

Google Search Engine Simulation

Aniqua Azad

1. Design and Implementation

Commented [1]: Briefly explain/illustrate how you design and implement your Google Search Engine Simulator such as diagram, graph, data structure, (Array, ArrayList, Linked List, Vector, etc.).

There are two classes which are part of the Google Search Engine Simulator. The first class, `GoogleSearchEngine`, covers the first part of the assignment(i.e. Asking users for a keyword, etc). It uses `ArrayLists` with the object as the class `URLS`, which holds the url and score and has get-set methods, as well as `ArrayLists` of `String` objects. When the user enters a keyword, the urls that are found are stored in an `ArrayList` of `String`, and then the first 30 links are stored in another `ArrayList` of `String`. When the PageRank score is calculated, each url and its score is stored in a `HashMap` with the url as key and PageRank as value. Finally, it prints the PageRank scores in descending order with their respective urls.

The second class, `GoogleSearch2`, focuses on Max-Heap priority queue. It uses `ArrayLists` of `URLS` only. When this part of the program runs, it takes the top 20 links and stores them in an `ArrayList` of `URLS`, which is then passed onto the `start()` function. Everytime a function ends, it passes the `ArrayList` to the `start()` function as the user can add to it, see the top ranked url, etc as many times as they want, so the `ArrayList` is constantly changing.

There are user prompts throughout the program which allows them to continue or terminate the program whenever they want.

2. A list of classes/subroutines/function calls

- a. **GoogleSearchEngine:** Starts the Google Search Engine Simulator and covers the first half of programming requirements. It uses Heapsort to sort the PageRank scores.
 - i. **userInput():** Prompts user to enter a keyword to start the search
 - ii. **displayUrls():** displays the first 30 urls related to the keyword the user entered. Calls on the `WebCrawler` class's `search()` and `getUrls()` functions to find and retrieve the urls.
 - iii. **generateFactors():** Explains the 4 PageRank factors of a website and then lists each url with its 4 PageRank factor scores(which are generated randomly with the following function). It then adds up all the scores and puts each url with its score into a `HashMap` with urls as keys and scores as values.

Commented [2]: Explain purpose for each

- iv. **generateRandNum():**Generates random numbers between 1-100(inclusive) for each of the 4 PageRank factors. It was created for efficiency.
 - v. **pageRankOrder():** Prints out the sorting result in sequence order based on PageRank score
 - 1. 1) It takes each url and its associated score from the Map and adds it to the ArrayList<URLS>
 - 2. Then, it uses the Heapsort algorithm in the HeapSortCode class to sort the PageRank scores in ascending order
 - 3. Finally, it prints out each url with its score in descending order (greatest score -> smallest score)
 - vi. **main():** calls functions in classes and also asks user if they would like to continue to the next portion of the simulator or terminate the simulator.
- b. **GoogleSearch2:** Covers the second half of the program's requirements. Allows users to manipulate the list of links by adding a new link and score, changing the current PageRank score of a link, or viewing the first ranked page.
- i. **takeFirst20():** Takes the top 20 url links from the original 30 links in the previous class. Then, it implements the Heap priority queue to store the first 20 links by calling maxHeapInsert().
 - ii. **insertNewLink():** Allows user to insert a new web url and a corresponding PageRank score. They are then added to the ArrayList and maxHeapInsert() is implemented to make sure that the elements follow Heap property.
 - iii. **viewFirstRankedPage():** Allows user to view the first ranked page with url and score. Calls heapMaxExtract() to get information.
 - iv. **increasePageRanking():** Allows user to increase page ranking of any link in the ArrayList. Prompts them to enter the number of the link and enter a new score. Calls heapIncrease() key to make sure that the new number is in the correct place and follows Heap property. The urls and scores are printed out again for users to see the changes.

- v. **start():** Lists options for the user and prompts them to enter a number based on the option. They can also choose to terminate the program by pressing 0 when prompted.

c. HeapAndPQCode:

- i. **maxHeapify():** maxHeapify makes sure that part of the ArrayList (the element at node i and its subtree) follows the heap property, where the value of the parent node is greater than or equal to the value of its child node.
- ii. **buildMaxHeap():** Builds a max heap where all parent nodes are greater than or equal to the children, and the root node is the greatest value. Calls maxHeapify() to make sure that the entire ArrayList follows the heap property and double checks that all nodes satisfy the property.
- iii. **heapsort():** Takes an unsorted ArrayList with urls and scores and modifies it to be sorted in ascending order based on PageRank scores. It calls on buildMaxHeap() to make sure that all nodes follow the Heap property.
- iv. **maxHeapInsert():** Modifies ArrayList to include the value of key. Calls heapIncreaseKey() to place the new value in its correct position in the heap
- v. **heapIncreaseKey():** As heapIncreaseKey goes through its path, it compares key with the element of its parent, exchanging their keys and continuing if key' value is larger. It stops once the key is smaller than its parents' value and when the max-heap property holds.
- vi. **heapMaximum():** Returns the maximum URLS object of the heap
- vii. **heapExtractMax():** Allows user to view the first ranked web url and score stored in max-heap. Then, the heap removes the element and bumps the next element to the top.

d. URLS: This class stores the url and its associated score. It also contains getter-setter methods to access this information.

- i. **getURL():** Returns the String url
- ii. **getScore():** Returns the PageRank score
- iii. **setScore():** Sets the current PageRank score to a new score

3. **Self-testing screen shots**

Please find the screenshots at the end of this PDF.

Commented [3]: Provide enough screen shots for each function including inputs and outputs for each of function listed in "Functional Requirements" section. This is to verify your codes and to check to see if your codes match the functional requirements and run correctly by yourself

4. The procedure (step by step) of how to unzip your files, install your application, and run/test your codes.

- a. Extract files by unzipping folder
- b. Double-click folder titled **PA1-Section 7-Azad**
- c. To access *.java files:
 - i. Unzip the folder titled "src.zip"
 - ii. double-click folder titled "src", then double-click the folder titled "gses"
 - iii. Copy/paste the code into an IDE
- d. To access formal report, double-click PDF titled "AzadPA1_Formal Report.pdf"
- e. In order to run code, open command prompt/terminal.
 - i. Change directory to where AzadPA1.jar is stored
 - ii. type: java -jar AzadPA1.jar
 1. You will be prompted to enter a keyword; if your keyword has spaces(i.e. San Jose), please enter it without spaces(i.e. SanJose)
 2. You will then be prompted to continue or terminate the program
 - a. Every time you are prompted, you have the option of terminating.
 3. Keep on continuing and choosing various options until you would like to stop the program.

Commented [4]: (You will receive 0 point if no self-testing screenshots provided. Points will be taken off if not enough self-testing screen shots provided.)

5. **Problems encountered during the implementation**

During the implementation of this programming assignment, I ran into numerous problems.

- a) The first problem I ran into was the difference between heap-size and array length. I thought they were the same at first, so wherever it said to use heap-size in the necessary methods, I would just use array length. However, when I ran the program with a sample array, I kept on getting errors. First, I emailed the professor to find

Commented [5]: Describe all problems found/encountered during your implementation and how you resolved them. (-15 points if not provided or too simple.)

out what the difference was between the two. After getting his reply, I realized that heap-size depended on array length to some extent. So, to fix my problem, I made heap-size a global variable, initialized to array's length; all my methods finally worked after that.

- b) The second problem I ran into was how I would associate each url with its PageRank score. I was going through many data structures when I remembered Maps. As a result, I made a HashMap of the urls as keys and PageRank scores as values. At first in the Heap class, I made the ArrayLists of type class GoogleSearchEngine, which made it very difficult to get keep track of the url and scores, and there were methods all over the place. So, I also made a URLS class which has get-set() methods for the url and its score, making it easier to access them in other methods and classes, also making my code much more readable.
- c) Another problem I ran into during implementation was how I should write my classes. At first, I had one giant class which covered both parts of the assignment. However, it was difficult to keep track of all the variables and ArrayLists I had, so I made a separate class for the second half of the assignment, following object-oriented programming design.
- d) Another problem I ran into and was having the most difficulty with was implementing the maxHeapInsert() and heapIncreaseKey(). The problem was that I kept on getting either NullPointerException or IndexOutOfBoundsException. So, I first tried debugging my classes, and I thought that the problem was most likely with heapSize. I played around with the heapSize variable, subtracting 1, adding 1, etc. Subtracting 1 worked, but I kept getting Integer.MIN_VALUE as the score for all my urls. I also emailed the professor asking for help. He gave me code for reference, but when that didn't work, I debugged a little bit more by stepping into methods and inspecting variables. I realized that in heapIncreaseKey, the parameter key was of type URLS, which was the same as the parameter ArrayList<URLS> arr. By doing arr.set(i,key), I was setting Integer.MIN_VALUE to my key, thus changing all my scores. So, I changed the parameter of key to int, and changed all my calls to maxHeapInsert. Finally, all my code was running, and I was getting the correct outputs!

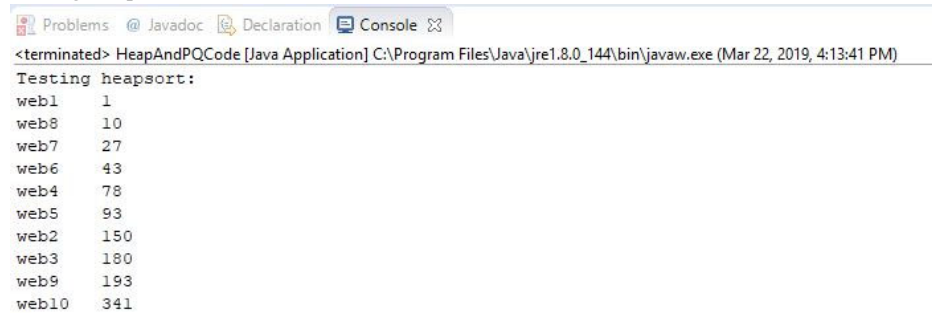
6. **Lessons Learned**

- a. The first lesson I learned was Heapsort works. At first I was confused as to why the array was in ascending order. I thought that it was supposed to be in descending order like the trees shown in class. After watching videos and from lecture in class, I realized that Heapsort puts the values in ascending order.
- b. Another lesson I learned was the difference between heap-size and array length. I thought they were the same thing, just with interchangeable names. However, after testing the Heapsort methods, I realized that they were actually different to an extent. I learned that when you manipulate a Heap tree, you use size and change it accordingly. Also, when you add a node to Heap, the size changes.
- c. I also learned that `maxHeapInsert()` maintains the heap property by making sure that the parent node is greater than or equal to the child node. When I was testing the individual methods, I saw that my output wasn't in order from least to greatest or vice versa, and I was worried. However, after class on Thursday and reading more about it, I realized that the output was printing how it would look like on an array, with the first element being the root node, and the following nodes being its children.
- d. I really enjoyed doing this project; however, I do believe my time complexity for this assignment is bad; I'm not sure what it is, but with the amount of for loops I have, it's probably a really large time complexity. When I have time, I would like to improve my code making it more efficient and trying to make worst-case run time smaller. This will not only help me become a better software engineer in the future, but it will also help me when I write future programs.

Commented [6]: Describe the concepts and skills you have learned from this programming assignment and any inspiration or motivation you got from this programming assignment. (-10 pints if not provided or too simple.)

Screenshots:

Testing Heapsort



The screenshot shows an IDE console window with the following content:

```
<terminated> HeapAndPQCode [Java Application] C:\Program Files\Java\jre1.8.0_144\bin\javaw.exe (Mar 22, 2019, 4:13:41 PM)
Testing heapsort:
web1      1
web8      10
web7      27
web6      43
web4      78
web5      93
web2     150
web3     180
web9     193
web10    341
```

Testing Max-Heap Priority Queue:

Problems Javadoc Declaration Console

<terminated> HeapAndPQCode [Java Application] C:\Program Files\Java\jre1.8.0_144\bin\javaw.exe (Mar 22, 2019, 4:11:13 PM)

Testing Max-Heap Priority:

web12	839
web10	341
web9	193
web4	78
web2	150
web3	180
web7	27
web8	10
web1	1
web5	93
web6	43
web11	156

Testing heapExtractMax():
Heap maximum1 is: web12 839
Testing heapMaximum():
Heap maximum2 is: web10 341

Testing the Google Search Engine Simulator:

Problems @ Javadoc Declaration Console

<terminated> GoogleSearchEngine [Java Application] C:\Program Files\Java\jre1.8.0_144\bin\javaw.exe (Mar 22, 2019, 5:59:26 PM)

Please type in a keyword with no spaces (i.e. San Jose = SanJose) and then press 'Enter': **sanjose**

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

Found (339) links

Searching for the word sanjose...

****Success**** Word sanjose found at <https://google.com/search?q=sanjose&num=80>

****Done**** Visited 91 web page(s)

Here are the first 30 URL links:

1. www.sanjose.org/photo-video
2. www.sjbarracuda.com/&sa=U&ved=
3. www.sanjoseca.gov/&sa=U&ved=
4. weather.com/weather/tenday/realstate.usnews.com/place
5. www.sanjose.org/&sa=U&ved=0
6. www.sanjose.org/blog&sa=U&v
7. sjsuspartans.com/&sa=U&ved=
8. www.sjmunl.com/&sa=U&ved=0a
9. www.sjws.org/&sa=U&ved=0ahUK
10. www.sanjoseca.gov/index.aspx
11. www.minetaairport.com/&sa=U&ved=
12. www.apple.com/newsroom/2019
13. www.operasj.org/&sa=U&ved=0
14. sanjosejazz.org/&sa=U&ved=0
15. www.sjpl.org/&sa=U&ved=0ahU
16. www.pticket.com/sanjose/&sa
17. www.lonelyplanet.com/costa
18. www.cmts1.org/&sa=U&ved=0ah
19. www.cmts1.org/&sa=U&ved=0ah

Problems @ Javadoc Declaration Console

<terminated> GoogleSearchEngine [Java Application] C:\Program Files\Java\jre1.8.0_144\bin\javaw.exe (Mar 22, 2019, 5:59:26 PM)

19. www.cmts1.org/&sa=U&ved=0ah
20. en.wikipedia.org/wiki/Downt
21. www.sjica.org/&sa=U&ved=0ah
22. www.redfin.com/city/17420/C
23. www.fairmont.com/san-jose/s
24. en.wikipedia.org/wiki/Histo
25. www.zillow.com/san-jose-ca/
26. www.zillow.com/homedetails/
27. www.cdm.org/&sa=U&ved=0ahUK
28. www.sjqultmuseum.org/&sa=U
29. taiko.org/&sa=U&ved=0ahUKew
30. www.hyatt.com/en-US/hotel/c

A Web page's PageRank depends on the FREQUENCY of keyword, AGE of the webpage, LINKS to the webpage, and AMOUNT paid for ads
Here are the following scores for each of the websites, respectively:

- | | | | | | |
|-----|--|----|----|----|----|
| 1. | www.sanjose.org/photo-video : | 26 | 57 | 39 | 90 |
| 2. | www.sjbarracuda.com/&sa=U&ved= : | 27 | 1 | 47 | 27 |
| 3. | www.sanjoseca.gov/&sa=U&ved= : | 73 | 30 | 22 | 45 |
| 4. | weather.com/weather/tenday/realstate.usnews.com/place : | 22 | 53 | 6 | 30 |
| 5. | realstate.usnews.com/place : | 66 | 15 | 57 | 50 |
| 6. | www.sanjose.org/&sa=U&ved=0 : | 96 | 34 | 47 | 70 |
| 7. | www.sanjose.org/blog&sa=U&v : | 32 | 26 | 71 | 82 |
| 8. | sjsuspartans.com/&sa=U&ved= : | 17 | 23 | 35 | 93 |
| 9. | www.sjmunl.com/&sa=U&ved=0a : | 38 | 36 | 75 | 60 |
| 10. | www.sjws.org/&sa=U&ved=0ahUK : | 16 | 68 | 14 | 11 |
| 11. | www.sanjoseca.gov/index.aspx : | 73 | 92 | 74 | 69 |
| 12. | www.minetaairport.com/&sa=U&ved= : | 90 | 59 | 81 | 52 |
| 13. | www.apple.com/newsroom/2019 : | 8 | 67 | 88 | 85 |

```
Problems @ Javadoc Declaration Console
<terminated> GoogleSearchEngine [Java Application] C:\Program Files\Java\jre1.8.0_144\bin\javaw.exe (Mar 22, 2019, 5:59:26 PM)
13. www.apple.com/newsroom/2019: 8 97 89 95
14. www.operasj.org/&sa=U&ved=0: 75 33 88 20
15. sanjosejazz.org/&sa=U&ved=0: 39 63 39 36
16. www.sjpl.org/&sa=U&ved=0ahU: 12 3 51 61
17. www.pticket.com/sanjose/&sa: 41 97 70 97
18. www.lonelyplanet.com/costa-: 56 21 56 37
19. www.cmts.org/&sa=U&ved=0ah: 90 59 46 86
20. en.wikipedia.org/wiki/Downt: 78 86 84 65
21. www.sjica.org/&sa=U&ved=0ah: 24 30 24 72
22. www.redfin.com/city/17420/C: 66 86 11 56
23. www.fairmont.com/san-jose/&: 57 57 90 70
24. en.wikipedia.org/wiki/Histo: 28 26 47 73
25. www.zillow.com/san-jose-ca/: 65 73 24 59
26. www.zillow.com/homedetails/: 50 66 8 93
27. www.cdm.org/&sa=U&ved=0ahUK: 3 7 21 60
28. www.sjqultmuseum.org/&sa=U: 70 92 59 35
29. taiko.org/&sa=U&ved=0ahUKEw: 28 36 82 19
30. www.hyatt.com/en-US/hotel/c: 48 86 56 93

If you would like to see the links based on PageRank score, press 1. Else press 0:
1
1. en.wikipedia.org/wiki/Downt PageRank Score: 313
2. www.sanjoseca.gov/index.aspx PageRank Score: 308
3. www.pticket.com/sanjose/&sa PageRank Score: 305
4. www.apple.com/newsroom/2019 PageRank Score: 289
5. www.hyatt.com/en-US/hotel/c PageRank Score: 283
6. www.minetaairport.com/&sa=U& PageRank Score: 282
7. www.cmts.org/&sa=U&ved=0ah PageRank Score: 281
8. www.fairmont.com/san-jose/& PageRank Score: 274

Problems @ Javadoc Declaration Console
<terminated> GoogleSearchEngine [Java Application] C:\Program Files\Java\jre1.8.0_144\bin\javaw.exe (Mar 22, 2019, 5:59:26 PM)
9. www.sjqultmuseum.org/&sa=U PageRank Score: 256
10. www.sanjose.org/&sa=U&ved=0 PageRank Score: 247
11. www.zillow.com/san-jose-ca/ PageRank Score: 221
12. www.redfin.com/city/17420/C PageRank Score: 219
13. www.zillow.com/homedetails/ PageRank Score: 217
14. www.operasj.org/&sa=U&ved=0 PageRank Score: 216
15. www.sanjose.org/photo-video PageRank Score: 212
16. www.sanjose.org/blog&sa=U&v PageRank Score: 211
17. www.sjmun.com/&sa=U&ved=0a PageRank Score: 209
18. realestate.usnews.com/place PageRank Score: 188
19. sanjosejazz.org/&sa=U&ved=0 PageRank Score: 177
20. en.wikipedia.org/wiki/Histo PageRank Score: 174
21. www.sanjoseca.gov/&sa=U&ved= PageRank Score: 170
22. www.lonelyplanet.com/costa- PageRank Score: 170
23. sjsuspartans.com/&sa=U&ved= PageRank Score: 168
24. taiko.org/&sa=U&ved=0ahUKEw PageRank Score: 165
25. www.sjica.org/&sa=U&ved=0ah PageRank Score: 150
26. www.sjpl.org/&sa=U&ved=0ahU PageRank Score: 127
27. weather.com/weather/tenday/ PageRank Score: 111
28. www.sjws.org/&sa=U&ved=0ahUK PageRank Score: 109
29. www.sjbarracuda.com/&sa=U&ve PageRank Score: 102
30. www.cdm.org/&sa=U&ved=0ahUK PageRank Score: 91

If you would like to continue, press 1. Else, press 0:
1
What would you like to do? Press the number according to the options below. If you would like to exit, press 0.
1. View the first ranked page
2. Increase the PageRank of any page on the list
3. Insert a new link
0. Terminate the program
<
```

```
Problems @ Javadoc Declaration Console
<terminated> GoogleSearchEngine [Java Application] C:\Program Files\Java\jre1.8.0_144\bin\javaw.exe (Mar 22, 2019, 5:59:26 PM)

1
URL: en.wikipedia.org/wiki/Downt PageRank Score: 313

What would you like to do? Press the number according to the options below. If you would like to exit, press 0.
1. View the first ranked page
2. Increase the PageRank of any page on the list
3. Insert a new link
0. Terminate the program
1
URL: www.sanjoseca.gov/index.aspx PageRank Score: 308

What would you like to do? Press the number according to the options below. If you would like to exit, press 0.
1. View the first ranked page
2. Increase the PageRank of any page on the list
3. Insert a new link
0. Terminate the program
2
Please type in the number of the website you would like to increase the ranking of:
1. www.pticket.com/sanjose/ssa PageRank Score: 305
2. www.apple.com/newsroom/2019 PageRank Score: 289
3. www.hyatt.com/en-US/hotel/c PageRank Score: 283
4. www.cmtsj.org/ssa=U&ved=0ah PageRank Score: 281
5. www.sjqiltmuseum.org/ssa=U PageRank Score: 256
6. www.minetaairport.com/ssa=U& PageRank Score: 282
7. www.zillow.com/homedetails/ PageRank Score: 217
8. www.fairmont.com/san-jose/ PageRank Score: 274
9. www.sjmunl.com/ssa=U&ved=0a PageRank Score: 209
10. www.sanjose.org/ssa=U&ved=0 PageRank Score: 247
11. www.zillow.com/san-jose-ca/ PageRank Score: 221

Problems @ Javadoc Declaration Console
<terminated> GoogleSearchEngine [Java Application] C:\Program Files\Java\jre1.8.0_144\bin\javaw.exe (Mar 22, 2019, 5:59:26 PM)
11. www.zillow.com/san-jose-ca/ PageRank Score: 221
12. www.redfin.com/city/17420/C PageRank Score: 219
13. sanjosejazz.org/ssa=U&ved=0 PageRank Score: 177
14. www.operasj.org/ssa=U&ved=0 PageRank Score: 216
15. www.sanjose.org/photo-video PageRank Score: 212
16. www.sanjose.org/blog&sa=U&v PageRank Score: 211
17. en.wikipedia.org/wiki/Histo PageRank Score: 174
18. realestate.usnews.com/place PageRank Score: 188
19. sanjosejazz.org/ssa=U&ved=0 PageRank Score: 177
20. en.wikipedia.org/wiki/Histo PageRank Score: 174
10
Please enter a new, increased score: 470
Please see your website increase in rank:
1. www.sanjose.org/ssa=U&ved=0 PageRank score: 470
2. www.pticket.com/sanjose/ssa PageRank score: 305
3. www.apple.com/newsroom/2019 PageRank score: 289
4. www.cmtsj.org/ssa=U&ved=0ah PageRank score: 281
5. www.hyatt.com/en-US/hotel/c PageRank score: 283
6. www.minetaairport.com/ssa=U& PageRank score: 282
7. www.zillow.com/homedetails/ PageRank score: 217
8. www.fairmont.com/san-jose/ PageRank score: 274
9. www.sjmunl.com/ssa=U&ved=0a PageRank score: 209
10. www.sjqiltmuseum.org/ssa=U PageRank score: 256
11. www.zillow.com/san-jose-ca/ PageRank score: 221
12. www.redfin.com/city/17420/C PageRank score: 219
13. sanjosejazz.org/ssa=U&ved=0 PageRank score: 177
14. www.operasj.org/ssa=U&ved=0 PageRank score: 216
15. www.sanjose.org/photo-video PageRank score: 212
16. www.sanjose.org/blog&sa=U&v PageRank score: 211
<
```

```
Problems @ Javadoc Declaration Console
<terminated> GoogleSearchEngine [Java Application] C:\Program Files\Java\jre1.8.0_144\bin\javaw.exe (Mar 22, 2019, 5:59:26 PM)
17. en.wikipedia.org/wiki/Histo PageRank score: 174
18. realestate.usnews.com/place PageRank score: 188
19. sanjosejazz.org/&sa=U&ved=0 PageRank score: 177
20. en.wikipedia.org/wiki/Histo PageRank score: 174
What would you like to do? Press the number according to the options below. If you would like to exit, press 0.
1. View the first ranked page
2. Increase the PageRank of any page on the list
3. Insert a new link
0. Terminate the program
1
URL: www.sanjose.org/&sa=U&ved=0 PageRank Score: 470
What would you like to do? Press the number according to the options below. If you would like to exit, press 0.
1. View the first ranked page
2. Increase the PageRank of any page on the list
3. Insert a new link
0. Terminate the program
3
Please enter a new web url with www.: www.ilovesjsu.com
Please enter a PageRank score (integer) for the webpage: 100
1. www.pticket.com/sanjose/&sa PageRank score: 305
2. www.apple.com/newsroom/2019 PageRank score: 289
3. www.hyatt.com/en-US/hotel/c PageRank score: 283
4. www.cmts.org/&sa=U&ved=0ah PageRank score: 281
5. www.sjqultmuseum.org/&sa=U PageRank score: 256
6. www.minetaairport.com/&sa=U& PageRank score: 282
7. www.zillow.com/homedetails/ PageRank score: 217
8. www.fairmont.com/san-jose/& PageRank score: 274
9. www.sjmunl.com/&sa=U&ved=0a PageRank score: 209
10. realestate.usnews.com/place PageRank score: 188
11. www.zillow.com/san-jose-ca/ PageRank score: 221
12. www.redfin.com/city/17420/C PageRank score: 219
13. sanjosejazz.org/&sa=U&ved=0 PageRank score: 177
14. www.operasj.org/&sa=U&ved=0 PageRank score: 216
15. www.sanjose.org/photo-video PageRank score: 212
16. www.sanjose.org/blog&sa=U&v PageRank score: 211
17. en.wikipedia.org/wiki/Histo PageRank score: 174
18. realestate.usnews.com/place PageRank score: 188
19. sanjosejazz.org/&sa=U&ved=0 PageRank score: 177
20. en.wikipedia.org/wiki/Histo PageRank score: 174
21. www.ilovesjsu.com PageRank score: 100
What would you like to do? Press the number according to the options below. If you would like to exit, press 0.
1. View the first ranked page
2. Increase the PageRank of any page on the list
3. Insert a new link
0. Terminate the program
0
Good-bye!
```