William Peng

Professor Smita

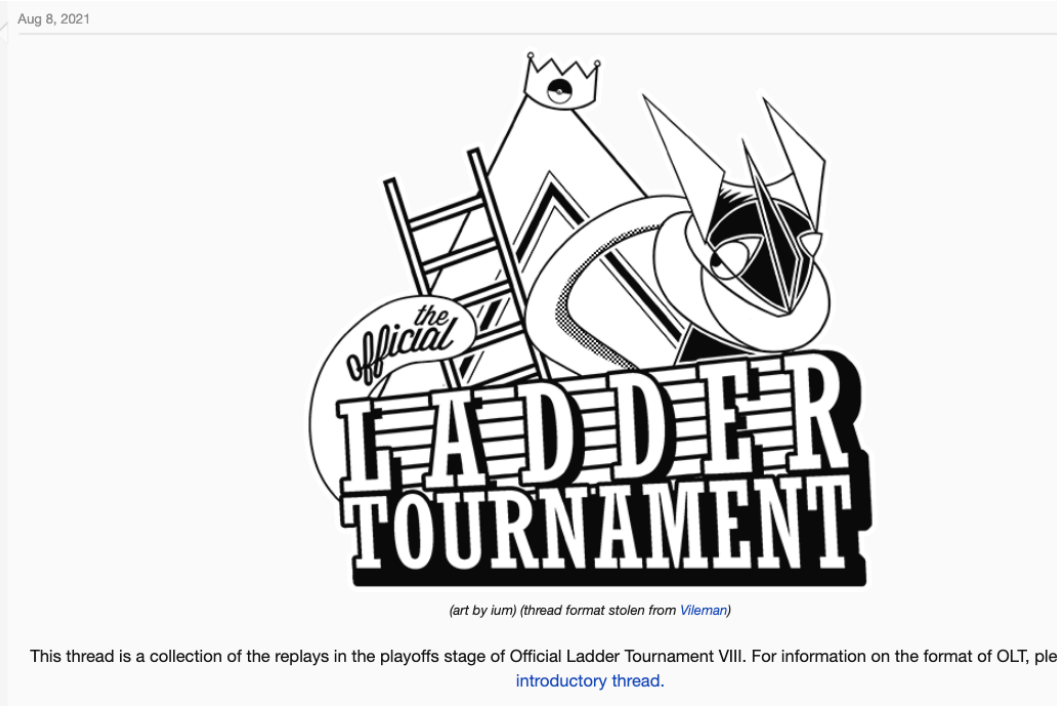
CSCI 168

14 March 2021

Project Report

Introduction.

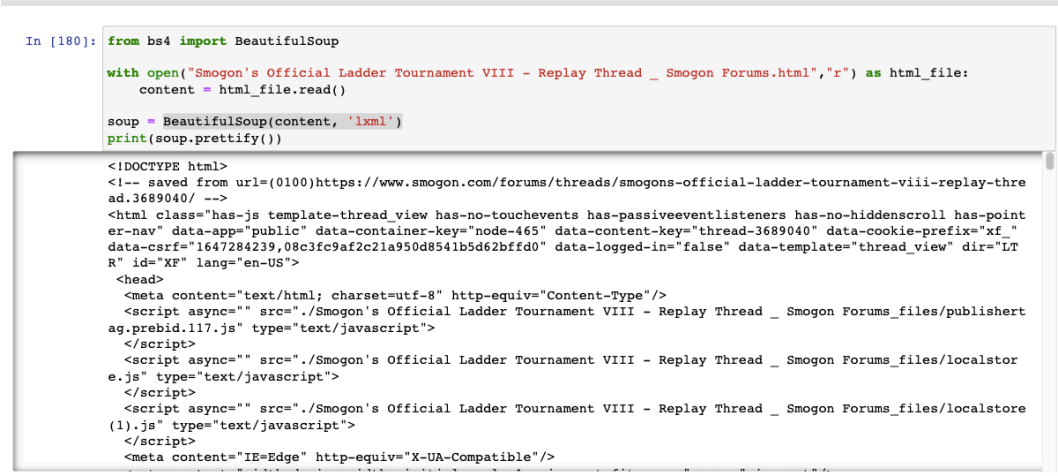
In this project, I will scrape data from **smogon.com**, a competitive Pokemon website, to find out individual player's performances in tournaments. The performances of a player will be measured by his/her wins and losses in tournament games. Specifically, this project will focus on the Generation 8(Sword&Shield) Overused tier(OU), which is the most popular and the standard tier on this forum. As a result, I will start by looking at the Smogon's Official Ladder Tournament VIII, which exclusively consists of SWSH OU games. To calculate the final results, I will employ two methods: the first by simple getting the win loss ratio, and the second being using PageRank and observing the relationship between the results I get by using two methods.



The Official Ladder Tournament VIII replay thread (https://www.smogon.com/forums/threads/smogons-official-ladder-tournament-viii-replay-thread.3689040/)

Implementation.

The first step to start this project is to find a way to scrape the data I need. After researching on the internet, I stumbled upon a youtube video that shows me how to use Beautiful Soup to complete this step. According to **crummy.com**, "Beautiful Soup is a Python library for pulling data out of HTML and XML files. It works with your favorite parser to provide idiomatic ways of navigating, searching, and modifying the parse tree." To use beautiful soup in python, I first imported Beautiful soup from bs4. Then, just like opening a text file, I opened the html file and used .read() to input the page contents into a variable 'content'. Beautiful soup is applied to 'content' with the parser method as 'lxml' and the result is saved in the variable 'soup'. Lastly, I can print out the soup with print(soup.prettify()) to view the page source in a comprehensible format.



```
In [180]: from bs4 import BeautifulSoup

with open("Smogon's Official Ladder Tournament VIII - Replay Thread _ Smogon Forums.html","r") as html_file:
    content = html_file.read()

soup = BeautifulSoup(content, 'lxml')
print(soup.prettify())
```

```
<!DOCTYPE html>
<!-- saved from url=(0100)https://www.smogon.com/forums/threads/smogons-official-ladder-tournament-viii-replay-thre
ad.3689040/ -->
<html class="has-js template-thread_view has-no-touchevents has-passiveeventlisteners has-no-hiddenscroll has-point
er-nav" data-app="public" data-container-key="node-465" data-content-key="thread-3689040" data-cookie-prefix="xf_"
data-csrf="1647284239,08c3fc9af2c21a950d8541b5d62bffd0" data-logged-in="false" data-template="thread_view" dir="LT
R" id="XF" lang="en-US">
 <head>
  <meta content="text/html; charset=utf-8" http-equiv="Content-Type"/>
  <script async="" src="./Smogon's Official Ladder Tournament VIII - Replay Thread _ Smogon Forums_files/publishert
ag.prebid.117.js" type="text/javascript">
  </script>
  <script async="" src="./Smogon's Official Ladder Tournament VIII - Replay Thread _ Smogon Forums_files/localstor
e.js" type="text/javascript">
  </script>
  <script async="" src="./Smogon's Official Ladder Tournament VIII - Replay Thread _ Smogon Forums_files/localstore
(1).js" type="text/javascript">
  </script>
  <meta content="IE=Edge" http-equiv="X-UA-Compatible"/>
```

Using BeautifulSoup to scrape the replay thread.

The next step is to find the usernames of all the tournament contenders in the playoffs. Luckily for me, there is a replay thread of this tournament that contains all the replays of the games in the playoffs. Upon inspecting the page, the tag to find username is <a class="username" ….>. Therefore, I can use soup.find_all('a', class_="username") to get all the lines containing usernames of the members on this page. Then I iterate through each name and use names.text.replace(',') to make the crop out the excessive information and only leave the names. However, I realized that

there is one username I shouldn't include, which is the name of the thread's poster and "Wigglytuff" and the possible co-manager of the thread "Vileman". During the for loop, I set up an if statement with the condition that if the name is not "Wigglytuff" and "Vileman", I will put them in a list that contains all usernames. After getting the list of usernames, I turned the list into a set and back into a list to get the list of names in which one names only appears once. The list is confirmed correct when the printed length is 32 because there are actually 32 players contending in the playoffs.

```
In [211]: usernames = soup.find_all('a', class_="username")
          unames = []
          for names in usernames:
              names = names.text.replace('','')
              if names != 'Wigglytuff'and names != "Vileman":
                  unames.append(names)

In [212]: print(unames)

['Sagiri', 'Ahsan-219', 'Punny', 'Lusa', 'haxrme', 'devin', 'MichaelderBeste2', 'ABR', 'Tace', 'Tricking', 'Insult',
'Chaitanya', 'TDNT', 'Ayoukoo', 'Empo', 'BIHI', 'Rubyblood', 'Ox the Fox', 'eclipseY', 'Misterioussaint', 'HSA', 'Ete
rnal Spirit', 'Gray', 'Lord_Enz', 'beatiful', 'SoulWind', 'John W', 'UnzipsCrogre', 'Storm Zone', 'Flex OKLM', 'Busht
ush', 'SpookyZ', 'Misterioussaint', 'Ayoukoo', 'ABR', 'Insult', 'Sagiri', 'Storm Zone', 'Tricking', 'Ox the Fox', 'Et
ernal Spirit', 'Empo', 'Lord_Enz', 'SoulWind', 'devin', 'Lusa', 'John W', 'SpookyZ', 'Punny', 'BIHI', 'TDNT', 'Bushtu
sh', 'beatiful', 'Rubyblood', 'MichaelderBeste2', 'Chaitanya', 'eclipseY', 'HSA', 'haxrme', 'Ahsan-219', 'Flex OKLM',
'UnzipsCrogre', 'Gray', 'Tace', 'Misterioussaint', 'Insult', 'SoulWind', 'Empo', 'Tricking', 'Lusa', 'John W', 'Storm
Zone', 'MichaelderBeste2', 'Sagiri', 'Ox the Fox', 'SpookyZ', 'Eternal Spirit', 'Ayoukoo', 'Bushtush', 'Lord_Enz', 'b
eatiful', 'Flex OKLM', 'ABR', 'Gray', 'devin', 'BIHI', 'HSA', 'Ahsan-219', 'Rubyblood', 'Tace', 'Chaitanya', 'eclipse
Y', 'haxrme', 'TDNT', 'UnzipsCrogre', 'Punny', 'Ox the Fox', 'John W', 'MichaelderBeste2', 'Bushtush', 'beatiful', 'd
evin', 'ABR', 'Tricking', 'SoulWind', 'Eternal Spirit', 'Ahsan-219', 'Misterioussaint', 'BIHI', 'Ayoukoo', 'HSA', 'Ru
byblood', 'Gray', 'Sagiri', 'TDNT', 'Chaitanya', 'Punny', 'Flex OKLM', 'Lord_Enz', 'SpookyZ', 'SpookyZ', 'beatiful',
'Sagiri', 'Eternal Spirit', 'Ahsan-219', 'BIHI', 'John W', 'Bushtush', 'TDNT', 'Rubyblood', 'ABR', 'Punny', 'Michaeld
erBeste2', 'SoulWind', 'Ahsan-219', 'Rubyblood', 'Empo', 'Punny', 'Sagiri', 'Misterioussaint', 'Bushtush', 'Ox the Fo
x', 'devin', 'Insult', 'Storm Zone', 'Tricking', 'Lusa', 'beatiful', 'Ahsan-219', 'SoulWind', 'Ox the Fox', 'beatifu
l', 'Sagiri', 'devin', 'Punny', 'Storm Zone', 'Ox the Fox', 'Punny', 'devin', 'SoulWind', 'Ox the Fox', 'SoulWind',
'John W', 'Misterioussaint']
```

Successfully getting all the names of the players, including the process of removing wigglytuff and vileman.

```
In [213]: setnames = set(unames)
          finalnames = list(setnames)
          print(len(finalnames))
          print(finalnames)

32
['Misterioussaint', 'eclipseY', 'Ahsan-219', 'Lord_Enz', 'Eternal Spirit', 'TDNT', 'John W', 'Ox the Fox', 'Chaitany
a', 'HSA', 'Tace', 'Lusa', 'Storm Zone', 'Insult', 'BIHI', 'SoulWind', 'SpookyZ', 'Sagiri', 'ABR', 'MichaelderBeste
2', 'devin', 'Gray', 'beatiful', 'Punny', 'Tricking', 'Empo', 'Flex OKLM', 'Rubyblood', 'UnzipsCrogre', 'Bushtush',
'Ayoukoo', 'haxrme']
```

Getting a list of names of 32 players.

The next thing for me to do is to find the link of all the battles. I noticed that the links of all the battles are in tags <'a', class_= 'link link—external'>. I used the same soup to get every line containing the links. For those lines, the links are after href, and I was able to retrieve the link with the help of an answered post on Stackoverflow. Following the example on Stackoverflow, I used a

for loop to iterate through all the items in linksoup and append the links to an array first. Then I use another for loop to append to a list with links that are all replays of games. As a result, I now have a list of the links to all the replays of the battles.

```
In [214]: linksoup = soup.find_all('a', class_= 'link link--external')

In [215]: print(linksoup)
```

```
575504" rel="nofollow noopener" target="_blank">3</a>, <a class="link link--external" href="https://replay.pokemons
howdown.com/smogtours-gen8ou-576393" rel="nofollow noopener" target="_blank">1</a>, <a class="link link--external"
href="https://replay.pokemonshowdown.com/smogtours-gen8ou-576398" rel="nofollow noopener" target="_blank">2</a>, <a
class="link link--external" href="https://replay.pokemonshowdown.com/smogtours-gen8ou-576402" rel="nofollow noopene
r" target="_blank">3</a>, <a class="link link--external" href="https://replay.pokemonshowdown.com/smogtours-gen8ou-
576805" rel="noopener" target="_blank">1</a>, <a class="link link--external" href="https://replay.pokemonshowdown.c
om/smogtours-gen8ou-576816" rel="nofollow noopener" target="_blank">2</a>, <a class="link link--external" href="htt
ps://replay.pokemonshowdown.com/smogtours-gen8ou-576752" rel="nofollow noopener" target="_blank">1</a>, <a class="l
ink link--external" href="https://replay.pokemonshowdown.com/smogtours-gen8ou-576759" rel="nofollow noopener" targe
t="_blank">2</a>, <a class="link link--external" href="https://replay.pokemonshowdown.com/smogtours-gen8ou-576762"
rel="nofollow noopener" target="_blank">3</a>, <a class="link link--external" href="https://replay.pokemonshowdown.
com/smogtours-gen8ou-576842" rel="nofollow noopener" target="_blank">1</a>, <a class="link link--external" href="ht
tps://replay.pokemonshowdown.com/smogtours-gen8ou-576843" rel="nofollow noopener" target="_blank">2</a>, <a class
="link link--external" href="https://replay.pokemonshowdown.com/smogtours-gen8ou-577082" rel="nofollow noopener" ta
rget="_blank">1</a>, <a class="link link--external" href="https://replay.pokemonshowdown.com/smogtours-gen8ou-57708
4" rel="nofollow noopener" target="_blank">2</a>, <a class="link link--external" href="https://replay.pokemonshowdo
wn.com/smogtours-gen8ou-577275" rel="nofollow noopener" target="_blank">1</a>, <a class="link link--external" href
="https://replay.pokemonshowdown.com/smogtours-gen8ou-577276" rel="nofollow noopener" target="_blank">2</a>, <a cla
ss="link link--external" href="https://replay.pokemonshowdown.com/smogtours-gen8ou-577277" rel="nofollow noopener"
target="_blank">3</a>, <a class="link link--external" href="https://replay.pokemonshowdown.com/smogtours-gen8ou-577
```

Getting all the tags containing links on the thread

```
In [216]: links_temp = []
          for x in linksoup:
              links_temp.append(x['href'])

In [217]: print(links_temp)
```

```
['https://replay.pokemonshowdown.com/smogtours-gen8ou-572330', 'https://replay.pokemonshowdown.com/smogtours-gen8ou
-572331', 'https://www.youtube.com/watch?v=cqNLvrfgw3c', 'https://replay.pokemonshowdown.com/smogtours-gen8ou-57283
3', 'https://replay.pokemonshowdown.com/smogtours-gen8ou-572845', 'https://www.youtube.com/watch?v=cqNLvrfgw3c', 'h
ttps://replay.pokemonshowdown.com/smogtours-gen8ou-572276', 'https://replay.pokemonshowdown.com/smogtours-gen8ou-57
2277', 'https://replay.pokemonshowdown.com/smogtours-gen8ou-572278', 'https://replay.pokemonshowdown.com/smogtours-
gen8ou-572661', 'https://replay.pokemonshowdown.com/smogtours-gen8ou-572664', 'https://replay.pokemonshowdown.com/s
mogtours-gen8ou-572670', 'https://replay.pokemonshowdown.com/smogtours-gen8ou-572172', 'https://replay.pokemonshowd
own.com/smogtours-gen8ou-572173', 'https://www.youtube.com/watch?v=cqNLvrfgw3c', 'https://replay.pokemonshowdown.co
m/smogtours-gen8ou-572313', 'https://replay.pokemonshowdown.com/smogtours-gen8ou-572316', 'https://replay.pokemonsh
owdown.com/smogtours-gen8ou-572319', 'https://replay.pokemonshowdown.com/smogtours-gen8ou-572575', 'https://replay.
pokemonshowdown.com/smogtours-gen8ou-572584', 'https://replay.pokemonshowdown.com/smogtours-gen8ou-572588', 'http
s://replay.pokemonshowdown.com/smogtours-gen8ou-572702', 'https://replay.pokemonshowdown.com/smogtours-gen8ou-57270
6', 'https://www.youtube.com/watch?v=cqNLvrfgw3c', 'https://replay.pokemonshowdown.com/smogtours-gen8ou-572808', 'h
ttps://replay.pokemonshowdown.com/smogtours-gen8ou-572814', 'https://www.youtube.com/watch?v=cqNLvrfgw3c', 'http
s://replay.pokemonshowdown.com/smogtours-gen8ou-572118', 'https://replay.pokemonshowdown.com/smogtours-gen8ou-57211
9', 'https://replay.pokemonshowdown.com/smogtours-gen8ou-572120', 'https://replay.pokemonshowdown.com/smogtours-gen
8ou-572472', 'https://replay.pokemonshowdown.com/smogtours-gen8ou-572490', 'https://www.youtube.com/watch?v=cqNLvrf
gw3c', 'https://replay.pokemonshowdown.com/smogtours-gen8ou-572323', 'https://replay.pokemonshowdown.com/smogtours-
gen8ou-572324', 'https://replay.pokemonshowdown.com/smogtours-gen8ou-572325', 'https://replay.pokemonshowdown.com/s
```

Using href to get all the links

After getting the link to all the replays, I've decided to show the win and losses by using a list of lists with each list having a winner in index 0 and a loser in index 1 because this can be convenient when I'm doing PageRank. However, I overlooked a problem: the names in battles and

```
In [218]: finallinks = []
          for x in links_temp:
              if 'replay' in x:
                  finallinks.append(x)

          print(finallinks)
```

```
['https://replay.pokemonshowdown.com/smogtours-gen8ou-572330', 'https://replay.pokemonshowdown.com/smogtours-gen8ou
-572331', 'https://replay.pokemonshowdown.com/smogtours-gen8ou-572833', 'https://replay.pokemonshowdown.com/smogtou
rs-gen8ou-572845', 'https://replay.pokemonshowdown.com/smogtours-gen8ou-572276', 'https://replay.pokemonshowdown.co
m/smogtours-gen8ou-572277', 'https://replay.pokemonshowdown.com/smogtours-gen8ou-572278', 'https://replay.pokemonsh
owdown.com/smogtours-gen8ou-572661', 'https://replay.pokemonshowdown.com/smogtours-gen8ou-572664', 'https://replay.
pokemonshowdown.com/smogtours-gen8ou-572670', 'https://replay.pokemonshowdown.com/smogtours-gen8ou-572172', 'http
s://replay.pokemonshowdown.com/smogtours-gen8ou-572173', 'https://replay.pokemonshowdown.com/smogtours-gen8ou-57231
3', 'https://replay.pokemonshowdown.com/smogtours-gen8ou-572316', 'https://replay.pokemonshowdown.com/smogtours-gen
```

Getting the links to the replays of all battles.

the user IDs on the smog forum don't necessarily match. So I created a function that makes sure the nicknames in battles will get converted to names in ID after going checking all the replays. After that, I imported request, which I learned on Stackoverflow, so I'm able to open the links of the replays on the smogon forum and get data out of them. For all the replays, I apply beautiful soup and find both players names first. Then, I go into the battle log and extract that line from that shows who wins. I'll append an list to the total edges in which the first item of the list is the loser and the second is the winner. In fact, I've tested it in the opposite order but the order of loser first then winner seems more accurate. After that, we can import the players names as nodes and the wins and losses list of lists as edges to the graph and I can get the rankings of each players.

```
In [224]: def matchPlayerID(name):
              if(name == 'Sagiri I.' or name == 'Sagiri I'):
                  return 'Sagiri'
              elif(name == 'Lopunny kicks'):
                  return "Punny"
              elif(name == 'devwin'):
                  return 'devin'
              elif(name == 'Tace🐶'):
                  return 'Tace'
              elif(name == 'SCILLATRON9000' or name == 'SCILLATRON 9000' or name == 'uhuhuhu2'):
                  return 'Tricking'
              elif(name == 'INSULT'):
                  return 'Insult'
              elif(name == 'Ayouko'):
                  return 'Ayoukoo'
              elif(name == 'BIHI Gregson' or name == 'BIHI Asogi' or name == 'Cotisation URSSAF'):
                  return 'BIHI'
              elif(name == 'EclipseY'):
                  return 'eclipseY'
              elif(name == 'misterioussaint'):
                  return 'Misterioussaint'
              elif(name == 'hsa'):
                  return 'HSA'
              elif(name == 'imjustgray'):
                  return 'Gray'
              elif(name == 'LordEnz'):
                  return 'Lord_Enz'
              elif(name == 'Bbeeaa' or name == 'FREE XIMRAPTOR' or name == 'Freeximraptor'):
                  return 'beatiful'
              elif(name == 'SOULWIND'):
                  return 'SoulWind'
              elif(name == 'john w'):
                  return 'John W'
              elif(name == 'storm zone'):
                  return 'Storm Zone'
              else:
                  return name
```

Function to match nicknames in replays and usernames on the forum

```
In [227]: import networkx as nx

          G = nx.DiGraph()
          G.add_nodes_from(finalnames)
          G.add_edges_from(edges)

In [228]: PRank = nx.pagerank(G, alpha=0.85, personalization=None, weight='weight', dangling=None)

In [229]: print(PRank)

          {'Misterioussaint': 0.02549965177329615, 'eclipseY': 0.009022649901286463, 'Ahsan-219': 0.03160954918986624, 'Lord_En
          z': 0.021747604852309147, 'Eternal Spirit': 0.03303380571697398, 'TDNT': 0.025928670934111803, 'John W': 0.0281133653
          63534037, 'Ox the Fox': 0.04436152039609455, 'Chaitanya': 0.018713187791268952, 'HSA': 0.013895543857272609, 'Tace':
          0.00933498250829307, 'Lusa': 0.04349759555810308, 'Storm Zone': 0.0672786447495486, 'Insult': 0.03005751430513458, 'B
          IHI': 0.029936072361592465, 'SoulWind': 0.058932448928755335, 'SpookyZ': 0.014283127653085655, 'Sagiri': 0.0380197378
          0668112, 'ABR': 0.042462647915019705, 'MichaelderBeste2': 0.040416163769665, 'devin': 0.039463677709693015, 'Gray':
          0.020033363261949478, 'beatiful': 0.05053089513200319, 'Punny': 0.07114100345622383, 'Tricking': 0.0474117254184601,
          'Empo': 0.03883544300020633, 'Flex OKLM': 0.02507824545987253, 'Rubyblood': 0.027338199181903146, 'UnzipsCrogre': 0.0
          04687500000000001, 'Bushtush': 0.023406739632385303, 'Ayoukoo': 0.014532384520241727, 'haxrme': 0.011396337895168654}
```

Using PageRank Algorithm

To calculate win loss ratio, I created two dictionaries out of the username list to record wins and losses respectively. After that, I create another dictionary called dictWinLoss and all the values in it are wins divided by losses, and it will be the final outcome of of this method.

```
In [232]: dictWin = {i : 0 for i in finalnames}
          dictLoss = {i : 0 for i in finalnames}
          for x in edges:
              dictLoss[x[0]] = dictLoss[x[0]] + 1
              dictWin[x[1]] = dictWin[x[0]] + 1

In [233]: print(dictWin)

          {'Misterioussaint': 8, 'eclipseY': 1, 'Ahsan-219': 11, 'Lord_Enz': 4, 'Eternal Spirit': 6, 'TDNT': 7, 'John W': 4, 'O
          x the Fox': 12, 'Chaitanya': 4, 'HSA': 4, 'Tace': 2, 'Lusa': 7, 'Storm Zone': 10, 'Insult': 4, 'BIHI': 8, 'SoulWind':
          11, 'SpookyZ': 5, 'Sagiri': 9, 'ABR': 8, 'MichaelderBeste2': 8, 'devin': 10, 'Gray': 3, 'beatiful': 10, 'Punny': 13,
          'Tricking': 7, 'Empo': 8, 'Flex OKLM': 1, 'Rubyblood': 8, 'UnzipsCrogre': 0, 'Bushtush': 5, 'Ayoukoo': 3, 'haxrme':
          2}

In [234]: print(dictLoss)

          {'Misterioussaint': 7, 'eclipseY': 4, 'Ahsan-219': 10, 'Lord_Enz': 7, 'Eternal Spirit': 6, 'TDNT': 7, 'John W': 2, 'O
          x the Fox': 4, 'Chaitanya': 6, 'HSA': 4, 'Tace': 6, 'Lusa': 3, 'Storm Zone': 4, 'Insult': 4, 'BIHI': 7, 'SoulWind':
          8, 'SpookyZ': 7, 'Sagiri': 7, 'ABR': 7, 'MichaelderBeste2': 6, 'devin': 7, 'Gray': 6, 'beatiful': 8, 'Punny': 9, 'Tri
          cking': 5, 'Empo': 4, 'Flex OKLM': 6, 'Rubyblood': 8, 'UnzipsCrogre': 6, 'Bushtush': 7, 'Ayoukoo': 7, 'haxrme': 6}
```

Two dictionaries to keep track of each player's wins and losses

```
In [235]: dictWinLoss = {i : 0 for i in finalnames}
          for key in dictWinLoss:
              dictWinLoss[key] = dictWin[key]/dictLoss[key]

In [236]: print(dictWinLoss)

{'Misterioussaint': 1.1428571428571428, 'eclipseY': 0.25, 'Ahsan-219': 1.1, 'Lord_Enz': 0.5714285714285714, 'Eternal
Spirit': 1.0, 'TDNT': 1.0, 'John W': 2.0, 'Ox the Fox': 3.0, 'Chaitanya': 0.6666666666666666, 'HSA': 1.0, 'Tace': 0.3
333333333333333, 'Lusa': 2.3333333333333335, 'Storm Zone': 2.5, 'Insult': 1.0, 'BIHI': 1.1428571428571428, 'SoulWin
d': 1.375, 'SpookyZ': 0.7142857142857143, 'Sagiri': 1.2857142857142858, 'ABR': 1.1428571428571428, 'MichaelderBeste
2': 1.3333333333333333, 'devin': 1.4285714285714286, 'Gray': 0.5, 'beatiful': 1.25, 'Punny': 1.4444444444444444, 'Tri
cking': 1.4, 'Empo': 2.0, 'Flex OKLM': 0.16666666666666666, 'Rubyblood': 1.0, 'UnzipsCrogre': 0.0, 'Bushtush': 0.7142
857142857143, 'Ayoukoo': 0.42857142857142855, 'haxrme': 0.3333333333333333}
```

Dictionary with win/loss ratio

Results.

```
In [201]: sorted(PRank.items(), key=lambda x: x[1])

Out[201]: [('UnzipsCrogre', 0.004687500000000001),
           ('eclipseY', 0.009022649901286461),
           ('Tace', 0.00933498250829307),
           ('haxrme', 0.011396337895168654),
           ('HSA', 0.013895543857272609),
           ('SpookyZ', 0.014283127653085656),
           ('Ayoukoo', 0.014532384520241725),
           ('Chaitanya', 0.018713187791268952),
           ('Gray', 0.020033363261949478),
           ('Lord_Enz', 0.021747604852309147),
           ('Bushtush', 0.023406739632385307),
           ('Flex OKLM', 0.02507824545987253),
           ('Misterioussaint', 0.02549965177329615),
           ('TDNT', 0.025928670934111803),
           ('Rubyblood', 0.027338199181903142),
           ('John W', 0.028113365363534037),
           ('BIHI', 0.02993607236159247),
           ('Insult', 0.03005751430513458),
           ('Ahsan-219', 0.03160954918986624),
           ('Eternal Spirit', 0.03303380571697398),
           ('Sagiri', 0.03801973780668112),
           ('Empo', 0.03883544300020633),
           ('devin', 0.039463677709693015),
           ('MichaelderBeste2', 0.040416163769665),
           ('ABR', 0.042462647915019705),
           ('Lusa', 0.04349759555810308),
           ('Ox the Fox', 0.04436152039609455),
           ('Tricking', 0.0474117254184601),
           ('beatiful', 0.05053089513200319),
           ('SoulWind', 0.05893244892875535),
           ('Storm Zone', 0.0672786447495486),
           ('Punny', 0.07114100345622383)]
```

The Sorted dictionary for PageRank Algorithm

For the PageRank Algorithm, we can see that UnzipsCrogre has the lowest rank, which matches what happened in the tournament as UnzipsCrogre didn't win a single game throughout the playoffs. For other players, the results seems pretty accurate as well. Surprisingly, the winner of the tournament Ox the Fox is only the sixth highest ranking player. On the other hand, Punny, who is eliminated during quarterfinals, is the highest ranked.

```
orted(dictWinLoss.items(), key=lambda x: x[1])

('UnzipsCrogre', 0.0),
('Flex OKLM', 0.16666666666666666),
('eclipseY', 0.25),
('Tace', 0.3333333333333333),
('haxrme', 0.3333333333333333),
('Ayoukoo', 0.42857142857142855),
('Gray', 0.5),
('Lord_Enz', 0.5714285714285714),
('Chaitanya', 0.6666666666666666),
('SpookyZ', 0.7142857142857143),
('Bushtush', 0.7142857142857143),
('Eternal Spirit', 1.0),
('TDNT', 1.0),
('HSA', 1.0),
('Insult', 1.0),
('Rubyblood', 1.0),
('Ahsan-219', 1.1),
('Misterioussaint', 1.1428571428571428),
('BIHI', 1.1428571428571428),
('ABR', 1.1428571428571428),
('beatiful', 1.25),
('Sagiri', 1.2857142857142858),
('MichaelderBeste2', 1.3333333333333333),
('SoulWind', 1.375),
('Tricking', 1.4),
('devin', 1.4285714285714286),
('Punny', 1.4444444444444444),
('John W', 2.0),
('Empo', 2.0),
('Lusa', 2.3333333333333335),
('Storm Zone', 2.5),
('Ox the Fox', 3.0)]
```

Sorted dictionary for win/loss ratio

For win/loss ratio, the tournament winner Ox the Fox does get the highest out of all. Nevertheless, the runner up SoulWind is only ranked 9th, implying that SoulWind perhaps has taken more losses early or have won more third games in bo3s.

Conclusion.

Both methods are merely relative measures of each player's performances but can't directly tell who will win the tournament or which player is eliminated earlier. They are somewhat accurate as losing and winning are the biggest factors for both methods, and losing more is more likely to get a player eliminated sooner and winning more is more likely to get a player farther in a tournament. However, they are not absolute. First, tournament goes in best of three formats, which means a player who wins in a lot of game threes may not be ranked for win/loss ratio and PageRank as high but still go deep in this tournament. Also, players are allowed to lose less than three sets before Top 16 for 5 rounds. Last but not least, accidents happen as one of the players, John W, gets eliminated

early because he got banned during the tour, but John's win loss ratio is still high. For pagerank, it is relevant about which players lost to which, this factor allows the results of two methods to differ drastically.

References

JimShapedCoding. (2020, November 18). *Web scraping with python - beautiful soup crash course*. YouTube. Retrieved from https://www.youtube.com/watch?v=XVv6mJpFOb0&ab_channel=freeCodeCamp.org

Beautiful Soup documentation¶. Beautiful Soup Documentation - Beautiful Soup 4.9.0 documentation. (n.d.). Retrieved from https://www.crummy.com/software/BeautifulSoup/bs4/doc/

dkgirldkgirl 4, & Mark LongairMark Longair 410k7070 gold badges400400 silver badges320320 bronze badges. (2011, March 28). *Beautifulsoup getting href*. Stack Overflow. Retrieved from https://stackoverflow.com/questions/5815747/beautifulsoup-getting-href

Helen NeelyHelen Neely 4, woozykingwoozyking 4, i.n.n.mi.n.n.m 2, JaredJared 72011 gold badge1010 silver badges1919 bronze badges, Martin ThomaMartin Thoma 104k132132 gold badges541541 silver badges830830 bronze badges, Akash KinwadAkash Kinwad 63477 silver badges1919 bronze badges, ARVIND CHAUHANARVIND CHAUHAN 3522 bronze badges, ATOzTOAATOzTOA 32.4k2121 gold badges8989 silver badges115115 bronze badges, ggglniggglni 6933 bronze badges, ksonoksono 1, & 荷兰哲学家Elvira荷兰哲学家Elvira 9. (1961, January 1). *How can I read the contents of an URL with python?* Stack Overflow. Retrieved from https://stackoverflow.com/questions/15138614/how-can-i-read-the-contents-of-an-url-with-python