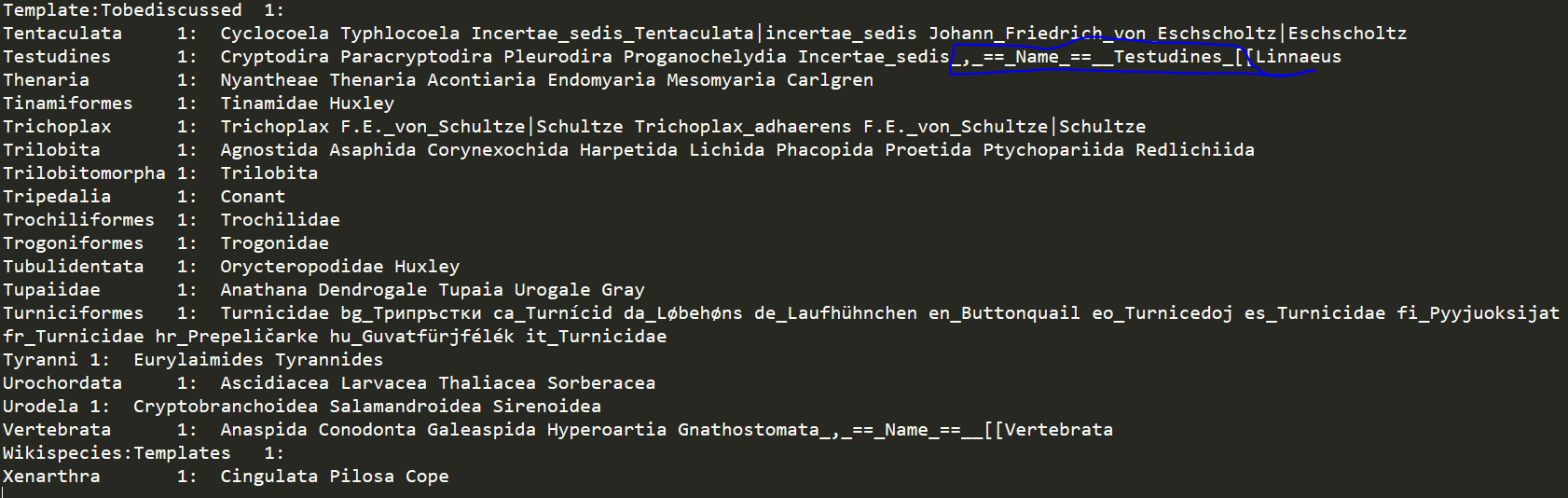
**Assignment 4**

**Part 1 – Stand Alone Application**

1. Graph Builder **(1.0 is given as a seed.)**
   1. Mapper / Reducer / Driver

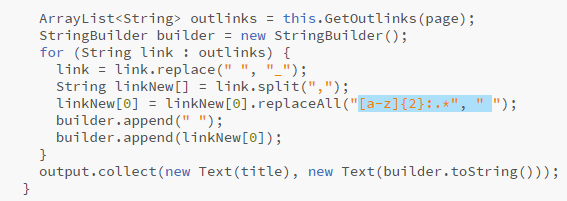
**O/P**



**Here we can see the inlinks and outlinks but all those are covered with dirty data.**

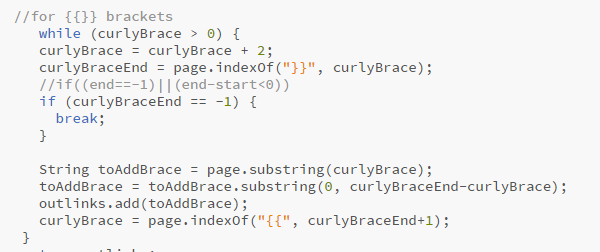
* 1. Writing Regular Expression in order to clean data.

**Code- :**

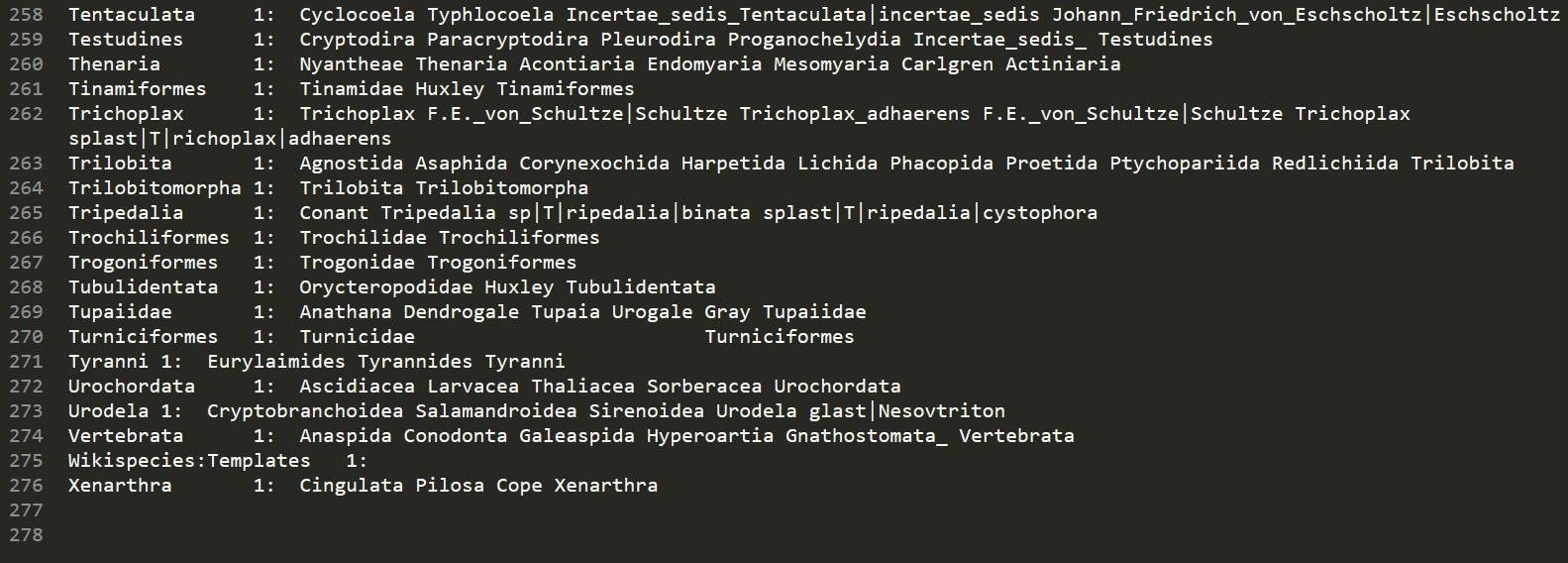


As we have to take all that values that comes under {{}} braces.

Writing code -:

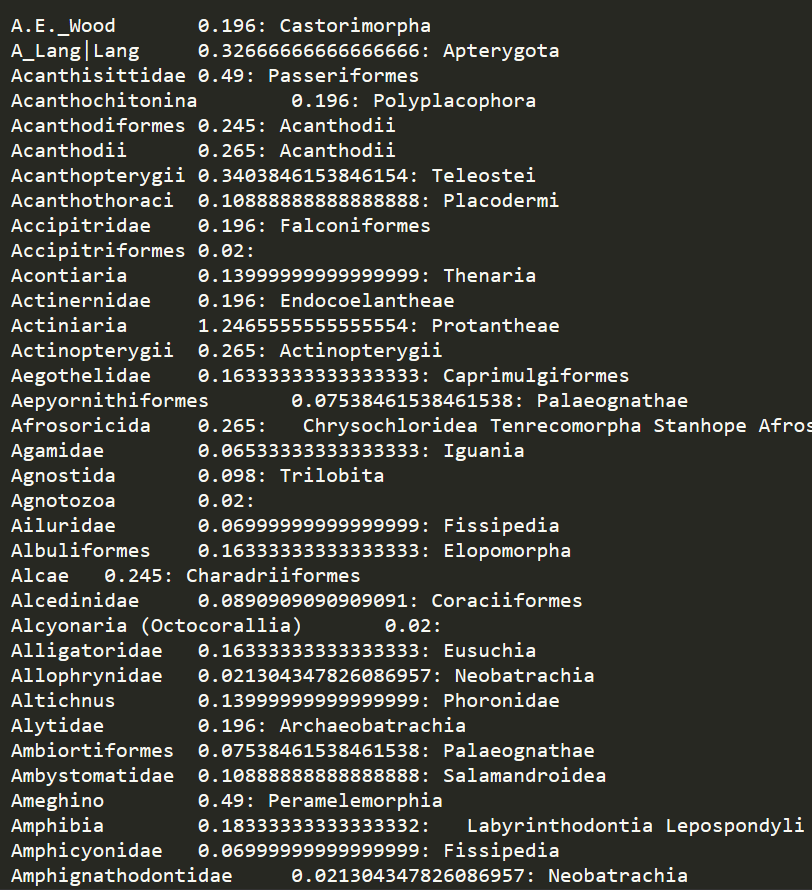


After writing this data get cleaned

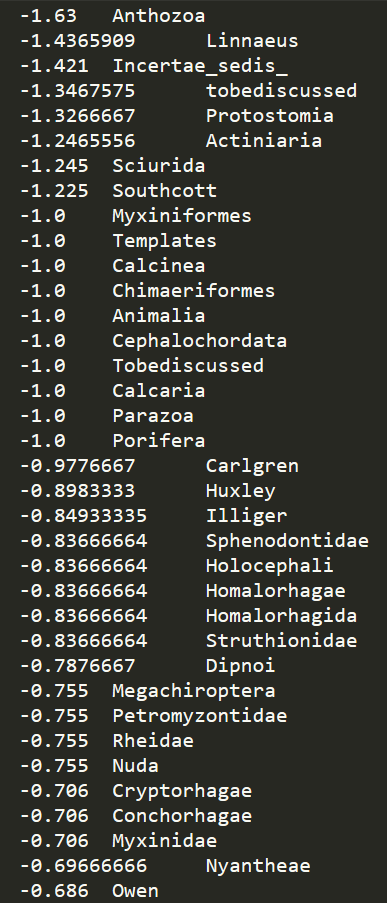


This shows Data is cleaned.

1. **Iterator (Taking Damping Factor as .98)**
2. Giving o/p of Graph Builder as an Input to iterator.
3. After running Mapper / Reducer / Runner
4. Got Output as-:

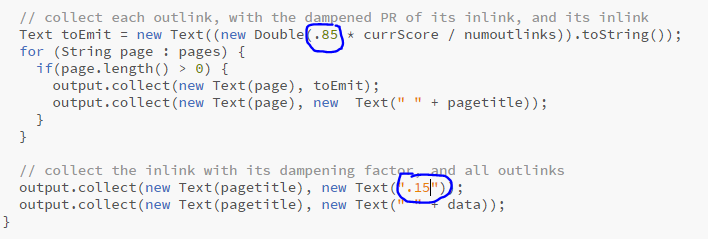


1. **Viewer**
   1. Running Mapper and Driver in order to see Page Rank.



**Part 2 –**

1. **Taking Damping Factor as .85**



1. **Iterating 250 times to check when the page rank deltas are converging.**

**Showing Iteration up to 250.**



**Converging -:**

**Checking some files from stating.**

**For species name SouthCott**

**Iterator 1-:** Southcott 1.0625

**Iterator 3-:** Southcott 0.5415834635416666

**Iterator 5-:** Southcott 0.22

**Iterator 7-:** Southcott 0.37133257786539714

**Iterator 10-:** Southcott 0.1403339996421875

**Iterator 20-:** Southcott 0.02

**Iterator 21-:** Southcott 0.02

**Iterator 22-:** Southcott 0.02

**Iterator 23-:** Southcott 0.02

**Iterator 24-:** Southcott 0.02

**Iterator 27-:** Southcott 0.19921069992135967

**Iterator 200-:** Southcott 0.02

**Iterator 205-:** Southcott 0.02

**Iterator 210-:** Southcott 0.02

As we can see from here for species Southcott the for starting some of the Iterators it was different but from 20 to 24 its constant. Then it gets change for some values and then it became constant.

We can say it is converging.

**Part 3 -: Viewer**

1. **Calculating Page rank for some Pages and comparing.**

**After running Iterator 20**

**We can see page rank of Southcott**



**Note-: All the code is attached in the folder with specific outputs.**

**OPTIONAL - Extra Credits: Predict long-term dynamics of a Markovian system of your choice.**

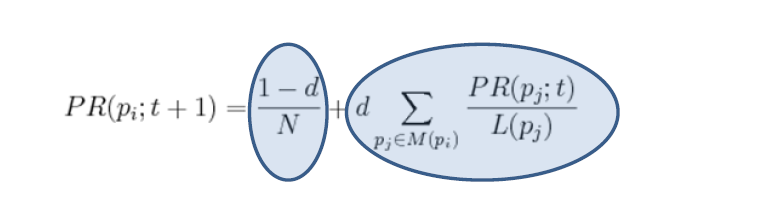
**I am choosing Page Rank Algorithm as a Markovain System**

A Markov system is a system that can be in one of several numbered states and can be pass from one state to another each time step according to fixed probabilities.

If a Markov System is in state**i** there is a fixed probability, Pij of it going into state j the next time step, and Pij is called transition probability. Google Page Rank is one of the example of Markov System.

**Page Rank Algorithm**

For page rank algorithm, we start with a number of web pages, where each page points to another web page. Each web page pointing towards a another web page is considered a vote for the another web page. More the count for a web page, the higher the rank for the web page. Google came with a formula for calculating the web page

****

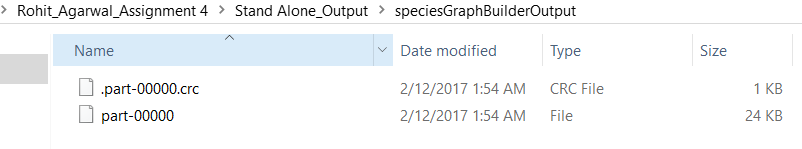
We are going to compute the Page Rank for a web page using map reduce. We are going to being with **(URL, list-of-URLs)**where list-of-URLs is the outlinks from the webpage. We are going to transform the above expression to **(URL, (PageRank, list-of-URLs))** where **PageRank is the initial guess, usually 0.5 or 0.1**.

The above expression will act as input to the **Map**, the map for each ***u*** in the **list-of-URLs**will output **(u, URL, PR(URL)/Length of list-of-URLs)**i.e., the output link, with the voter, the page rank of the voter divided by number of output urls from the web page, the large the page rank of the URL, the more support to the web page will be give and we are going to output the **(URL, list-of-URLs)**.

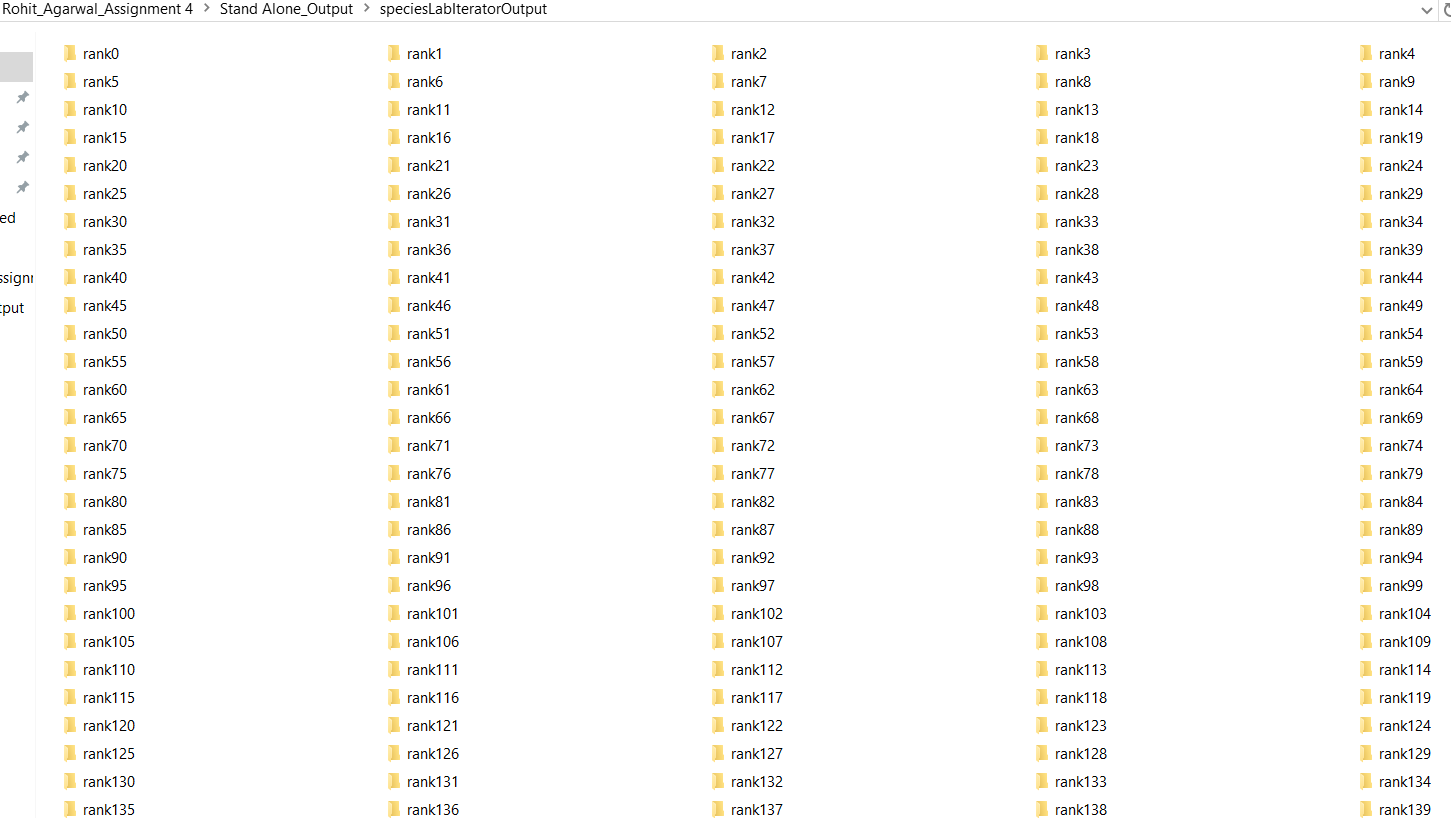
Each Web page’s hyperlinks (outlinks, or votes) are “pivoted”, and we now tally up each Web page’s inlinks (voters) instead. In the **Reducer**, the input will be (URL, list-of-URLs) and many URLs pairs and it going to calculate the **(URL, (new PR, list-of-URLs))**. We are going to aggregate the votes from each voter and then going to multiply the aggregated value with the dampening factor. We are also going to add the the probability of the direc typing the URL and reaching the web page. The sum of the above expression will be the page rank of the web page. We are going to iterate through this mapper and reducer part couple of times and will observe that page rank is converging, after a while page rank will be constant and that will be the final page rank of the web page.

**Screen Short showing All Outputs -:**

1. **Graph Builder Output**



1. **Iterator Output**



**It goes through 0 to 249.**

1. **Viewer Output**

