Protocols, Routing, and More Lists

- Internet Protocol
  - Useful command line tools like `ping` and `traceroute/tracert`
- More lists in python
- Internet Routes

Your future in CS

I used to include this on my slides, but since these slides have changed - going to just leave it up here for every notebook. I get a lot of questions about more programming courses, the concentrations, and minors in computer science. Here is a brief reminder.

CS 164 – Next Course In Sequence, also consider CS 220 (math and stats especially)

- CO Jobs Report 2021 – 77% of *all* new jobs in Colorado require programming
- 60% of all STEM jobs requires *advanced* (200-300 level)
- 31% of all Bachelor of Arts degree titled jobs also required coding skills
- 2016 Report found on average jobs that require coding skills paid $22,000 more

- Concentrations in CS:

  - Computer science has a number of concentrations.
    - General concentration is the most flexible, and even allows students to double major or minor pretty easily.
    - Software Engineering
    - Computing Systems
    - Human Centered Computing
    - Networks and Security
    - Artificial Intelligence
    - Computer Science Education.
  - Minors:
    - Minor in Computer Science - choose your own adventure minor
    - Minor in Machine Learning - popular with stats/math, and engineering
    - Minor in Bioinformatics - Biology + Computer Science

A System of Protocols

The internet is built using 'layers' each layer adding more functionality / features.

Definition review:

- Bits

    - A single 0 or 1 in a file, which is made up of many bytes.
- Bit Rate
    - the speed in which bits are transfers over the net
    - 10mb/s - 10 megabytes per second or ~80,000,000 bits
- Protocol
    - An agreed upon standard that folks follow for communication
- IP - "Internet Protocol"
    - An agreed system in which computers communicate with each other (part of the 'network layer')
- IP Address
    - Addresses assigned to devices on the network
    - IPv4 - 129.82.45.48
    - IPv6 - 0000:0000:0000:0000:0000:ffff:8152:2d30
- URL - Uniform Resource Locator
    - Text based address that maps to IP addresses via router tables

Code.org Video on IP Protocol

Video Link on YouTube

Testing it out

Let's test it out on your computer.

Mac

- Type ⌘ + space bar -> brings up spotlight search
- type "terminal" to view and open the Terminal.app

Windows

- Type ⊞ (windows key) + S to open search
- Type "Terminal" to open the Windows Terminal app
    - You can also type Powershell - may vary based on your OS version

Ping

- The ping command
- Name comes from sonar 'ping'
- Determines amount of time it take to send a small amount of data (a packet) to a given location

Type: > ping www.cs.colostate.edu

Type: cntrl + c to end the command running

More Lists

Take a moment, you will notice most information we are getting are lists of information. Let's review python lists giving us more tools moving forward!

- Both Lists and Strings are “Sequential” Types
  - Lists can contain any value
  - Lists Are mutable
- Methods that modify the list:
  - list.append(item) – adds an item (without modification) to end
  - list.insert(item, index) – inserts item at set location
  - list.remove(item) – removes the first occurrence of the item
  - list.pop(index) – optional index, removes the item at location or location 0
  - list.sort() – sorts the list, it changes the list!
    - sorted(list) - returns a sorted copy of the list
  - list.reverse() – reverses the list, it changes the list!
    - reversed(list) - returns a reversed copy of the list.
- Support Methods
  - list.index(item) - returns the first location of the item, or -1 if not found (not find!)
  - list.count(item) – returns the number of times the item shows up
  - list.copy() – returns a new copy of the list
- Methods for sequences
  - len(list) - gives the length
  - max(list) - gives the largest value
  - min(list) - gives the smallest value
  - sum(list) - adds all values together, **only works for numeric types**
- Sequence Operations
  - - - concatenate two lists into one
  - <, >, <=, >=, == - compare items individually
  - in - tells if an item exists in a list

In [ ]:

```
cast = ["Westley", "Buttercup", "Inigo Montoya", "Vizzini",
        "Fezzik", "Rugan", "Humperdinck"]
cast2 = cast.copy()
cast.sort() # Modifies cast, but not cast2 because cast2 is a copy!!

print("Cast:", cast)
print("Cast 2:", cast2)


if cast == sorted(cast2):
    print("They are equal") # this does print!
for actor in reversed(cast):
    print(actor) # prints from Westley to Buttercup

print(max(cast)) # Westley
print(min(cast)) # Buttercup
```

```
Cast: ['Buttercup', 'Fezzik', 'Humperdinck', 'Inigo Montoya', 'Rugan', 'Vizzini', 'We
stley']
Cast 2: ['Westley', 'Buttercup', 'Inigo Montoya', 'Vizzini', 'Fezzik', 'Rugan', 'Hump
erdinck']
They are equal
Westley
Vizzini
Rugan
Inigo Montoya
Humperdinck
Fezzik
Buttercup
Westley
Buttercup
```

Student Practice:

Take a list of numbers, and find the following values:

- Min
- Max
- Total
- Average
  - You will have to calculate this one, the others have functions that can help
- Print them using the following format.
  - Your function can return the String generated, and then print!

```
Max: val,  Min: val, Total: val, Avg: val
```

or a sample output

```
[5, -29, -72, -38, 77, -28, 13, 24, 7, 78]
Max: 78, Min: -72, Total: 37, Average: 3.7
```

In [ ]:

```python
import random
rndlst = random.sample(range(-99, 99), 10) # this code helps generate a random list of
print(rndlst) ## just so you can see what is generated

def practice_values(lst):
    mx = max(lst)
    mn = min(lst)
    sm = sum(lst)
    avg = sm / len(lst)
    return f"Max: {mx}, Min: {mn}, Total: {sm}, Average: {avg}"

print(practice_values(rndlst))
```

```
[-74, -66, -77, -18, 86, -3, 37, 90, 76, -36]
Max: 90, Min: -77, Total: 15, Average: 1.5
```

Enumerate

- We want to cycle through elements
  - And, we want to keep track of locations!
  - introducing - enumerate

Helper function for sequences and loops

```
In [ ]: for index, actor in enumerate(cast):
            print(f"{index}: {actor}")
```

```
0: Buttercup
1: Fezzik
2: Humperdinck
3: Inigo Montoya
4: Rugan
5: Vizzini
6: Westley
```

More Practice

If you recall splitting a string gives us a list of strings. For example:

```
In [ ]: ip = "129.82.45.48"
        components = ip.split(".")
        print(components)
```

```
['129', '82', '45', '48']
```

However, if we want to add 2 or something to a component, we can't do that as a string!

Instead we need to convert them to int values (by building a new list).

- For this practice, you will take IP address strings (see the ones provided below).
- Use split to create a list of strings
- loop through the list of strings, creating a **new** list of int values
  - remember int(val) converts a string to an int
- Add 2 to the 4th value (3rd index) of the list
- return the list of int values

```
In [ ]: def add_two_to_ip(address):
            iplst = address.split(".")
            tmp = []
            for index, component in enumerate(iplst):
                tmp.append(component if index < 3 else str(int(component) + 2))
            return tmp
```

```
In [ ]: ip = "129.82.45.48"
        ip2 = "129.82.45.13"
        ip3 = "129.82.45.54"

        print(add_two_to_ip(ip))
        print(add_two_to_ip(ip2))
        print(add_two_to_ip(ip3))
```

```
['129', '82', '45', '50']
['129', '82', '45', '15']
['129', '82', '45', '56']
```

Routing

- What is the primary purpose of the internet?
  - To share information
  - Information that is stored in files
- Given a mesh network, how does one file get to another computer?
  - Through a list of computers that make up the route(s)

Code.org video on Routing

- What are some common definitions from the video?
  - Fault Tolerant
    - If an issue arrives, how does the network automatically adapt
  - Redundancy
    - The ability to send packets down multiple paths
  - Packet
    - a small subset of a file to make transmission easier
  - Transmission/Transfer Control Protocol (TCP)
    - A system setup to make sure all the data has arrived
    - Continues to request data until all parts arrive
  - Domain Name Service (DNS)
    - Companies and software that match domain names (URLs) to IP addresses
  - Internet Service Provider (ISP)

Internet Service Provider

- Local individuals connect to the "larger" network through ISPs.
- Most ISPs have star networks based on your router to the ISP.
- They then connect to the broader world
- Often their router tables (DNS lookup tables) are critical components

> Discussion
> What happens when an ISP gets to determine which data goes through their machines?
> What happens if they get to determine the speed of that data through their machines?

Let's View Some Routes with traceroute / tracert

Mac

- Type ⌘ + space bar -> brings up spotlight search
- type "terminal" to view and open the Terminal.app
- >traceroute www.google.com
- >traceroute www.cs.colostate.edu/~cs150b

Windows

- Type ⊞ (windows key) + S to open search
- Type "Terminal" to open the Windows Terminal app
    - You can also type Powershell - may vary based on your OS version
- >tracert www.google.com
- >tracert www.cs.colostate.edu/~cs150b

How are these tables built and controlled? We will play with dictionaries in this class next week - but this is a topic for later classes!