float(num)**(0.5)  
    return squareroot
```

```
numstring="123456789"  
sqroots = [sqrt(x) for x in numstring]
```

```
sqroots = [float(x)**(0.5) for x in numstring]
```



```

209 ▼ def mergeClusts(clusts, k, indivThresh, clustThresh):
210     pairs2merge = []
211     for i in range(len(clusts)):
212         for j in range(i+1, len(clusts)):
213             hits=0
214             comps=0
215             for n1,s1 in clusts[i].items():
216                 for n2,s2 in clusts[j].items():
217                     comps+=1
218                     if kmerOvlp(s1, s2, k)>=indivThresh:
219                         hits+=1
220                     if hits/comps >= clustThresh:
221                         pairs2merge.append([i,j])
222             #
223     groups2merge = combPairs(pairs2merge)
224     #
225     newClusts = []
226     singles = []
227     merged = []
228     for each in groups2merge:
229         newClusts.append({})
230         for a in each:
231             merged.append(a)
232             for k,v in clusts[a].items():
233                 newClusts[-1][k] = v
234     for i,info in enumerate(clusts):
235         if i not in merged:
236             if len(info) == 1:
237                 singles.append(info)
238             else:
239                 newClusts.append(info)
240     return newClusts, singles

```

Sets

Sets

Compare collections of items

`A.intersection(B)`

`A.union(B)`

`A.difference(B)`

`A.symmetric_difference(B)`

`A.issubset(B)`

`A.issuperset(B)`

