

PA1 Google Search Engine Simulator Report

JianBin Wu

Design and Implementation

To design and implement a Google Search Engine Simulator, I used four separate classes, WebCrawler, PageRank, HeapSort, Urls, and GoogleSearchEngineSimulator. The WebCrawler class was given by our professor, its main function is to crawl the first 30 urls from the google search. PageRank class generates rank scores for a specific page. The rank score is based on the frequency and location of keywords within the web page, how long the web page has existed, the number of other web pages that link to the page in question, and how much the webpage owner has paid to Google for advertisement purposes. In order to generate a random score, I imported Random class. The PageRank class includes get() methods for each 4 score factors as well as the total score. Since the assignment requires heapsort to sort the urls, I also implement Heap Sort and Heap Priority Queue algorithms by following the pseudo codes from the book "Introduction To Algorithms Third Edition." I also created a class called Urls to store website string and PageRank object. Urls class has get() methods to get the PageRank scores and url string.

The GoogleSearchEngineSimulator class includes main() to run the program. I used ArrayList to store Urls objects, so the url string and pageRank are linked together. I used Arrays to store scores that were generated from PageRank and also used it for Heap Priority Queue. Following the requirement of the assignment, I Implemented Heap priority queue to store the first sorted 20 out of 30 web url links into a heap by calling the maxHeapInsert() method from HeapSort class. Next, I implement a method named insertWebsite(), it allows users to insert a new url into Heap priority queue by implementing maxHeapInsert() from the HeapSort class. Moreover, I implement a method named firstRankAndRemove() that allows users to view the first ranked web url link by implementing HeapExtractMax() from HeapSort class. Finally, I implemented a method named increaseRankScore(), it allow users to choose a url links stored in the Heap Priority Queue and increase its PageRank score and its priority in the queue to be listed by implementing HeapIncreaseKey() from HeapSort class.

Classes, Subroutines and Function calls

WebCrawler class - A web crawler to collect urls from Google Search engine using keywords.

- search() - This method start the search.
- getDomainName(String url) - The method will use pattern and matcher to extract the domain.
- getUrls() - This method get the set of urls result.
- crawl(String url) - This method will crawl the links and put them into a set to keep.
- searchForWord(String searchWord) - This method will check if the website contains a keyword.

PageRank class - This class rank with numerical score for each url page by the following: the frequency and location of keywords within the web page, how long the web page has existed, the number of other web pages that link to the page in question, and how much the webpage owner has paid to Google for advertisement purpose. The score of the urls are generated by a random number from 1 - 100.

- PageRank() - Construct a RankPage object with 4 random score factors.
- PageRank(int keyword , int ads, int page , int web) - Construct a RankPage object with 4 score factors (not random).
- numberOfFrequency() - Generate a random score for the frequency and location of keywords within the web page.
- timeExisted() - Generate a random score for how long the web page has existed.
- numberWebLinks() -Generate a random score for the number of other web pages that link to the page in question.
- adsPriority() - Generate a random score for how much the webpage owner has paid to Google for advertisement purposes.
- getFrequency() - Get score for the frequency and location.
- getTimeExisted() - Get score for how long the web page has existed.
- getNumberWebLinks() - Get score for number of other web pages.
- getAdsPriority() - Get score for AdsPriority.
- getTotalScore() - Get the total rank score.
- rank() - sum up of all four scores.
- higherPriority(int money) - the amount of money that Webpage owners paid to increase webpage's exposure for advertisement purposes.

HeapSort class - Implementation of Heap Sort and Heap Priority Queue algorithm, following the pseudo codes from the book Introduction To Algorithms Third Edition.

- parent(int i) - Return the parent of node i.
- left(int i) - Return the left child of node i.
- right (int i) - Return the right child of node i.
- maxHeapify(int A[],int i) - Maintain the max-heap property of a node

- buildMaxHeap(int A[]) - Convert the array into a max-heap.
- heapSort(int A[]) - Sort the elements in the array by maintaining max heap property.
- maxHeapInsert(int A[], int key) - Insert an element key into a max-heap.
- heapExtractMax(int A[]) - Return and remove the largest element in the max-heap.
- heapIncreaseKey(int A[], int i, int key) - Increase the value of the element to a larger value in a max-heap.
- heapMaximum(int A[]) - Return the maximum element from the max-heap (root).

Urls class - A Urls object contain url strings and PageRank object

- Urls(String url , PageRank pageRank)
- getFrequency() - Get score for the frequency and location of the PageRank object.
- getTimeExisted() - Get score for how long the web page has existed of the PageRank object.
- getNumberWebLinks() - Get score for number of other web pages of the PageRank object.
- getAdsPriority() - Get score for AdsPriority of the PageRank object.
- getTotalScore() - Get the total score of the PageRank object.
- getPageRank() - Get PageRank object.
- getUrls()- Get the website url.

GoogleSearchEngineSimulator class - A Google Search Engine Simulator

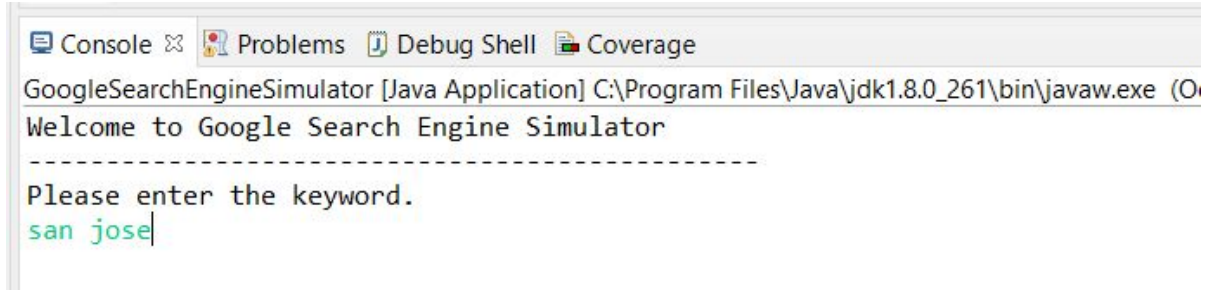
- main() - runs the simulator
- repeatPromote() - Promote user to select options
- heapPriorityQueue() - Heap Priority Queue with the first 20 websites
- insertWebsite(String url,int keyword , int ads, int page , int web) - Insert a website to the list with an url and 4 scores.
- increaseRankScore(int index, int money) - Increase a websites' rank score by adding money.
- firstRankAndRemove() - view the first rank url and remove it from the list.
- reverseArray (int A[]) - Reverse an array.

- syncLinkAndScore() - sync the heapWebsites list and heapPriority. Also, print out the updated heap websites list
- HeapScoreInfo(int i) - Get other 4 scores info of a heap website

Self-testing Screenshots

The crawling process

The program will begin with promoting users to enter keywords to search. In this test case, I entered the keyword "san jose" as an example.



```

GoogleSearchEngineSimulator [Java Application] C:\Program Files\Java\jdk1.8.0_261\bin\javaw.exe (O
Welcome to Google Search Engine Simulator
-----
Please enter the keyword.
san jose

```

A WebCrawler object will be constructed and take the keyword as its parameters. Next, it will call search() method and collect a list of websites based on the keyword. I store the first 30 results into an arraylist of Urls objects. I use an enhanced loop to print out the output. Inside the enhanced loop, a RankPage object will be constructed to assign rank score for each Urls. Using counter to stop generating since the result is more than 30 url links. My code shown as below:

```

//main method to run the program
public static void main (String args[]) {
    //Promote the user to enter keywords
    System.out.println("Welcome to Google Search Engine Simulator");
    System.out.println("-----");
    System.out.println("Please enter the keyword.");
    Scanner scanner = new Scanner(System.in);
    String keyword = scanner.nextLine();

    //search keyword with crawler
    WebCrawler crawler = new WebCrawler (keyword);
    crawler.search();

    //print results from crawler
    ArrayList<Urls> websites = new ArrayList<Urls>();
    int counter = 1;
    for (String w : crawler.getUrls()) {
        PageRank a = new PageRank();
        websites.add(new Urls(w,a));
        System.out.println(counter + "." + w + " RankScore: " + a.getTotalScore());
        counter++;
        if(counter == 31 ) {
            break;
        }
    }
    System.out.println("Total: " + (counter-1));
    System.out.println("-----");
}

```

Output:

Please enter the keyword.

san jose

Visiting Received web page at <https://google.com/search?q=san+jose&num=80>

Found (139) links

Searching for the word san jose...

Success Word san jose found at <https://google.com/search?q=san+jose&num=80>

Done Visited 92 web page(s)

Here are the first 30 URL links:

1. www.sjUSD.org/&sa=U&ved=2ah RankScore: 202
2. www.sjmusart.org/&sa=U&ved=2ahU RankScore: 227
3. www.sjsu.edu/&sa=U&ved=2ahU RankScore: 206
4. www.csZsanjose.com/&sa=U&ved=2a RankScore: 171
5. www.cnn.com/2020/10/22/us/s RankScore: 197
6. sanjosetheaters.org/&sa=U&v RankScore: 244
7. realestate.usnews.com/place RankScore: 146
8. www.cdm.org/&sa=U&ved=2ahUK RankScore: 249
9. www.visitcalifornia.com/pla RankScore: 166
10. www.sanjose.org/&sa=U&ved=2 RankScore: 292
11. sjsuspartans.com/&sa=U&ved= RankScore: 253
12. www.operasj.org/&sa=U&ved=2 RankScore: 272
13. en.wikipedia.org/wiki/Santa RankScore: 196
14. www.ndsj.org/&sa=U&ved=2ahU RankScore: 166
15. www.sjica.org/&sa=U&ved=2ah RankScore: 257
16. en.wikipedia.org/wiki/Timel RankScore: 300
17. www.city-data.com/city/San- RankScore: 174
18. www.japantownsanjose.org/&s RankScore: 175
19. www.redfin.com/city/17420/C RankScore: 206
20. www.sanjosehotel.com/&sa=U& RankScore: 185
21. www.fairmont.com/san-jose/& RankScore: 134
22. worldpopulationreview.com/u RankScore: 200
23. en.wikipedia.org/wiki/Histo RankScore: 234
24. www.zillow.com/san-jose-ca/ RankScore: 167
25. collegefootballnews.com/202 RankScore: 195
26. www.sjquiltmuseum.org/&sa=U RankScore: 122
27. www.nbcbayarea.com/tag/san- RankScore: 139
28. en.wikipedia.org/wiki/Saint RankScore: 171
29. broadwaysanjose.com/&sa=U&v RankScore: 203
30. www.facebook.com/CityofSanJ RankScore: 247

Total: 30

Please press s to sort

Next, I promote the user to sort the websites using heapsort. I used a while loop to guide the user enter letter “s” to sort the websites. After the user entered “s”, I used an array named *scoreArray* to store the rank score of the websites. Then, sort the *scoreArray* using *heapSort()*

method in HeapSort class. Next, reverse the array using reverse() method since headSort() sorted the array in ascending order. Finally, sync the order of *sortedWebsites* and *scoreArray* by comparing the scores in *sortedWebsites* with scores in *scoreArray*. I also print out the sorted website list and their four score factors.

```
//Promote the user to sort the websites
System.out.println("Please press s to sort");
String sortInput = scanner.nextLine().toLowerCase();
while(!sortInput.equals("s"))
{
    System.out.println("press s please");
    sortInput = scanner.nextLine().toLowerCase();
}
if(sortInput.equals("s")){
    //store scores into an array
    for(int i = 0 ; i < 30 ; i++) {
        scoreArray[i] = websites.get(i).getTotalScore();
    }

    //sort website with heapsort
    sorter.heapSort(scoreArray);
    //invert sorted array so the websites in largest to least order
    reverseArray(scoreArray);

    //sync the score with websites
    for(int i = 0; i < 30; i++) {
        for(int j = 0 ; j < websites.size(); j++) {
            if(scoreArray[i] == websites.get(j).getTotalScore()) {
                sortedWebsites.add(new Urls(websites.get(j).getUrls(),websites.get(j).getPageRank()));
                websites.remove(j);
            }
        }
    }
    //print every websites from the list and their other four scores.
    for(int i = 0; i < 30; i++) {
        System.out.println(i+1 + "." + sortedWebsites.get(i).getUrls() + " RankScore: " + sortedWebsites.get(i).getTotalScore()
            + " [Frequency Score: " + sortedWebsites.get(i).getFrequency()
            + ", Existence Score: " + sortedWebsites.get(i).getTimeExisted()
            + ", WebLinks Score: " + sortedWebsites.get(i).getNumberWebLinks()
            + ", AdsPriority Score: " + sortedWebsites.get(i).getAdsPriority() + "]);
    }
}
```


Output:

```
-----
Please press s to sort
s
1.en.wikipedia.org/wiki/Time1 RankScore: 300 [Frequency Score: 84, Existence Score: 79, WebLinks Score: 93, AdsPriority Score: 44]
2.www.sanjose.org/&sa=U&ved=2 RankScore: 292 [Frequency Score: 56, Existence Score: 38, WebLinks Score: 100, AdsPriority Score: 98]
3.www.operasj.org/&sa=U&ved=2 RankScore: 272 [Frequency Score: 96, Existence Score: 10, WebLinks Score: 76, AdsPriority Score: 90]
4.www.sjica.org/&sa=U&ved=2ah RankScore: 257 [Frequency Score: 50, Existence Score: 80, WebLinks Score: 52, AdsPriority Score: 75]
5.sjsuspartans.com/&sa=U&ved= RankScore: 253 [Frequency Score: 47, Existence Score: 72, WebLinks Score: 95, AdsPriority Score: 39]
6.www.cdm.org/&sa=U&ved=2ahUK RankScore: 249 [Frequency Score: 64, Existence Score: 45, WebLinks Score: 68, AdsPriority Score: 72]
7.www.facebook.com/CityofSanJ RankScore: 247 [Frequency Score: 32, Existence Score: 82, WebLinks Score: 58, AdsPriority Score: 75]
8.sanjosetheaters.org/&sa=U&v RankScore: 244 [Frequency Score: 9, Existence Score: 94, WebLinks Score: 50, AdsPriority Score: 91]
9.en.wikipedia.org/wiki/Histo RankScore: 234 [Frequency Score: 15, Existence Score: 94, WebLinks Score: 100, AdsPriority Score: 25]
10.sjmusart.org/&sa=U&ved=2ahU RankScore: 227 [Frequency Score: 85, Existence Score: 13, WebLinks Score: 53, AdsPriority Score: 76]
11.www.sjsu.edu/&sa=U&ved=2ahU RankScore: 206 [Frequency Score: 65, Existence Score: 68, WebLinks Score: 7, AdsPriority Score: 66]
12.www.redfin.com/city/17420/C RankScore: 206 [Frequency Score: 26, Existence Score: 37, WebLinks Score: 63, AdsPriority Score: 80]
13.broadwaysan jose.com/&sa=U&v RankScore: 203 [Frequency Score: 72, Existence Score: 45, WebLinks Score: 83, AdsPriority Score: 3]
14.www.sjisd.org/&sa=U&ved=2ah RankScore: 202 [Frequency Score: 59, Existence Score: 15, WebLinks Score: 67, AdsPriority Score: 61]
15.worldpopulationreview.com/u RankScore: 200 [Frequency Score: 46, Existence Score: 15, WebLinks Score: 58, AdsPriority Score: 81]
16.www.cnn.com/2020/10/22/us/s RankScore: 197 [Frequency Score: 23, Existence Score: 80, WebLinks Score: 37, AdsPriority Score: 57]
17.en.wikipedia.org/wiki/Santa RankScore: 196 [Frequency Score: 94, Existence Score: 27, WebLinks Score: 54, AdsPriority Score: 21]
18.collegefootballnews.com/202 RankScore: 195 [Frequency Score: 8, Existence Score: 95, WebLinks Score: 90, AdsPriority Score: 2]
19.www.sanjoselhotel.com/&sa=U& RankScore: 185 [Frequency Score: 42, Existence Score: 51, WebLinks Score: 78, AdsPriority Score: 14]
20.www.japantownsanjose.org/&s RankScore: 175 [Frequency Score: 88, Existence Score: 73, WebLinks Score: 4, AdsPriority Score: 10]
21.www.city-data.com/city/San- RankScore: 174 [Frequency Score: 53, Existence Score: 6, WebLinks Score: 57, AdsPriority Score: 58]
22.cszsansjose.com/&sa=U&ved=2a RankScore: 171 [Frequency Score: 36, Existence Score: 48, WebLinks Score: 2, AdsPriority Score: 85]
23.en.wikipedia.org/wiki/Saint RankScore: 171 [Frequency Score: 42, Existence Score: 5, WebLinks Score: 65, AdsPriority Score: 59]
24.www.zillow.com/san-jose-ca/ RankScore: 167 [Frequency Score: 34, Existence Score: 14, WebLinks Score: 82, AdsPriority Score: 37]
25.www.visitcalifornia.com/pla RankScore: 166 [Frequency Score: 34, Existence Score: 58, WebLinks Score: 4, AdsPriority Score: 70]
26.www.ndsj.org/&sa=U&ved=2ahU RankScore: 166 [Frequency Score: 51, Existence Score: 81, WebLinks Score: 27, AdsPriority Score: 7]
27.realestate.usnews.com/place RankScore: 146 [Frequency Score: 43, Existence Score: 11, WebLinks Score: 77, AdsPriority Score: 15]
28.www.nbcbayarea.com/tag/san- RankScore: 139 [Frequency Score: 22, Existence Score: 85, WebLinks Score: 20, AdsPriority Score: 12]
29.www.fairmont.com/san-jose/& RankScore: 134 [Frequency Score: 63, Existence Score: 35, WebLinks Score: 31, AdsPriority Score: 5]
30.www.sjquiltmuseum.org/&sa=U RankScore: 122 [Frequency Score: 19, Existence Score: 43, WebLinks Score: 37, AdsPriority Score: 23]
-----
Please select the following option:
-----
1.Heap Priority Queue for first 20 out of 30 website
you have to select 1 to do the options below
-----
2.Insert a new web url link
3.View the first ranked web url link and remove it.
4.Increase a webpages' PageRank score by adding money
5.View a website's info
```

I guide the user to select option “1”, since we need a heap priority queue to do the rest of the option. While user enter other number, I used a while loop to guide the user to select “1”

```
// Promote user to select options
System.out.println("-----");
System.out.println("Please select the following option:");
System.out.println("-----");
System.out.println("1.Heap Priority Queue for first 20 out of 30 website");
System.out.println("you have to select 1 to do the options below");
System.out.println("-----");
System.out.println("2.Insert a new web url link");
System.out.println("3.View the first ranked web url link and remove it.");
System.out.println("4.Increase a webpages' PageRank score by adding money");
System.out.println("5.View a website's info");
int selection = scanner.nextInt();

while(selection != 1 )
{
    System.out.println("Please enter 1! ");
    selection = scanner.nextInt();
}
```

Output:


```
-----  
Please select the following option:  
-----
```

```
1.Heap Priority Queue for first 20 out of 30 website  
you have to select 1 to do the options below  
-----
```

```
2.Insert a new web url link  
3.View the first ranked web url link and remove it.  
4.Increase a webpages' PageRank score by adding money  
5.View a website's info
```

```
2
```

```
Please enter 1!
```

After the user selects option 1, `heapPriorityQueue()` method will be called. The `promote` will call again after `heapPriorityQueue()` being called.

```
//heap priority queue if selected 1  
if(selection == 1) {  
    System.out.println("-----");  
    System.out.println("Here are the first 20 sorted urls: ");  
    heapPriorityQueue();  
    repeatPromote();  
}
```

`heapPriorityQueue()` method passes the first 20 url to `heapPriority queueAs` as well as store the first 20 in a new `ArrayList` name *heapWebites* and print out the result.

```
/**  
 * Heap Priority Queue with the first 20 websites  
 */  
public static void heapPriorityQueue()  
{  
    for (int i = 0; i < 20; i++) {  
        //insert sorted websites in to a heap.  
        heapPriority[i] = scoreArray[i];  
        heapWebsites.add(i, sortedWebsites.get(i));  
        System.out.println(i+1 + "." + heapWebsites.get(i).getUrls() + " RankScore: " + heapWebsites.get(i).getTotalScore());  
    }  
}
```

The method `repeatPromote()` is for repeat promotion to guide users to make selections. Option one won't be shown because a Heap Priority Queue had been created.

```

/**
 * Promote user to select options
 */
public static void repeatPromote(){
    System.out.println("-----");
    System.out.println("Please select the following option:");
    System.out.println("-----");
    System.out.println("2.Insert a new web url link"); ;
    System.out.println("3.View the first ranked web url link and remove it");
    System.out.println("4.Increase a webpages' PageRank score");
    System.out.println("5.View a website's info");
    System.out.println("**Press any other number to quit**");
}

```

Output:

```

1
-----
Here are the first 20 sorted urls:
1.en.wikipedia.org/wiki/Timel RankScore: 300
2.www.sanjose.org/&sa=U&ved=2 RankScore: 292
3.www.operasj.org/&sa=U&ved=2 RankScore: 272
4.www.sjica.org/&sa=U&ved=2ah RankScore: 257
5.sjsuspartans.com/&sa=U&ved= RankScore: 253
6.www.cdm.org/&sa=U&ved=2ahUK RankScore: 249
7.www.facebook.com/CityofSanJ RankScore: 247
8.sanjosetheaters.org/&sa=U&v RankScore: 244
9.en.wikipedia.org/wiki/Histo RankScore: 234
10.sjmusart.org/&sa=U&ved=2ahU RankScore: 227
11.www.sjsu.edu/&sa=U&ved=2ahU RankScore: 206
12.www.redfin.com/city/17420/C RankScore: 206
13.broadwaysanjose.com/&sa=U&v RankScore: 203
14.www.sjUSD.org/&sa=U&ved=2ah RankScore: 202
15.worldpopulationreview.com/u RankScore: 200
16.www.cnn.com/2020/10/22/us/s RankScore: 197
17.en.wikipedia.org/wiki/Santa RankScore: 196
18.collegefootballnews.com/202 RankScore: 195
19.www.sanjosehotel.com/&sa=U& RankScore: 185
20.www.japantownsanjose.org/&s RankScore: 175
-----
Please select the following option:
-----
2.Insert a new web url link
3.View the first ranked web url link and remove it
4.Increase a webpages' PageRank score
5.View a website's info
**Press any other number to quit**

```

While user selected option 2, it allows the user to insert a web url and its 4 score factors.

```

//Insert Website if selected 2
if(selection == 2) {
    System.out.println("-----");
    System.out.println("Please enter a url ");
    String url1 = scanner.next().toLowerCase();
    System.out.println("Please enter the frequency score for this url (Max:100)");
    int freqScore = scanner.nextInt();
    System.out.println("Please enter the existence score for this url (Max:100)");
    int existScore = scanner.nextInt();
    System.out.println("Please enter the web page links score for this url (Max:100)");
    int webScore = scanner.nextInt();
    System.out.println("Please enter the advertising score for this url (Max:100)");
    int adverScore = scanner.nextInt();
    insertWebsite(url1,freqScore,adverScore,existScore,webScore);
    repeatPromote();
}

```

After the user entered the url string and 4 scores. insertWebsite() will be called and it have the parameter for the 4 scores. In the beginning of insertWebsite() method, it will modify the scores to 100 if users entered more than 100. Then, a new RankPage object will be constructed with 4 score factors. A new Urls object will be constructed with the url string from users' input and the PageRank object was constructed previously. Add the Urls object to the *heapWebsites* ArrayList. Then call maxHeapInsert() method to insert the total score into Heap Priority Queue following the max-heap properties. After that syncLinkAndScore() method will be called.

```

/**
 * Insert a website to the list with an url and 4 scores.
 *
 * @param url website link.
 * @param keyword frequency score for this url.
 * @param ads advertising score for this url.
 * @param page existence score for this url.
 * @param web web page links score for this url.
 */
public static void insertWebsite(String url,int keyword , int ads, int page , int web)
{
    //Check every score
    if(keyword > 100) {
        keyword = 100;
    }
    if(ads > 100) {
        ads = 100;
    }
    if(page > 100) {
        page = 100;
    }
    if(web > 100) {
        web = 100;
    }
    PageRank a = new PageRank(keyword, ads, page, web);
    Urls b = new Urls(url,a); //creat a new websites object with rank score.
    heapWebsites.add(b);
    sorter.maxHeapInsert(heapPriority, a.getTotalScore());
    syncLinkAndScore();
}

```

syncLinkAndScore() method is for sync the heapWebsites list and the score in heapPriorityQueue by comparing the scores. Also, print out the current list.

```
/**
 * sync the heapWebsites list and heapPriority
 * Print out the updated heap websites list
 */
public static void syncLinkAndScore() {
    heapTempWebsites = new ArrayList<Urls>();
    for(int i = 0 ; i < 30; i++) {
        for(int j = 0; j < heapWebsites.size() ;j++ ) {
            //compare scores in array and ArrayList
            if(heapPriority[i] == heapWebsites.get(j).getTotalScore()) {
                heapTempWebsites.add(new Urls(heapWebsites.get(j).getUrls(),heapWebsites.get(j).getPageRank()));
                heapWebsites.remove(j);
            }
        }
    }

    heapWebsites = heapTempWebsites;
    System.out.println("Current List:");
    for(int i = 0 ; i < heapWebsites.size(); i++) {
        System.out.println(i+1 + "." + heapWebsites.get(i).getUrls() + " RankScore: " + heapWebsites.get(i).getTotalScore());
    }
}
```

In this case, I insert url “www.sjsu.edu”, assign a score 40,50,60,70 for each category, and the total score will be 220. The number of website in the list will increase to 11.

(See output next page)

Output:

```
**Press any other number to quit**
2
-----
Please enter a url
www.sjsu.edu
Please enter the frequency score for this url (Max:100)
40
Please enter the existence score for this url (Max:100)
50
Please enter the web page links score for this url (Max:100)
60
Please enter the advertising score for this url (Max:100)
70
Current List:
1.en.wikipedia.org/wiki/Timel RankScore: 300
2.www.sanjoose.org/&sa=U&ved=2 RankScore: 292
3.www.operasj.org/&sa=U&ved=2 RankScore: 272
4.www.sjica.org/&sa=U&ved=2ah RankScore: 257
5.sjsuspartans.com/&sa=U&ved= RankScore: 253
6.www.cdm.org/&sa=U&ved=2ahUK RankScore: 249
7.www.facebook.com/CityofSanJ RankScore: 247
8.sanjosetheaters.org/&sa=U&v RankScore: 244
9.en.wikipedia.org/wiki/Histo RankScore: 234
10.sjmusart.org/&sa=U&ved=2ahU RankScore: 227
11.www.sjsu.edu RankScore: 220
12.www.sjsu.edu/&sa=U&ved=2ahU RankScore: 206
13.broadwaysanjoose.com/&sa=U&v RankScore: 203
14.www.sjjud.org/&sa=U&ved=2ah RankScore: 202
15.worldpopulationreview.com/u RankScore: 200
16.www.cnn.com/2020/10/22/us/s RankScore: 197
17.en.wikipedia.org/wiki/Santa RankScore: 196
18.collegefootballnews.com/202 RankScore: 195
19.www.sanjoosehotel.com/&sa=U& RankScore: 185
20.www.japantownsanjoose.org/&s RankScore: 175
21.www.redfin.com/city/17420/C RankScore: 206
-----
Please select the following option:
-----
2.Insert a new web url link
3.View the first ranked web url link and remove it
4.Increase a webpages' PageRank score
5.View a website's info
**Press any other number to quit**
```

While the user selects the third option (3), it allows the user to view the first ranked web url link and remove it. The firstRankAndRemove() method will be called and repeatPromote() will be called as well.

```
//view and remove the first rank website if select 3
if(selection == 3) {
    firstRankAndRemove();
    repeatPromote();
}
```


In the beginning of `firstRankAndRemove()` method, I shows the user the first rank url using `get()` method to return the url link, and I implemented `heapMaximum()` method in the `HeapSort` class to return the the first score in the heap priority queue. Then, `heapExtractMax` will be called on the heap priority queue to remove the first element in the *heapPriority* array. The first `Urls` object in *heapWebsites* `ArrayList` will be removed as well. Then, sync the link and score using `syncLinkAndScore()` method.

```
/**
 * view the first rank url and remove it from the list.
 */
public static void firstRankAndRemove() {
    System.out.println("-----");
    System.out.println("Here is the first ranked website ");
    System.out.println("1. " + heapWebsites.get(0).getUrls() + " RankScore:" + sorter.heapMaximum(heapPriority));
    System.out.println("<This website have been removed from the list>");
    System.out.println("-----");
    sorter.heapExtractMax(heapPriority);
    heapWebsites.remove(0);
    syncLinkAndScore();
}
```

Output: (see next page)

```

5.view a website's info
**Press any other number to quit**
3
|-----
Here is the first ranked website
1. en.wikipedia.org/wiki/Time1 RankScore:300
<This website have been removed from the list>
|-----
Current List:
1.www.sanjose.org/&sa=U&ved=2 RankScore: 292
2.www.operasj.org/&sa=U&ved=2 RankScore: 272
3.sjsuspartans.com/&sa=U&ved= RankScore: 253
4.www.sjica.org/&sa=U&ved=2ah RankScore: 257
5.en.wikipedia.org/wiki/Histo RankScore: 234
6.www.cdm.org/&sa=U&ved=2ahUK RankScore: 249
7.www.facebook.com/CityofSanJ RankScore: 247
8.sanjosetheaters.org/&sa=U&v RankScore: 244
9.www.sjsu.edu/&sa=U&ved=2ahU RankScore: 206
10.www.redfin.com/city/17420/C RankScore: 206
11.sjmusart.org/&sa=U&ved=2ahU RankScore: 227
12.www.sjsu.edu RankScore: 220
13.broadwaysanjose.com/&sa=U&v RankScore: 203
14.www.sjUSD.org/&sa=U&ved=2ah RankScore: 202
15.worldpopulationreview.com/u RankScore: 200
16.www.cnn.com/2020/10/22/us/s RankScore: 197
17.en.wikipedia.org/wiki/Santa RankScore: 196
18.collegefootballnews.com/202 RankScore: 195
19.www.sanjosehotel.com/&sa=U& RankScore: 185
20.www.japantownsanjose.org/&s RankScore: 175
|-----
Please select the following option:
|-----
2.Insert a new web url link
3.View the first ranked web url link and remove it
4.Increase a webpages' PageRank score
5.View a website's info
**Press any other number to quit**

```

While users select option 5, it allows the user to see the 4 other factor scores. It promotes users to enter the index of the url and call the `HeapScoreInfo()` method and `repeatPromote()` at the end.

```

//get website info if select 5
if(selection == 5 ) {
    System.out.println("-----");
    System.out.println("Please enter the index of the url you want get info ");
    int url = scanner.nextInt();
    System.out.println("-----");
    HeapScoreInfo(url);
    System.out.println("-----");
    repeatPromote();
}

```

The `HeapScoreInfo` method decreases the index of users' input because the `ArrayList` and `Array` start with index 0. Then, print out the scores of a website using the `get` method.

```

/**
 * Get other 4 scores info of a heap website
 * @param i the website being get info
 */
public static void HeapScoreInfo(int i) {
    i--;
    System.out.println("Website: " + heapWebsites.get(i).getUrls() + " Total Score: " + heapWebsites.get(i).getTotalScore());
    System.out.println("Frequency Score: " + heapWebsites.get(i).getFrequency());
    System.out.println("Existence Score: " + heapWebsites.get(i).getTimeExisted());
    System.out.println("WebLinks Score: " + heapWebsites.get(i).getNumberWebLinks());
    System.out.println("AdsPriority Score: " + heapWebsites.get(i).getAdsPriority());
}

```

In this case, I am going to see the scores for www.sjsu.edu , which I inserted previously. The score matches what I input previously.

Output:

```

/.www.facebook.com/CityofSanJ RankScore: 24/
8.sanjosetheaters.org/&sa=U&v RankScore: 244
9.www.sjsu.edu/&sa=U&ved=2ahU RankScore: 206
10.www.redfin.com/city/17420/C RankScore: 206
11.sjmusart.org/&sa=U&ved=2ahU RankScore: 227
12.www.sjsu.edu RankScore: 220
13.broadwaysanjose.com/&sa=U&v RankScore: 203
14.www.sjUSD.org/&sa=U&ved=2ah RankScore: 202
15.worldpopulationreview.com/u RankScore: 200
16.www.cnn.com/2020/10/22/us/s RankScore: 197
17.en.wikipedia.org/wiki/Santa RankScore: 196
18.collegefootballnews.com/202 RankScore: 195
19.www.sanjosehotel.com/&sa=U& RankScore: 185
20.www.japantownsanjose.org/&s RankScore: 175

```

Please select the following option:

2.Insert a new web url link
3.View the first ranked web url link and remove it
4.Increase a webpages' PageRank score
5.View a website's info
Press any other number to quit

5

Please enter the index of the url you want get info

12

Website: www.sjsu.edu Total Score: 220
Frequency Score: 40
Existence Score: 50
WebLinks Score: 60
AdsPriority Score: 70

While the user selects option 4, it allows the user to increase rank score for a website. Begin with promoting users to select the index of the website and the amount of money they want to pay for an increased rank score. Then, the `increaseRankScore()` method will be called and `repeatPromote` will be called.

```
//increase a websites' rank score if select 4
if(selection == 4) {
    System.out.println("-----");
    System.out.println("Please enter the index of the website you modify ");
    int url = scanner.nextInt();
    System.out.println("Please enter a the amount of money you want to pay ");
    int money = scanner.nextInt();
    increaseRankScore(url,money);
    repeatPromote();
}
```

The `increaseRankScore` have parameters taking the index and money from users' input. Then, use the `get()` method to locate the object and call `higherPriority()` method to increase the score for the advertising factor. Next, initialize a int variable to get() total score of the `Urls` object. The call `heapIncreaseKey()` method from `HeapSort` class to increase the value of the corresponding index in `heapPriority` queue. Then, call `syncLinkAndScore()` methods to link the website and score.

```
/**
 * Increase a websites' rank score by adding money.
 * Maximum score for Adspriority is 100.
 */
public static void increaseRankScore(int index, int money){
    heapWebsites.get(index-1).getPageRank().higherPriority(money);
    int a = heapWebsites.get(index-1).getTotalScore();
    sorter.heapIncreaseKey(heapPriority, index-1, a);
    syncLinkAndScore();
}
```

`higherPriority()` method in `PageRank` class allows users to increase the advertising score. It will begin to add up the advertising score and the amount of money input by the user. Checking the amount of money users have input. Set score to 100 if more than 100. The total score will be the sum of the updated advertising score and other 3 scores.


```

/**
 * the amount of money that Webpage owners paid to increase webpage's exposure for advertisement purpose.
 * the highest priority score is 100.
 *
 * @param paid the amount of money being paid.
 */
public void higherPriority(int money)
{
    adsPriority = adsPriority + money;
    if (adsPriority > 100) {
        adsPriority = 100;
    }
    totalScore = keywordFrequency + pageExisted + webLinks + adsPriority;
}

```

In this case, I increase the www.sjsu.edu websites' advertising score by 40.

Output:

```

-----
Please enter the index of the url you want get info
12
-----
Website: www.sjsu.edu Total Score: 220
Frequency Score: 40
Existence Score: 50
WebLinks Score: 60
AdsPriority Score: 70
-----
Please select the following option:
-----
2.Insert a new web url link
3.View the first ranked web url link and remove it
4.Increase a webpages' PageRank score
5.View a website's info
**Press any other number to quit**
4
-----
Please enter the index of the website you modify
12
Please enter a the amount of money you want to pay
40
Current List:
1.www.sanjose.org/&sa=U&ved=2 RankScore: 292
2.www.operasj.org/&sa=U&ved=2 RankScore: 272
3.sjsuspartans.com/&sa=U&ved= RankScore: 253
4.www.sjica.org/&sa=U&ved=2ah RankScore: 257
5.en.wikipedia.org/wiki/Histo RankScore: 234
6.www.sjsu.edu RankScore: 250
7.www.facebook.com/CityofSanJ RankScore: 247
8.sanjosetheaters.org/&sa=U&v RankScore: 244
9.www.sjsu.edu/&sa=U&ved=2ahU RankScore: 206
10.sjmusart.org/&sa=U&ved=2ahU RankScore: 227
11.www.cdm.org/&sa=U&ved=2ahUK RankScore: 249
12.broadwaysanjose.com/&sa=U&v RankScore: 203
13.www.sjjud.org/&sa=U&ved=2ah RankScore: 202
14.worldpopulationreview.com/u RankScore: 200
15.www.cnn.com/2020/10/22/us/s RankScore: 197
16.en.wikipedia.org/wiki/Santa RankScore: 196
17.collegefootballnews.com/202 RankScore: 195
18.www.sanjosehotel.com/&sa=U& RankScore: 185
19.www.japantownsanjose.org/&s RankScore: 175
20.www.redfin.com/city/17420/C RankScore: 206
<

```


To see if it function, call check info by entering #5 and select the index. The score had increased by 30 because it is the maximum score. www.sjsu.edu had ranked up to 6 places by heap properties.

Output:

```
5.en.wikipedia.org/wiki/Histo RankScore: 234
6.www.sjsu.edu RankScore: 250
7.www.facebook.com/CityofSanJ RankScore: 247
8.sanjosetheaters.org/&sa=U&v RankScore: 244
9.www.sjsu.edu/&sa=U&ved=2ahU RankScore: 206
10.sjmusart.org/&sa=U&ved=2ahU RankScore: 227
11.www.cdm.org/&sa=U&ved=2ahUK RankScore: 249
12.broadwaysanjose.com/&sa=U&v RankScore: 203
13.www.sjusd.org/&sa=U&ved=2ah RankScore: 202
14.worldpopulationreview.com/u RankScore: 200
15.www.cnn.com/2020/10/22/us/s RankScore: 197
16.en.wikipedia.org/wiki/Santa RankScore: 196
17.collegefootballnews.com/202 RankScore: 195
18.www.sanjosehotel.com/&sa=U& RankScore: 185
19.www.japantownsanjose.org/&s RankScore: 175
20.www.redfin.com/city/17420/C RankScore: 206
-----
Please select the following option:
-----
2.Insert a new web url link
3.View the first ranked web url link and remove it
4.Increase a webpages' PageRank score
5.View a website's info
**Press any other number to quit**
5
-----
Please enter the index of the url you want get info
6
|-----
Website: www.sjsu.edu Total Score: 250
Frequency Score: 40
Existence Score: 50
WebLinks Score: 60
AdsPriority Score: 100
-----
Please select the following option:
-----
2.Insert a new web url link
3.View the first ranked web url link and remove it
4.Increase a webpages' PageRank score
5.View a website's info
**Press any other number to quit**
```

Quit if selected any other number. Use an if block to check the condition.

```
//quit if select any other number
if( selection != 1 && selection != 2 && selection != 3
    && selection != 4 && selection != 5) {
    System.out.println("Thanks for using! Bye! ");
    scanner.close();
}
```

I entered 9 to quit in this case.

Output:

```
EXISTENCE SCORE: 50
WebLinks Score: 60
AdsPriority Score: 100
-----
Please select the following option:
-----
2.Insert a new web url link
3.View the first ranked web url link and remove it
4.Increase a webpages' PageRank score
5.View a website's info
**Press any other number to quit**
9
Thanks for using! Bye!
```

Procedure to unzip files, install application and run code.

Window OS:

1. Unzip PA1- Wu.zip on prefer directory.
2. Run cmd (Command Prompt) on the computer.
3. Use command cd in cmd to change the directory to where PA-1.jar located.
4. Use command java -jar PA-1.jar in cmd to run the program.

Problems encountered during the implementation

One of the problems that I encountered during the implementation is index out of bounds while implementing the HeapSort class. Since the textbook started from index 1, so every index should be off by 1 while I implement HeapSort algorithms. I used the Debug tool in the IDE to step in every step and go through the lecture slides from Professor Wu to figure out what should do with the “off by 1 index” to solve the problem.

Another problem that I faced during implementation is linking the score with websites. I asked Professor Wu and he gave me some insight. He told me to create an URL class containing an URL string, 4 scores, and PageRank, etc. Then use an ArrayList to store URL objects. So, I created a class named Urls. Its constructor takes a url String and PageRank object. I store Urls objects into an ArrayList so the score and websites are linked. Since I stored scores in an Array and website in an ArrayList, the website won't sync with the score array after I sorted the score array with heap sort. Then, I figured out that the scores are the same. I can sort the Urls ArrayList by comparing the scores. So, I used nested loops to scan through every single elements' score in the ArrayList and compare it with the score in the sorted array. While the scores are the same, I add the Urls Object to a new Urls ArrayList.

There are also some small problems that I encountered during the implementation. One of the examples is that while I promote users to enter "s" to sort the list, users might enter other letters and it will return errors. I used a while loop to ensure users enter "s". I did the same thing on promoting users to make a selection. Another example will be the first time I implemented insertWebsite() method. It allows the user to insert a website into the heap priority queue, the method only has one parameter. This parameter takes the total score of the inserted website. However, the other four score factors will be randomized when I call HeapScoreInfo() methods. So, I created another PageRank constructor that allows users to enter the score for other 4 score factors and modify insertWebsite() method to have 4 parameters.

Lessons Learned

This program assignment is the first program that I had fully developed by myself, excluding the WebCrawler class since it was provided by Professor Wu. I learned the concept of heap sort, heap priority queue as well as the max heap properties. Heap sort is a sorting algorithm using max-heap. A max-heap is a binary tree that has a root and parents node, every parent node has a left and right child node. By following the max heap properties, the value of a parent is always bigger than their two children. The maximum value in a heap is the root of the heap. Heap priority queue allows inserting, removing and modifying nodes in a heap following the max heap properties. I learned how to implement the heap sort algorithm by following the pseudo codes from the book Introduction To Algorithms Third Edition. I learned that I can store objects into an ArrayList in order to link strings and integers. Throughout the assignment, I have a better understanding on what to do as a software engineer. I realized that developing a program is a long process. I spent many times on this programming assignment and I am getting better at it. I am looking forward to improving myself and becoming a better software engineer.