

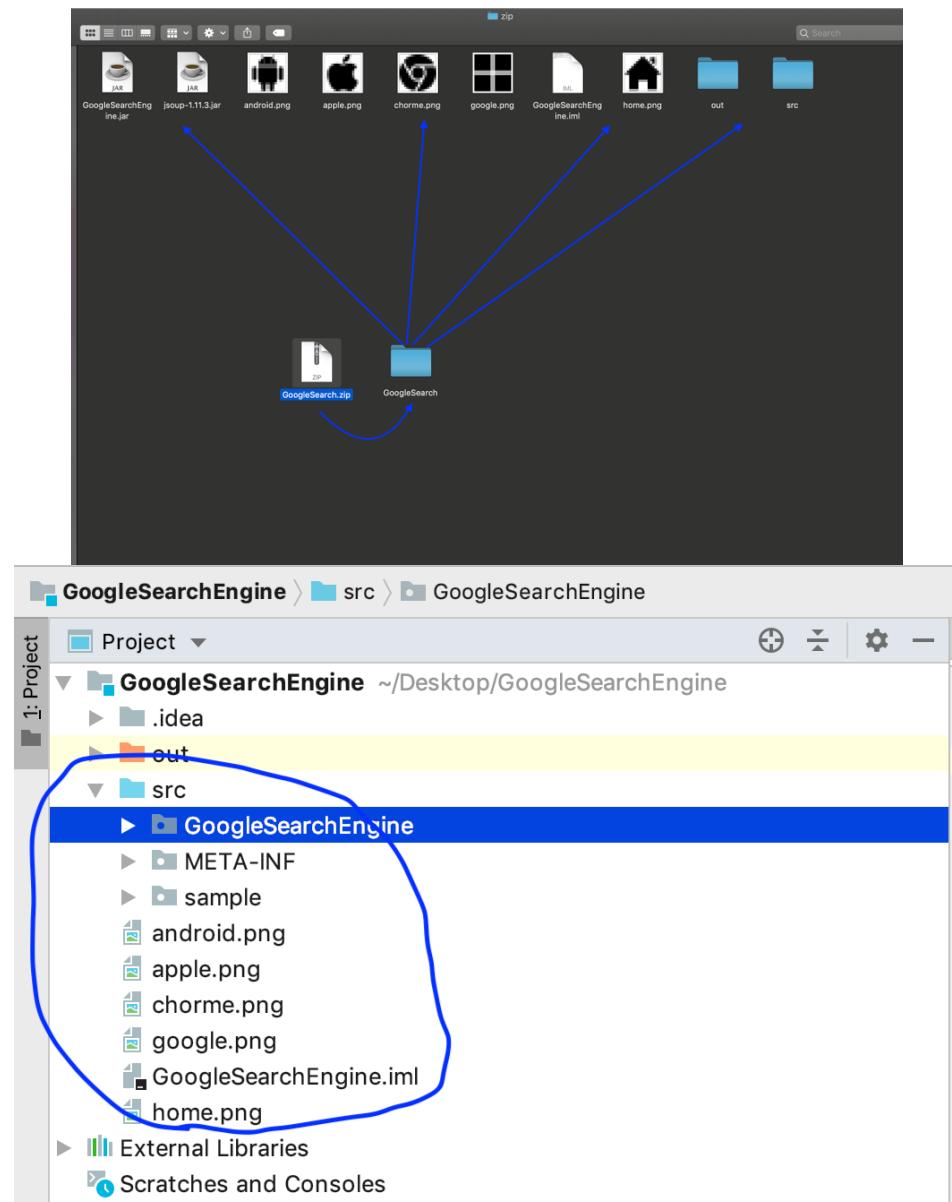
Programming Assignment 1: Google Search Engine Simulation

Yuxiao (Tom) Zheng
CS146-Section7

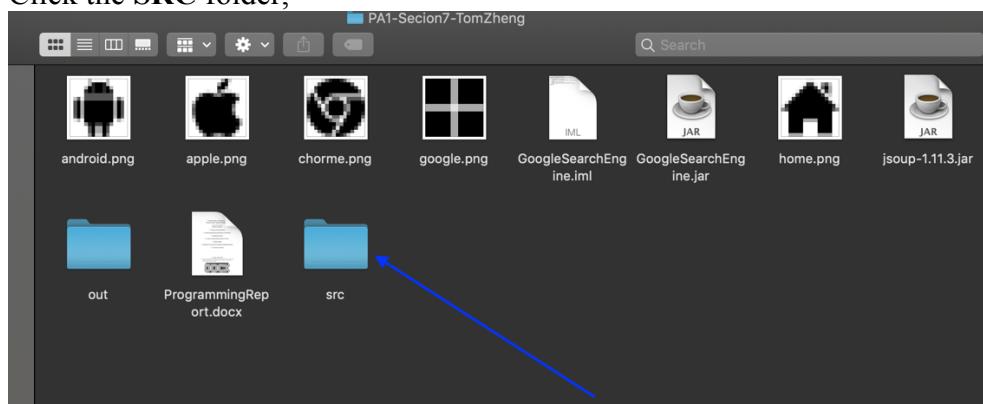
1. Unzip and Installation
2. Programming Design(Object-Oriented)
3. Implementation
4. classes/subroutines/function class:
5. Self-testing
6. Problems encountered during implementation
7. Leasson Learned
8. Updating (Debugging Report)

1. unzip and install

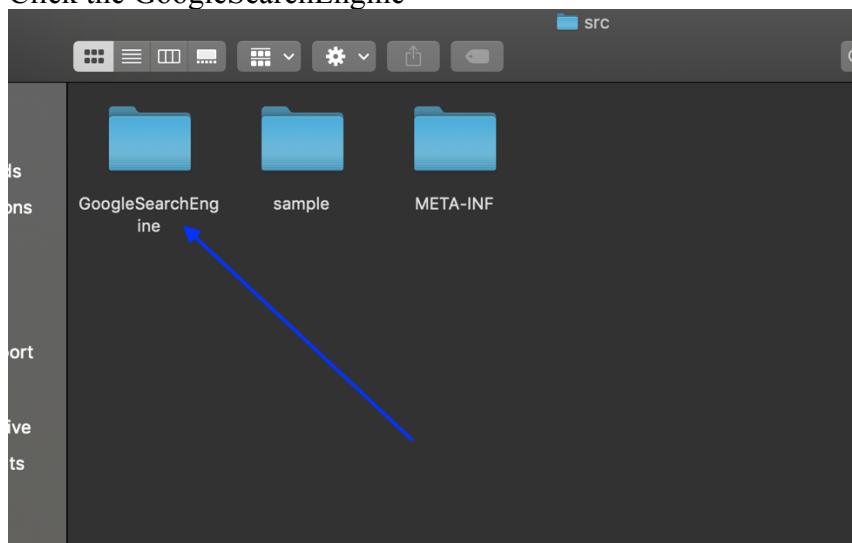
- Clicking the GoogleSearch.zip and unzip PA-1.zip into an empty folder, all .java files will be on the **src** folder. **Note:** the all .png imgs must be outside of **src folders**, then the IntelliJ or Eclipse can find the .png file.

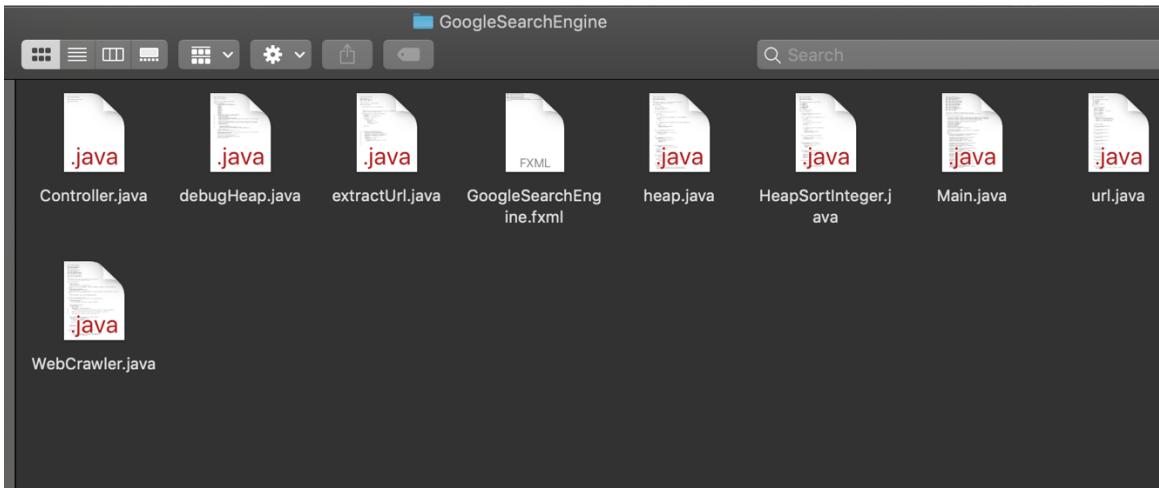


- Click the SRC folder,

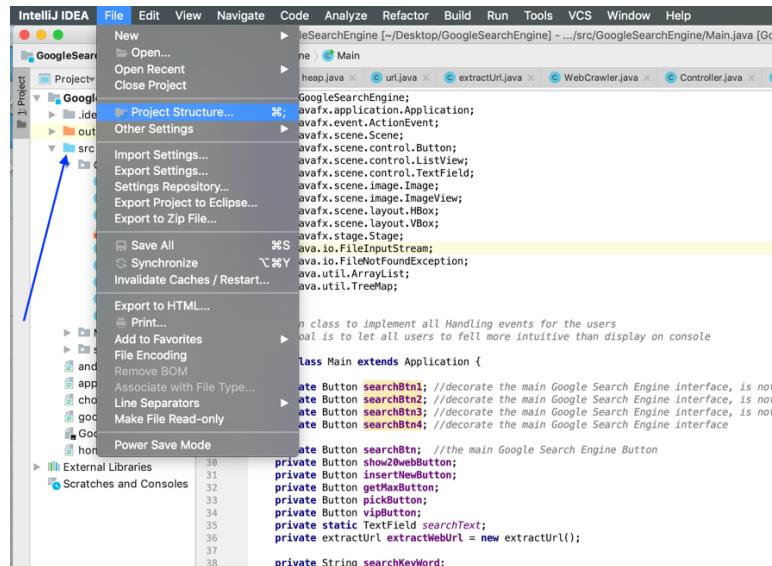


- Click the GoogleSearchEngine

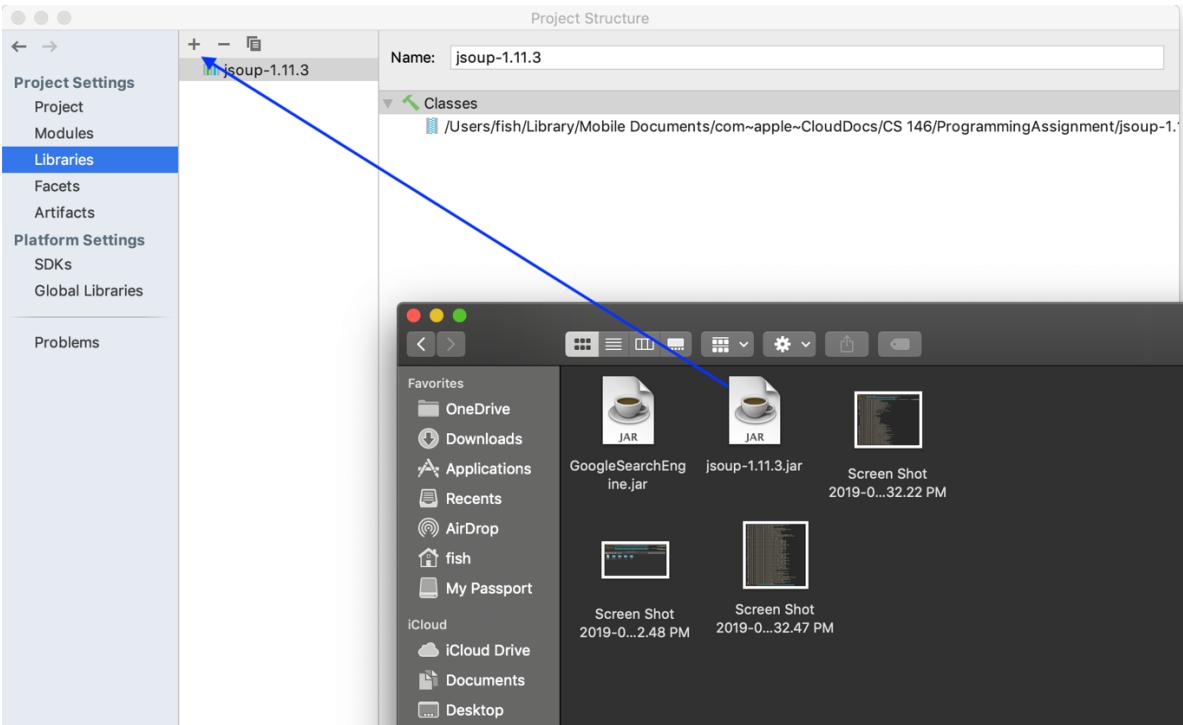




- Clicking the File → click Project Structure. First creating a Package GoogleSearchEngine put all source files including .java files into the Package, this Package must be into SRC folder.

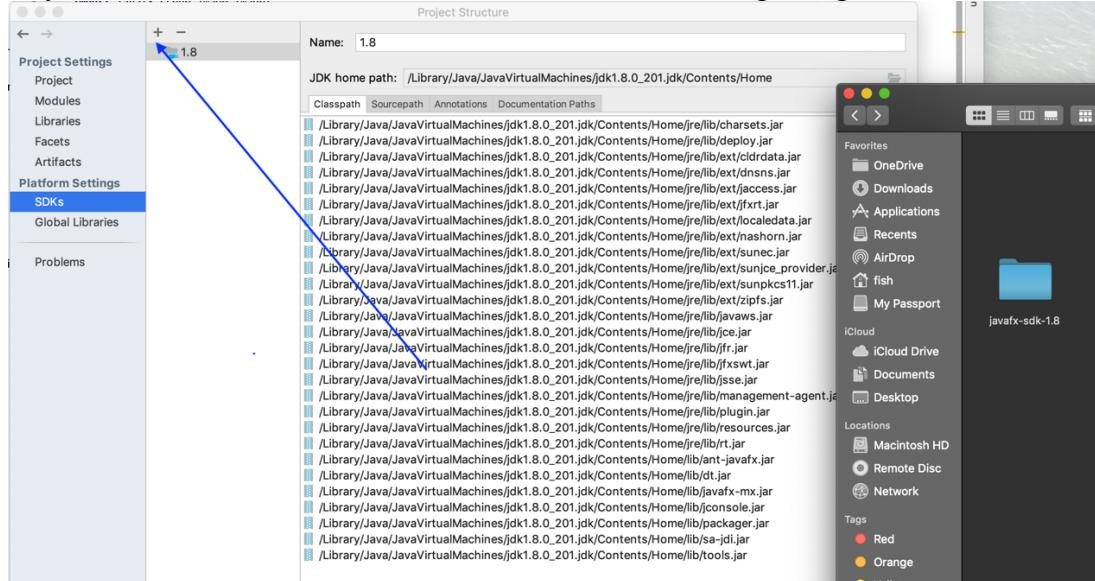


- After clicking Project Structure, click “+” it will import jsoup-1.11.3.jar. You will be using the all methods for extracting the web url by WebCrawler class.



- Note: Eclipse with JDK 8 under can export project into Runnable Jar, but not JDK 9. Therefore, I import JDK 8 for IntelliJ or Eclipse.

Firstly, click the “SDKs” button, and click “+” button. → importing JDK 8.



```

<?import javafx.geometry.Insets>
<?import javafx.scene.layout.GridPane?>
<?import javafx.scene.control.Button?>
<?import javafx.scene.control.Label?>
<GridPane fx:controller="GoogleSearch.Controller"
  xmlns:fx="http://javafx.com/fxml" alignment="center" hgap="10" vgap="10">
</GridPane>

```

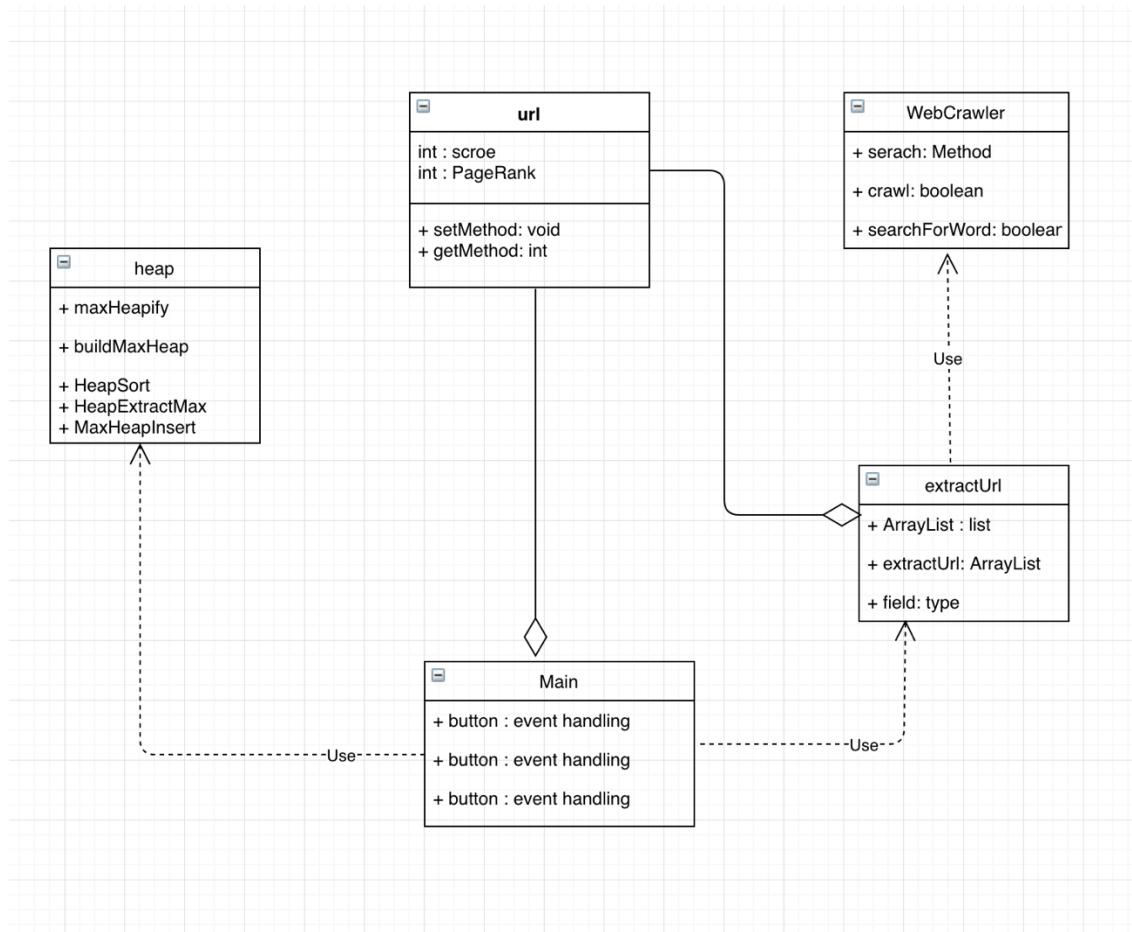
JavaFX required a .fxml file for IntelliJ

- To summary, all .java files → GoogleSearchEngine Package; all .png files is out of SRC folder; You must import JDK 8.

2. Programming Design

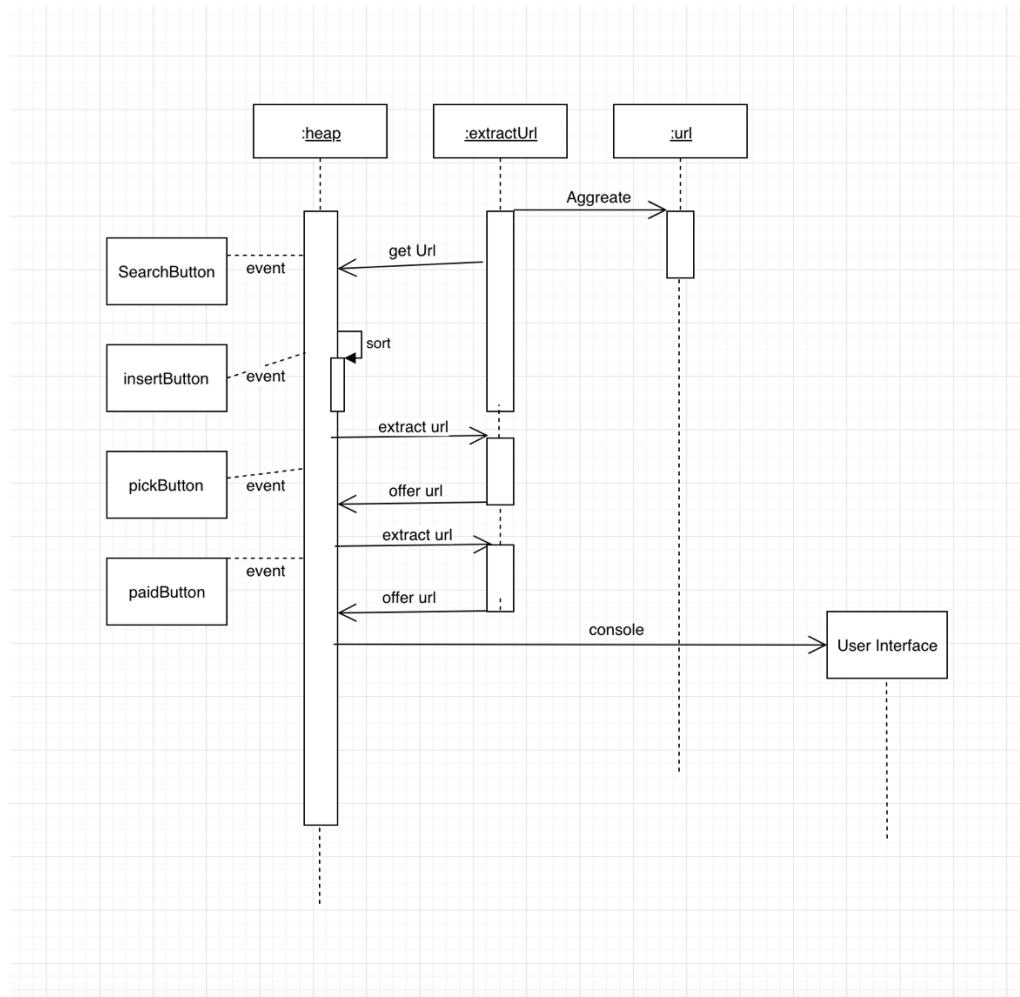
- Importing JavaFX library, it will make the **USER INTERFACE** more perceptual intuition, beautiful, and clean.
- Adding some .png icon images into this project, they will make the buttons more active and detailed.
- Importing the Jsoup library into this project in order to use the WebCrawler class; we need WebCrawler to implement the extractUrl class.
- My extractUrl class must aggregate the **URL** class, because all web urls that WebCrawler will capture, which will add into the ArrayList<url> for the Main class implementing different functioning for the users.
- The Main class depends on heap class and extractUrl, it was also aggregating with url class. The extractUrl class depends on WebCrawler class; it was also aggregating with url class.
- The different button-event method will use lambda expression to implement functioning.

There is a **UML diagram** below:



- The Main class will register any required event listeners on all the **Buttons** components.

The below is the Main-class **Sequence Diagram**:

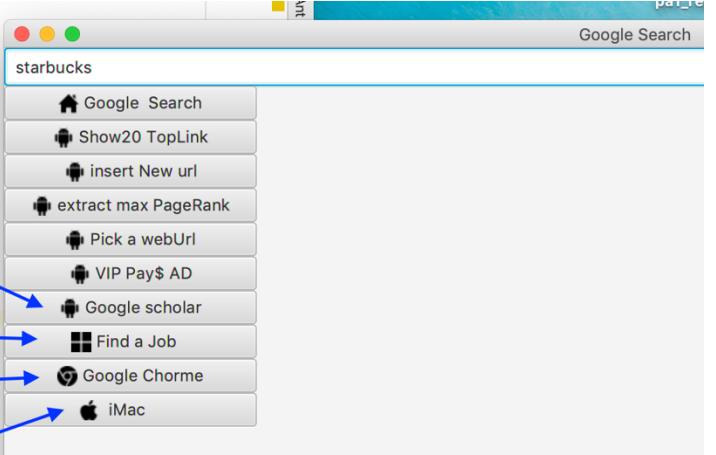


- All buttons will each fulfill their purpose to finish different functioning except the **SearchButton** execute the primary task.
- The heap class will call itself by invoking the method HeapSort in ascending order when the user hope to resort the `ArrayList<url>` to execute their appeals.
- In the lifetime, the `extractUrl` always play a large role to provide the web url that the users hope to get.

3. Implementing functioning

- The user UI Interface will hold the primary buttons: Google Search, Show20topLink, insertNewUrl, extract max PageRank, Pick a webUrl, and VIP Pay\$ AD.

- Note: the Google scholar, Find a Job, Google Chorme, iMac is only to decorate the user interface. They aren't meaningful.



```

@Override
public void start(Stage primaryStage) throws FileNotFoundException {
    searchText = new TextField("input your keyword: ");
    searchText.setPrefHeight(30);

    FileInputStream input = new FileInputStream( name: "home.png");
    Image image = new Image(input);
    ImageView imageView = new ImageView(image);
    searchBtn = new Button( text: "Google Search", imageView);
    searchBtn.setPrefWidth(200);

    FileInputStream input1 = new FileInputStream( name: "android.png");
    Image image1 = new Image(input1);
    ImageView imageView1 = new ImageView(image1);
    searchBtn1 = new Button( text: "Google scholar", imageView1);
    searchBtn1.setPrefWidth(200);

    FileInputStream input2 = new FileInputStream( name: "google.png");
    Image image2 = new Image(input2);
    ImageView imageView2 = new ImageView(image2);
    searchBtn2 = new Button( text: "Find a Job ", imageView2);
    searchBtn2.setPrefWidth(200);

    FileInputStream input3 = new FileInputStream( name: "chorme.png");
    Image image3 = new Image(input3);
    ImageView imageView3 = new ImageView(image3);
    searchBtn3 = new Button( text: "Google Chorme", imageView3);
    searchBtn3.setPrefWidth(200);

    FileInputStream input4 = new FileInputStream( name: "apple.png");
    Image image4 = new Image(input4);
    ImageView imageView4 = new ImageView(image4);
    searchBtn4 = new Button( text: " iMac      ", imageView4);
    searchBtn4.setPrefWidth(200);

    FileInputStream input5 = new FileInputStream( name: "android.png");
    Image image5 = new Image(input5);
    ImageView imageView5 = new ImageView(image5);
    show20webButton = new Button( text: "Show20 TopLink", imageView5);
    show20webButton.setPrefWidth(200);

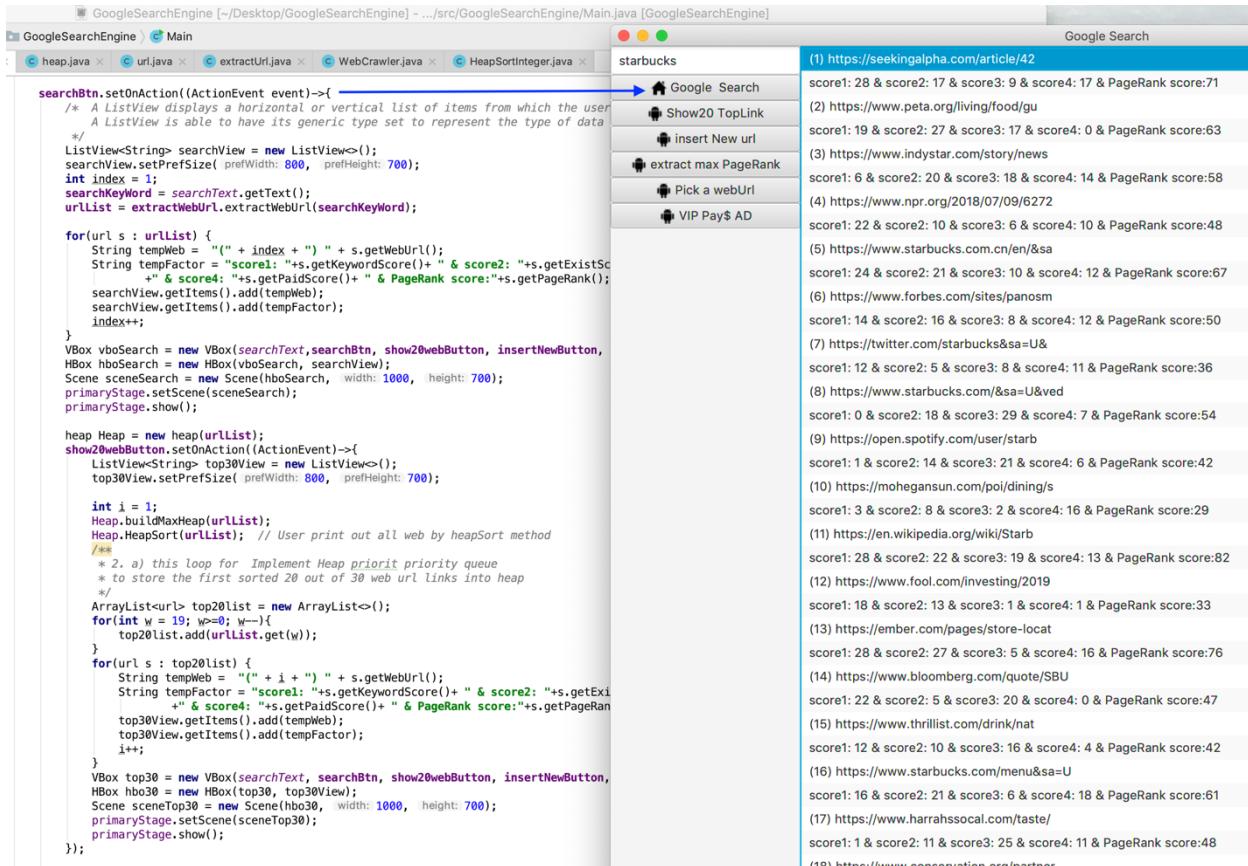
    FileInputStream input6 = new FileInputStream( name: "android.png");
    Image image6 = new Image(input6);
    ImageView imageView6 = new ImageView(image6);
    insertNewButton = new Button( text: "insert New url", imageView6);
    insertNewButton.setPrefWidth(200);

    FileInputStream input7 = new FileInputStream( name: "android.png");
    Image image7 = new Image(input7);
    Main > start()
}

```

This four buttons with icons have not meaning, only decorating user interface !

- When clicking the SearchButton, it will fire an event to all of the registered listeners with the object being updated, the cation of the update, the value before, and the value now.
- → calling the **extractWebUrl(KeyWord)** to get the web url into the ArrayList.
- → Printing out the first 30 elements.



- Implementing ‘Show20 TopLink’ button → calling **Heap.buildMaxHeap(A)** and **Heap.HeapSort(A)**. Displaying the 20 out of 30 web url by descending order since the **HeapSort** method sorting the **ArrayList** in desceding order.

- The web url elements sorting like this: PageRank 61 → PageRank 58 → PageRank 54 → PageRank 52 → PageRank 51 → PageRank 50

The screenshot shows a Java IDE (NetBeans) and a JavaFX application running simultaneously.

Java IDE (Left):

```

heap Heap = new heap(urlList);
show20webButton.setOnAction(ActionEvent->{
    ListView<String> top30View = new ListView<>();
    top30View.setPrefSize( prefWidth: 800, prefHeight: 700);

    int i = 1;
    Heap.buildMaxHeap(urlList);
    Heap.HeapSort(urlList); // User print out all web by heapSort method
    /**
     * 2. a) this loop for Implement Heap priority queue
     * to store the first sorted 20 out of 30 web url links into heap
     */
    ArrayList<url> top20list = new ArrayList<>();
    for(int w = 19; w>0; w--){
        top20list.add(urlList.get(w));
    }
    for(url s : top20list) {
        String tempWeb = "(" + i + " ) " + s.getWebUrl();
        String tempFactor = "score1: "+s.getKeywordsScore()+" & score2: "+s.getExi
        +" & score4: "+s.getPaidScore()+" & PageRank score:"+s.getPageRan
        top30View.getItems().add(tempWeb);
        top30View.getItems().add(tempFactor);
        i++;
    }
    VBox top30 = new VBox(searchText, searchBtn, show20webButton, insertNewButton,
    HBox hb30 = new HBox(top30, top30View);
    Scene sceneTop30 = new Scene(hb30, width: 1000, height: 700);
    primaryStage.setScene(sceneTop30);
    primaryStage.show();
};

insertNewButton.setOnAction(ActionEvent->{
    ListView<String> addView = new ListView<>();
    addView.setPrefSize( prefWidth: 800, prefHeight: 700);
    url added = new url(searchText.getText());
    Heap.MaxHeapInsert(urlList,added);
    int j=1;
    for(url s : urlList) {
        String tempWeb = "(" + j + " ) " + s.getWebUrl();
        String tempFactor = "score1: "+s.getKeywordsScore()+" & score2: "+s.getExi
        +" & score4: "+s.getPaidScore()+" & PageRank score:"+s.getPageRan
        addView.getItems().add(tempWeb);
        addView.getItems().add(tempFactor);
        j++;
    }
    VBox insert = new VBox(searchText, searchBtn, show20webButton, insertNewButton
    HBox addh = new HBox(insert, addView);
    Scene addScene = new Scene(addh, width: 1000, height: 700);
    primaryStage.setScene(addScene);
    primaryStage.show();
};

getMaxButton.setOnAction(ActionEvent->{
    ListView<String> maxView = new ListView<>();
    maxView.setPrefSize( prefWidth: 800, prefHeight: 700);
    Heap.buildMaxHeap(urlList);
    url max = Heap.heapMaximum(urlList);
    String maxWeb = max.getWebUrl();
    maxWeb = "[max PageRank] "+maxWeb;
    String tempFactor = "score1: "+max.getKeywordsScore()+" & score2: "+max.getExi
    +" & score4: "+max.getPaidScore()+" & PageRank score:"+max.getPageRan
    maxView.getItems().add(maxWeb);
};

start() > (ActionEvent event) ->{} > (ActionEvent) ->{}

```

JavaFX Application (Right):

The application window has a title bar "Google Search" and a menu bar with items: Main, File, Edit, View, Insert, Tools, Help. A toolbar below the menu bar contains icons for "Google Search", "Show20 TopLink", "Insert New url", "extract max PageRank", "Pick a webURL", and "VIP Pay\$ AD".

The main content area displays a list of 20 web URLs, each with its URL, score, and PageRank. The list starts with "starbucks" and ends with "mohegansun.com/poi/dining/s". The list is ordered by descending PageRank.

A red arrow points from the line of code `top30View.getItems().add(tempWeb);` in the Java code to the "Show20 TopLink" icon in the toolbar. Another red arrow points from the line of code `top30View.getItems().add(tempFactor);` to the "extract max PageRank" icon in the toolbar. A third red arrow points from the text "This show Top20 event to show 20 out of 30 web url by descending order" to the "Show20 TopLink" icon.

Rank	URL	Score	PageRank
1	https://www.starbucks.com/menu&sa=U	16 & score2: 21 & score3: 6 & score4: 18 & PageRank score:61	61
2	https://www.forbes.com/companies/st	6 & score2: 2 & score3: 24 & score4: 26 & PageRank score:58	58
3	https://www.indystar.com/story/news	6 & score2: 20 & score3: 18 & score4: 14 & PageRank score:58	58
4	https://www.starbucks.com/&sa=U&ved	0 & score2: 18 & score3: 29 & score4: 7 & PageRank score:54	54
5	https://www.linkedin.com/company/st	3 & score2: 10 & score3: 13 & score4: 28 & PageRank score:54	54
6	https://hip2save.com/2019/03/22/fre	29 & score2: 16 & score3: 5 & score4: 2 & PageRank score:52	52
7	https://techcrunch.com/2019/03/20/s	10 & score2: 11 & score3: 13 & score4: 17 & PageRank score:51	51
8	https://www.forbes.com/sites/panosm	14 & score2: 16 & score3: 8 & score4: 12 & PageRank score:50	50
9	https://sundevildining.asu.edu/hour	21 & score2: 1 & score3: 27 & score4: 0 & PageRank score:49	49
10	https://www.harrasssocial.com/taste/	1 & score2: 11 & score3: 25 & score4: 11 & PageRank score:48	48
11	https://www.npr.org/2018/07/09/6272	22 & score2: 10 & score3: 6 & score4: 10 & PageRank score:48	48
12	https://www.bloomberg.com/quote/SBU	22 & score2: 5 & score3: 20 & score4: 0 & PageRank score:47	47
13	https://www.starbucksforlife.com/%3	19 & score2: 8 & score3: 16 & score4: 3 & PageRank score:46	46
14	https://www.thrillist.com/drink/nat	12 & score2: 10 & score3: 16 & score4: 4 & PageRank score:42	42
15	https://open.spotify.com/user/starb	1 & score2: 14 & score3: 21 & score4: 6 & PageRank score:42	42
16	https://nyu.campusdish.com/en/Locati	0 & score2: 4 & score3: 14 & score4: 22 & PageRank score:40	40
17	https://www.fastcompany.com/company	1 & score2: 12 & score3: 25 & score4: 1 & PageRank score:39	39
18	https://twitter.com/starbucks&sa=U&	12 & score2: 5 & score3: 8 & score4: 11 & PageRank score:36	36
19	https://www.fool.com/investing/2019	18 & score2: 13 & score3: 1 & score4: 1 & PageRank score:33	33
20	https://mohegansun.com/poi/dining/s	3 & score2: 8 & score3: 2 & score4: 16 & PageRank score:29	29

- Implementing “insert New url” button, insert Event → input <https://www.apple.com>
→ click **insert New url** Button → added a new web url into the heap By call MaxHeapInsert(A, key) method to implement the (31) element into the heap Tree !

The screenshot shows a Java application window titled "Google Search". The window contains a list of URLs with their scores and PageRank values. At the top of the list is the URL "https://www.apple.com". Below the list are several buttons: "insert New url", "extract max PageRank", "Pick a webUrl", and "VIP Pay\$ AD".

```

insertNewButton.setOnAction((ActionEvent)->{
    ListView<String> addView = new ListView<>();
    addView.setPrefSize( 800, prefHeight: 700);
    url added = new url(searchText.getText());
    Heap.MaxHeapInsert(urlList,added);
    int j=1;
    for(url s : urlList) {
        String tempWeb = "(" + j + " ) " + s.getWebUrl();
        String tempFactor = "score1: "+s.getKeywordScore()+" &
                           score4: "+s.getPaidScore()+" & PageRank ";
        addView.getItems().add(tempWeb);
        addView.getItems().add(tempFactor);
        j++;
    }
    VBox insert = new VBox(searchText, searchBtn, show20webButtons);
    HBox addh = new HBox(insert, addView);
    Scene addScene = new Scene(addh, width: 1000, height: 700);
    primaryStage.setScene(addScene);
    primaryStage.show();
});

getMaxButton.setOnAction((ActionEvent)->{
    ListView<String> maxView = new ListView<>();
    maxView.setPrefSize( 800, prefHeight: 700);
    Heap.buildMaxHeap(urlList);
    url max = Heap.heapMaximum(urlList);
    String maxWeb = max.getWebUrl();
    maxWeb = "[max PageRank]: "+maxWeb;
    String tempFactor = "score1: "+max.getKeywordScore()+" &
                           score4: "+max.getPaidScore()+" & PageRank ";
    maxView.getItems().add(maxWeb);
    maxView.getItems().add(tempFactor);

    VBox maxvb = new VBox(searchText, searchBtn, show20webButtons);
    HBox maxhb = new HBox(maxvb, maxView);
    Scene maxScene = new Scene(maxhb, width: 1000, height: 700);
    primaryStage.setScene(maxScene);
    primaryStage.show();
});

pickButton.setOnAction((ActionEvent )->{
    ListView<String> pickView = new ListView<>();
    pickView.setPrefSize( 800, prefHeight: 700);
    String str = searchText.getText();
    pickIndex = Integer.parseInt(str) - 1;
    url pickUrl = urlList.get(pickIndex);
    String tempUrl = "[Picked Original website]: " + pickUrl.getWebUrl();
    String tempFactor = "score1: "+pickUrl.getKeywordScore()+" &
                           score4: "+pickUrl.getPaidScore()+" & PageRank ";
    pickView.getItems().add(tempUrl);
    pickView.getItems().add(tempFactor);

    VBox pickvb = new VBox(searchText, searchBtn, show20webButtons);
    HBox pickhb = new HBox(pickvb, pickView);
    Scene pickScene = new Scene(pickhb, width: 1000, height: 700);
    primaryStage.setScene(pickScene);
    primaryStage.show();
});

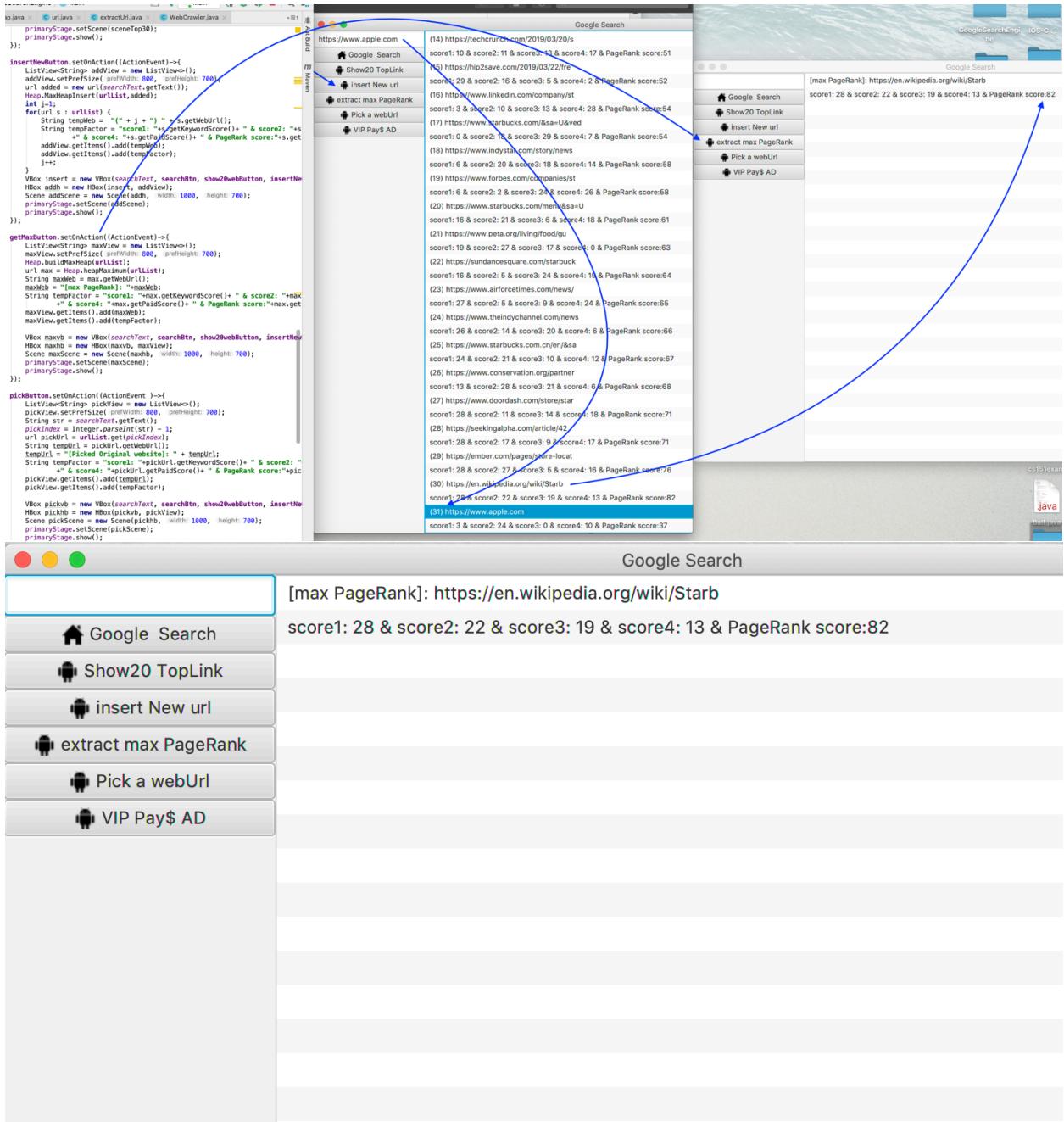
t1) > (ActionEvent event) -> {...} > (ActionEvent) -> {...}

```

	URL	score1	score2	score3	score4	PageRank
(14)	https://techcrunch.com/2019/03/20/s	10	11	13	17	51
(15)	https://hip2save.com/2019/03/22/fre	29	16	5	2	52
(16)	https://www.linkedin.com/company/st	3	10	13	28	54
(17)	https://www.starbucks.com/&sa=U&ved	0	18	29	7	54
(18)	https://www.indystar.com/story/news	6	20	18	14	58
(19)	https://www.forbes.com/companies/st	6	2	24	26	58
(20)	https://www.starbucks.com/menu/&sa=U	16	21	6	18	61
(21)	https://www.peta.org/living/food/gu	19	27	17	0	63
(22)	https://sundancesquare.com/starbuck	16	5	24	19	64
(23)	https://www.airforcetimes.com/news/	27	5	9	24	65
(24)	https://www.theindychannel.com/news	26	14	20	6	66
(25)	https://www.starbucks.com/cn/en/&sa	24	21	10	12	67
(26)	https://www.conservation.org/partner	13	28	21	6	68
(27)	https://www.doordash.com/store/star	28	11	14	18	71
(28)	https://seekingalpha.com/article/42	28	17	9	17	71
(29)	https://ember.com/pages/store-locat	28	27	5	16	76
(30)	https://en.wikipedia.org/wiki/Starb	28	22	19	13	82
(31)	https://www.apple.com	3	24	0	10	37

- Implementing “getMaxButton” → click button → calling Heap.**buildMaxHeap(A)**

- calling Heap.heapMaximum(A) method to get the max element
- displaying the result on the UI interface.



- Implementing pickButton and vipButton, → clicking “Pick a webUrl” → pick [\[www.sfgate.com\]](https://www.sfgate.com) → clicking “VIP Pay\$ AD” → add 33 rank score into original 27 rank → $33 + 27 = 60$
- Including two steps: (1) → pick; (2) → pay \$; (3) calculate total PageRank
- Create a new TreeMap → call method **TreeMap.put(key: Integer, Value: Integer)**; key using the index , value using the **getPageRank()**
Finding the key → then finding the value → increasing the PageRank

```

pickButton.setOnAction((ActionEvent )->{
    ListView<String> pickView = new ListView<>();
    pickView.setPrefSize(prefWidth: 800, prefHeight: 700);
    String str = searchText.getText();
    pickIndex = Integer.parseInt(str) - 1;
    url pickUrl = urlList.get(pickIndex);
    String tempUrl = "[Picked Original website]: " + tempUrl;
    String tempFactor = "score1: " + pickUrl.getKeywordScore() +
        " & score2: " + pickUrl.getPaidScore() + " & PageRank";
    pickView.getItems().add(tempUrl);
    pickView.getItems().add(tempFactor);
    VBox pickvb = new VBox(searchText, searchBtn, show20webButton);
    HBox pickhb = new HBox(pickvb, pickView);
    Scene pickScene = new Scene(pickhb, width: 1000, height: 700);
    primaryStage.setScene(pickScene);
    primaryStage.show();
}

vipButton.setOnAction((ActionEvent actionEvent)->{
    TreeMap<Integer, Integer>treeMap = new TreeMap<>();
    ListView<String> vipView = new ListView<>();
    vipView.setPrefSize(prefWidth: 800, prefHeight: 700);
    String paidMoney = searchText.getText();
    int increase = Integer.parseInt(paidMoney);

    Heap.buildMaxHeap(urlList);
    Heap.HeapSort(urlList);
    for(int p = 0; p<urlList.size(); p++){
        if((p+1) == pickIndex){
            Integer temp = urlList.get(pickIndex).getPageRank() + increase;
            treeMap.put(pickIndex, temp);
        }else{
            treeMap.put(p+1, urlList.get(p).getPageRank());
        }
    }
    int i = 1;
    for(url s : urlList) {
        String tempWeb = "(" + i + ")" + s.getWebUrl();
        String vipFactor = "score1: "+s.getKeywordScore() + " & score2: "+s.getExistScore() +
            " & score4: "+s.getPaidScore() + " & PageRank score:" +treeMap.get(i);
        vipView.getItems().add(tempWeb);
        vipView.getItems().add(vipFactor);
        i++;
    }
}

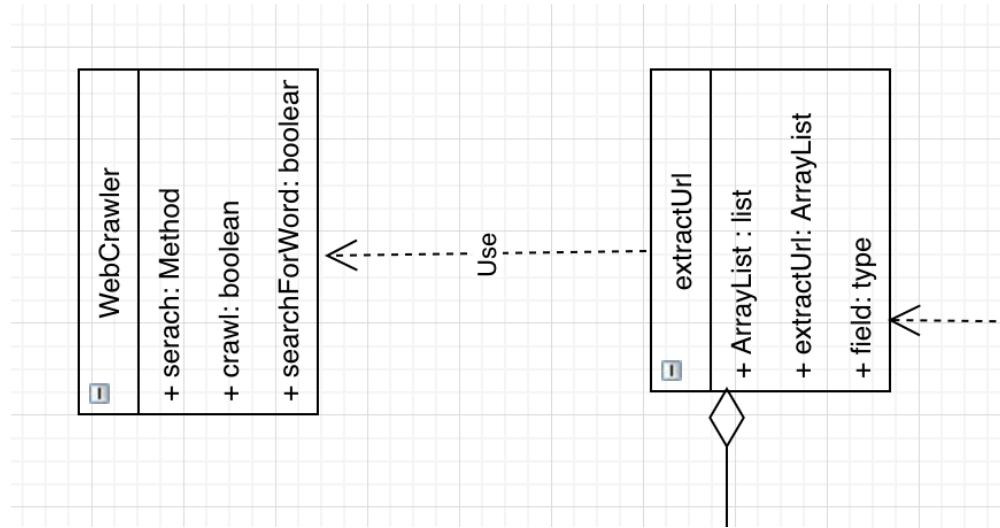
VBox vipvb = new VBox(searchText, searchBtn, show20webButton, insertNewButton, getHBox);
HBox viphb = new HBox(vipvb, vipView);
Scene vipScene = new Scene(viphb, width: 1000, height: 700);
primaryStage.setScene(vipScene);
primaryStage.show();
});

```

- To summary, so far so good! All functioning testing sucessfully. I have spent more than 120 hours!

4. classes/subroutines/function class:

- Class **extractUrl** → depending on Class **WebCrawler** :
 extractUrl's function **extractWebUrl(String keyword)** will help me get the website's address .
 constructor a new WebCrawler → calling wc.search()→ finding the required url elements → creating a new ArrayList<url> → A.add(All weburls).



```

/*
public ArrayList<url> extractWebUrl(String keyword){
    int index = 1;
    WebCrawler c = new WebCrawler(keyword);
    c.search();
    Set<String> set = c.getUrls();

    for(String s : set){
        if(index <= 30){
            s = "https://" + s;
            index++;
            url weburl = new url(s);
            list.add(weburl);
        }
    }
    return list;
}
  
```

- Class **WebCrawler** has a constructor **WebCrawler(keyword)** to hold the keyword that the user inputed through Main class calling the **Handling Events**.
 • It have imported the Jsoup lirbary, Jsoup lirbary is to help Crawler to get the url.

```

import java.io.IOException;
import java.util.HashSet;
import java.util.Set;

import org.jsoup.Connection;
import org.jsoup.Jsoup;
import org.jsoup.nodes.Document;
import org.jsoup.nodes.Element;
import org.jsoup.select.Elements;

import java.util.regex.Matcher;
import java.util.regex.Pattern;

/**
 * A webCrawler class to get the web address with the given keywords
 * This class is to help get the URL, and testing
 */
public class WebCrawler {

    private String url;
    private String keyword;
    private Set<String> urls = new HashSet<>();

    private static final String USER_AGENT = "Mozilla/5.0 (compatible; Googlebot/2.1; +http://www.google.com/bot.html)";
    private Document htmlDocument;
    private static Pattern patternDomainName;
    private Matcher matcher;
    private static final String DOMAIN_NAME_PATTERN = "([a-zA-Z0-9]([a-zA-Z0-9\\-]{0,61}[a-zA-Z0-9])?\\.){1,6}[a-zA-Z]{2,6}";

    static {
        patternDomainName = Pattern.compile(DOMAIN_NAME_PATTERN);
    }
}

```

- The crawl(String url) method will crawl the links and put them in to a set to keep; Give it a URL and it makes an HTTP request for a web page.

```

public boolean crawl(String url) {
    try {

        Connection connection = Jsoup.connect(url).userAgent(USER_AGENT);
        final Document htmlDocument = connection.timeout(5000).get();
        this.htmlDocument = htmlDocument;

        if (connection.response().statusCode() == 200) {
            System.out.println("\n**Visiting** Received web page at " + url);
        }
        if (!connection.response().contentType().contains("text/html")) {
            System.out.println("**Failure** Retrieved something other than HTML");
            return false;
        }

        Elements linksOnPage = htmlDocument.select(cssQuery: "a[href]");
        System.out.println("Found (" + linksOnPage.size() + ") links");

        for (Element link : linksOnPage) {
            String temp = link.attr(attributeKey: "href");
            if (temp.startsWith("/url?q=http")) {
                this.urls.add(getDomainName(temp));
            }
        }
        return true;
    } catch (IOException ioe) {
        return false;
    }
}

```

- Heap class including seven method:
 - maxHeapify → is a helper function to help buildMaxHeap to create a Heap Tree.

```

public void maxHeapify(ArrayList<Integer> A, int i) {
    int largest;
    int l = 2 * i + 1;
    int r = 2 * i + 2;

    if (l < heapSize && A.get(l) > A.get(i)) {
        largest = l;
    } else {
        largest = i;
    }

    if (r < heapSize && A.get(r) > A.get(largest)) {
        largest = r;
    }
    if (largest != i) {
        Collections.swap(A, i, largest);
        maxHeapify(A, largest);
    }
}

```

2. buildMaxHeap → we can build a max-heap from an unordered array in linear time.

```

public void buildMaxHeap(ArrayList<Integer> A) {

    this.heapSize = A.size();
    for (int i = (A.size() / 2) - 1; i >= 0; i--) {
        maxHeapify(A, i);
    }
}

```

3. HeapSort → After calling buildMaxHeap → sort an array in ascending order.

```

public void HeapSort(ArrayList<Integer> A) {
    buildMaxHeap(A);
    for (int i = A.size() - 1; i > 0; i--) {
        Collections.swap(A, i, j: 0);
        heapSize--;
        maxHeapify(A, i: 0);
    }
}

```

4. HeapExtractMax → if heap.size < -1 throw new RuntimeException("heap underflow") → calling heapMaximum(A) → put the last element on the first position to replace the maximum → calling maxHeapify(A, 0) → return maximum

```

public url HeapExtractMax(ArrayList<url> A) {
    if (heapSize < 1) {
        throw new RuntimeException("heap underflow");
    }
    url max = heapMaximum(A);
    A.set(0, A.get(heapSize - 1));
    heapSize--;
    maxHeapify(A, i: 0);
    return max;
}

```

5. heapIncreaseKey → Increases the value of the element I's key to the new value k, where k>= I's current key value

```

public void heapIncreaseKey(ArrayList<url> A, int i, url key) {
    if (key.getPageRank() < A.get(i).getPageRank()) {
        throw new RuntimeException("new key is smaller than current key");
    }
    A.set(i, key);
    while (i > 0 && A.get(getParent(i)).getPageRank() < A.get(i).getPageRank()) {
        Collections.swap(A, i, getParent(i));
        i = getParent(i);
    }
}

```

6. MaxHeapInsert → insert the key into the arraylist

```

public void MaxHeapInsert(ArrayList<url> A, url key) {
    heapSize++;
    A.add(key);
    A.get(heapSize - 1).setPageRank((int) Double.NEGATIVE_INFINITY);
    heapIncreaseKey(A, heapSize, key); //then give the leaf real value
}

```

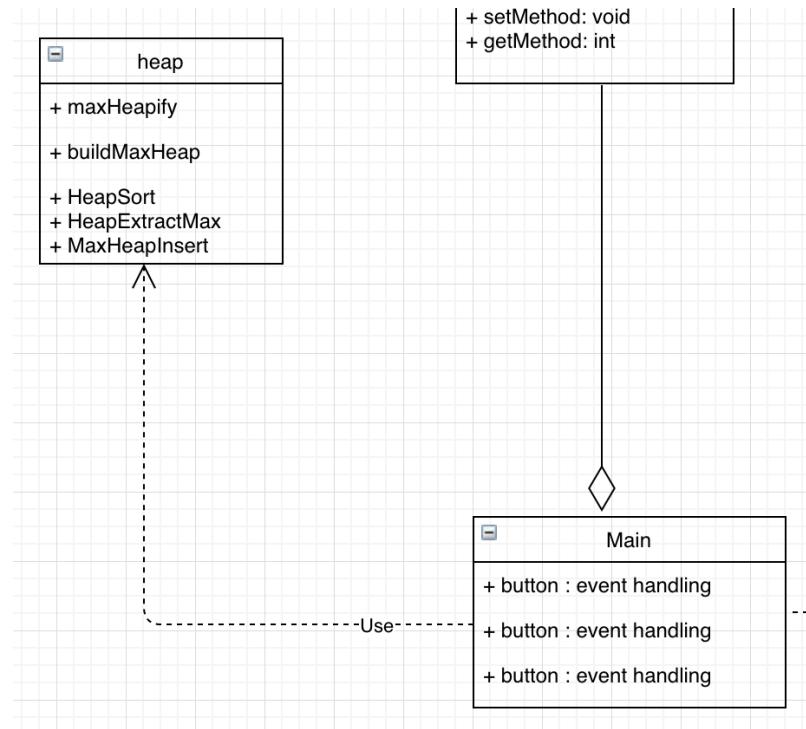
7. heapMaximum → return the element of arraylist with the largest key

```

public url heapMaximum(ArrayList<url> A){
    return A.get(0);
}

```

- The Main class need Heap to implement different functioning : get Maximum, sort an array, in brief, Main depents on Heap.



Main class method: start(Stage primaryStage)

- ListView → adding datas into the UI interface
- searchText.getText()→ save the data that the user input
- Vbox → vertical layout all buttons; the buttons will be vertical; contain the searchText
- Hbox → making vertical button on the left, the console of the UI interface will on the right
- Scene → contains the Vbox, Hbox, ListView
- primaryStage → holds all above components.

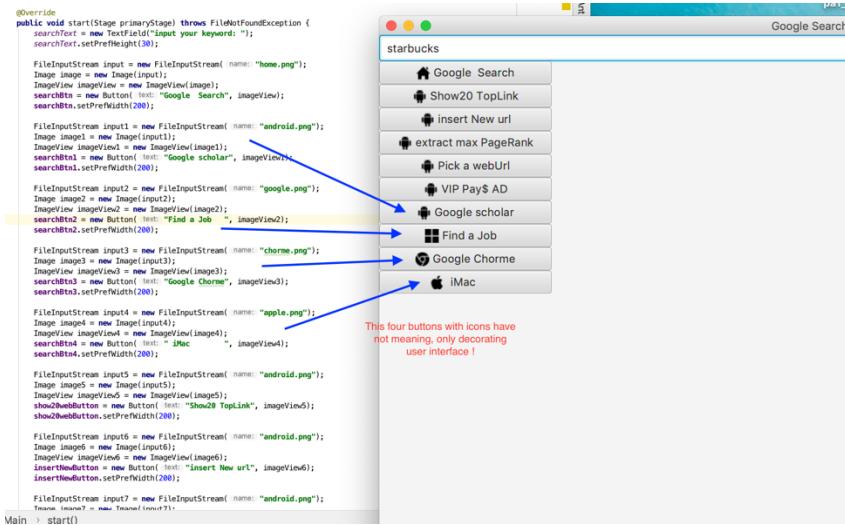
```

searchBtn.setOnAction((ActionEvent event) -> {
    /* A ListView displays a horizontal or vertical list of items from which the user may select
       A ListView is able to have its generic type set to represent the type of data in the backing
     */
    ListView<String> searchView = new ListView<>();
    searchView.setPrefSize(prefWidth: 800, prefHeight: 700);
    int index = 1;
    searchKeyWord = searchText.getText();
    urlList = extractWebUrl.extractWebUrl(searchKeyWord);

    for(url s : urlList) {
        String tempWeb = "(" + index + " ) " + s.getWebUrl();
        String tempFactor = "score1: "+s.getKeywordScore()+" & score2: "+s.getExistScore()+" & score3: "+s.getPaidScore()+" & score4: "+s.getPageRank();
        searchView.getItems().add(tempWeb);
        searchView.getItems().add(tempFactor);
        index++;
    }
    VBox vboSearch = new VBox(searchText, searchBtn, show20webButton, insertNewButton, getMaxButton, iMac);
    HBox hboSearch = new HBox(vboSearch, searchView);
    Scene sceneSearch = new Scene(hboSearch, width: 1000, height: 700);
    primaryStage.setScene(sceneSearch);
    primaryStage.show();
}

```

- All buttons:



- Show20webButton : displays the top 20 web url in descending order
→ call HeapSort → arrayList adds all elements in descending order → print data
 - ListView → adding datas into the UI interface
 - searchText.getText() → save the data that the user input
 - Vbox → vertical layout all buttons; the buttons will be vertical; contain the searchText
 - Hbox → making vertical button on the left, the console of the UI interface will on the right
 - Scene → contains the Vbox, Hbox, ListView
 - primaryStage → holds all above components.
-
- insertNewButton → add a new url element
 - getMaxButton → get the maximum url element
-
- ListView → adding datas into the UI interface
 - searchText.getText() → save the data that the user input
 - Vbox → vertical layout all buttons; the buttons will be vertical; contain the searchText
 - Hbox → making vertical button on the left, the console of the UI interface will on the right
 - Scene → contains the Vbox, Hbox, ListView
 - primaryStage → holds all above components.
-
- pickButton → pick a required url element
 - vipButton → pay the money for google AD
 - Note: vipButton inside the pickButton
-
- ListView → adding datas into the UI interface
 - searchText.getText() → save the data that the user input
 - Vbox → vertical layout all buttons; the buttons will be vertical; contain the searchText
 - Hbox → making vertical button on the left, the console of the UI interface will on the right
 - Scene → contains the Vbox, Hbox, ListView
 - primaryStage → holds all above components.

- Class url have a constructor: initialize score1 score2 score3 score 4

```
public url(String theUrl){
    webUrl = theUrl;
    paidScore = (int) (Math.random()* 30);
    keywordScore = (int) (Math.random()* 30);
    existScore = (int) (Math.random()* 30);
    linkScore = (int) (Math.random()* 30);
}
```

- Manipulate method && Asscess Method

```
/* manipulate method */

public void setKeywordScore(int ks) { keywordScore = ks; }

public void setExistScore(int es) { existScore = es; }

public void setLinkScore(int ls) { linkScore = ls; }

public void setPaidScore(int ps) { paidScore = ps; }

public void setPageRank(int pr) { pageRank = pr; }

public void setIndex(int i) { index = i; }

/* Asscess Method*/

public int getKeywordScore() { return keywordScore; }

public int getExistScore() { return existScore; }

public int getLinkScore() { return linkScore; }

public int getPaidScore() { return paidScore; }

public int getIndex() { return index; }

public int getPageRank(){
    pageRank = keywordScore + existScore + linkScore + paidScore;
    return pageRank;
}

public String getWebUrl() { return webUrl; }
```

5. Self-testing screen shots

- BuildMaxHeap(A) → A array with decending order

The screenshot shows the code for the `buildMaxHeap` method:

```
public void buildMaxHeap(ArrayList<Integer> A) {
    this.heapSize = A.size();
    for (int i = (A.size() / 2) - 1; i >= 0; i--) {
        maxHeapify(A, i);
    }
}
```

To the right is the terminal output for the `debugHeap` run:

```
Run: debugHeap x
/Library/Java/JavaVirtualMachines/jdk1.8.0_201.jdk/Contents/Home/bin/
Original heap print:
[10, 20, 33, 44, 111, 222, 33, 19, 55, 132]
Test build-max-heap
Expected build-max-heap: [222, 132, 33, 55, 111, 10, 33, 19, 44, 20]
[222, 132, 33, 55, 111, 10, 33, 19, 44, 20]
Process finished with exit code 0
```

- HeapSort(A) → An array sorts with acending order

The screenshot shows the code for the `HeapSort` method:

```
public void HeapSort(ArrayList<Integer> A) {
    buildMaxHeap(A);
    for (int i = A.size() - 1; i > 0; i--) {
        Collections.swap(A, i, 0);
        heapSize--;
        maxHeapify(A, 0);
    }
}
```

To the right is the terminal output for the `debugHeap` run:

```
Run: debugHeap x
/Library/Java/JavaVirtualMachines/jdk1.8.0_201.jdk/Contents/Home/bin/
Original heap list is: [222, 132, 33, 55, 111, 10, 33, 19, 44, 20]
Expected heap-sort: [10, 19, 20, 33, 33, 44, 55, 111, 132, 222]
[10, 19, 20, 33, 33, 44, 55, 111, 132, 222]
Process finished with exit code 0
```

- heapMaximum(A) → get the maximum

The screenshot shows the code for the `heapMaximum` and `getParent` methods:

```
public Integer heapMaximum(ArrayList<Integer> A){
    return A.get(0);
}

public int getParent(int i) {
    return (i - 1 / 2);
}
```

To the right is the terminal output for the `debugHeap` run:

```
Run: debugHeap x
testHeap.buildMaxHeap(a);
System.out.println("Expected heap-Maximum: 222");
System.out.println("Current Heap-Maximum: "+testHeap.heapMaximum(a));
Run: debugHeap x
/Library/Java/JavaVirtualMachines/jdk1.8.0_201.jdk/Contents/Home/bin/java ...
Expected heap-Maximum: 222
Current Heap-Maximum: 222
```

- tesing heapIncresekey() && MaxHeapInsert()
→ adding a element

The screenshot shows the code for the `heapIncreaseKey` and `MaxHeapInsert` methods:

```
public void heapIncreaseKey(ArrayList<Integer> A, int i, Integer key) {
    i--;
    if (key < A.get(i)) {
        throw new RuntimeException("new key is smaller than current");
    }
    A.set(i, key);
    while (i > 0 && A.get(getParent(i)) < A.get(i)) {
        Collections.swap(A, i, getParent(i));
        i = getParent(i);
    }
}

public void MaxHeapInsert(ArrayList<Integer> A, Integer key) {
    heapSize++;
    A.add(key);
    A.set(heapSize-1, (int)Double.NEGATIVE_INFINITY);
    heapIncreaseKey(A, heapSize, key); //then give the leaf real
}
```

To the right is the terminal output for the `debugHeap` run:

```
Run: debugHeap x
External I 30
Scratches 37
38
39
40
41
42
43
44
45
46
47
Run: debugHeap x
testHeap.buildMaxHeap(a);
System.out.println("Original heap list is: " + a);
System.out.println("Expected heap list is: [222, 132, 33, 55, 111, 10, 33, 19, 44, 20]");
testHeap.MaxHeapInsert(a, key: 67);
System.out.println("Current Heap list is: "+a);
Run: debugHeap x
/Library/Java/JavaVirtualMachines/jdk1.8.0_201.jdk/Contents/Home/bin/java ...
Original heap list is: [222, 132, 33, 55, 111, 10, 33, 19, 44, 20]
Expected heap list is: [222, 132, 33, 55, 111, 10, 33, 19, 44, 20, 67]
Current Heap list is: [222, 132, 33, 55, 111, 10, 33, 19, 44, 20, 67]
Process finished with exit code 0
```

- Testing HeapExtractMax(A)

```

        for (int i = A.size() - 1; i >= 0; i--) {
            Collections.swap(A, i, 0);
            heapSize--;
            maxHeapify(A, 0);
        }

    public Integer HeapExtractMax(ArrayList<Integer> A) {
        if (heapSize < 1) {
            throw new RuntimeException("heap underflow");
        }
        Integer max = heapMaximum(A);
        A.set(0, A.get(heapSize - 1));
        heapSize--;
        maxHeapify(A, 0);
        return max;
    }

    public void heapIncreaseKey(ArrayList<Integer> A, int i,
        i--;
    }
}

```

```

37
38
39
40
41
42
43
44
45
46
47
48
49

    testHeap.buildMaxHeap(a);
    System.out.println("Original heap list is: " + a);

    System.out.println("Expected heap list is: [222, 132, 33, 55, 111, 10, 33, 19, 44, 20]");
    testHeap.MaxHeapInsert(a, key: 67);
    System.out.println("Current Heap list is: "+a);

    Integer temp = testHeap.HeapExtractMax(a);
    System.out.println("Expected heap list is: [222]");
    System.out.println(temp);

}

Run: debugHeap > main()
/Libraries/Java/JavaVirtualMachines/jdk1.8.0_201.jdk/Contents/Home/bin/java ...
Original heap list is: [222, 132, 33, 55, 111, 10, 33, 19, 44, 20]
Expected heap list is: [222, 132, 33, 55, 111, 10, 33, 19, 44, 20, 67]
Current Heap list is: [222, 132, 33, 55, 111, 10, 33, 19, 44, 20, 67]
Expected heap list is: [222]
222

```

- Testing WebCrawler

```

det 39
ext 40 >
Go 41
he 42
He 43
Ma 44
Ma 45
url 46
We 47
We 48
META 49
sampl 50
android.p 51
apple.pn 52
apple.pn 53
chromer 54
chromer 55
google.p 56
GoogleSt 57
home.on 57

public static void main(String[] args){
    Scanner input = new Scanner(System.in);
    ArrayList<String> urlList = new ArrayList<>();
    System.out.println("Please enter your keyword: ");
    String keyword = input.nextLine(); //toLowerCase();
    WebCrawler c = new WebCrawler(keyword);
    c.search();
    Set<String> set = c.getUrls();
    int index = 1;
    for(String urlstr : set){
        System.out.println("[+"+index+"] " + urlstr);
        index++;
    }
}

extractUrl > main()
Run: extractUrl <
/Libraries/Java/JavaVirtualMachines/jdk1.8.0_201.jdk/Contents/Home/bin/java ...
Please enter your keyword:
apple
Found (270) links
[1] support.apple.com/6sa=U&ved
[2] www.jambajuice.com/menu-and-
[3] www.usatoday.com/story/tech
[4] variety.com/2019/music/news
[5] www.gsmarena.com/apple_phon
[6] 9to5mac.com/2019/03/23/app
[7] www.atlasobscura.com/places
[8] t1.gstatic.com/images/3Fq30
[9] www.cultofmac.com/489269/sa
[10] techcrunch.com/2019/03/22/a
[11] www.vulture.com/2019/03/app
[12] www.laptopmag.com/reviews/t
[13] www.youtube.com/user/Apple&
[14] finance.yahoo.com/quote/aap
[15] www.cnn.com/tag/applesa=U&
[16] bestapples.com/6sa=U&ved=0a
[17] www.medicalnewstoday.com/ar
[18] www.macrumors.com/6sa=U&ved
[19] www.theatlantic.com/technol
[20] med.stanford.edu/news/all-ne
[21] www.nytimes.com/2019/03/20/
[22] markets.businessinsider.com
[23] www.cnet.com/news/apples-tv
[24] bigapplecircus.com/6sa=U&ve
[25] www.marketwatch.com/investi
[26] www.apple.com/shop/accessori
[27] www.apple.com/ipad/6sa=U&ve
[28] www.verizonwireless.com/sma
[29] www.reuters.com/article/us-

```

- Testing searchBtn → input “windows” out 30 url

```

searchBtn.setOnAction((ActionEvent event)->{
    /* A ListView displays a horizontal or vertical list of items from which the user may select
       A ListView is able to have its generic type set to represent the type of data in the backing model.
    */
    ListView<String> searchView = new ListView<>();
    searchView.setPrefSize( 800, prefHeight: 700);
    int index = 1;
    searchKeyWord = searchText.getText();
    urlList = extractWebUrl.extractWebUrl(searchKeyWord);

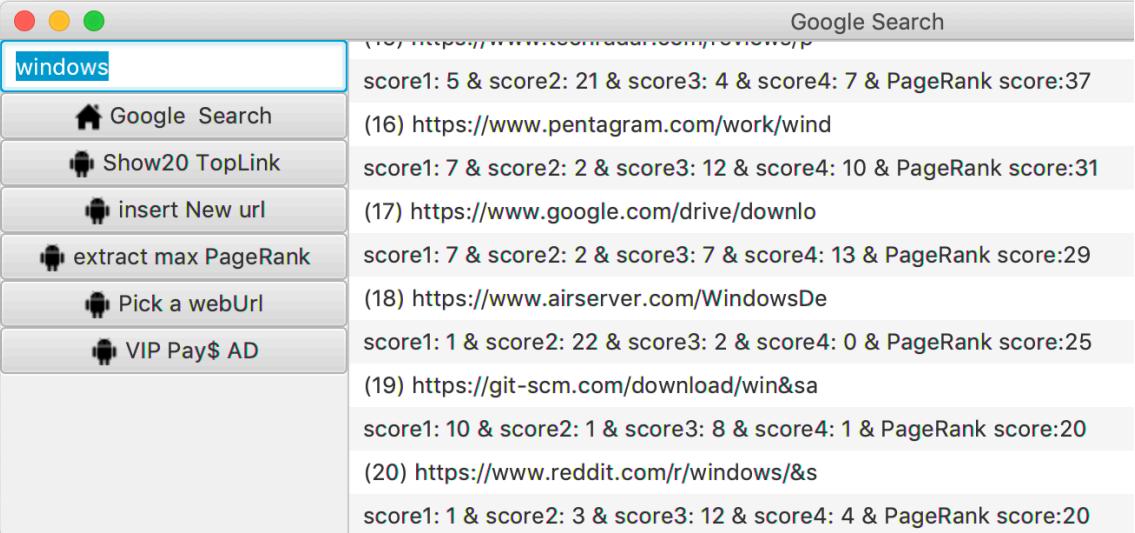
    for(url s : urlList) {
        String tempWeb = "(" + index + ") " + s.getWebUrl();
        String tempFactor = "score1: "+s.getKeywordScore()+" &
                           score2: "+s.getPaidScore()+" & PageRank score3: "+s.getPageRankScore()+" & score4: "+s.getScore();
        searchView.getItems().add(tempWeb);
        searchView.getItems().add(tempFactor);
        index++;
    }
    VBox vboSearch = new VBox(searchText,searchBtn, show20webButton);
    HBox hboSearch = new HBox(vboSearch, searchView);
    Scene sceneSearch = new Scene(hboSearch, width: 1000, height: 700);
    primaryStage.setScene(sceneSearch);
    primaryStage.show();

    heap Heap = new heap(urlList);
    show20webButton.setOnAction((ActionEvent)->{ ...
        primaryStage.close();
    });
}

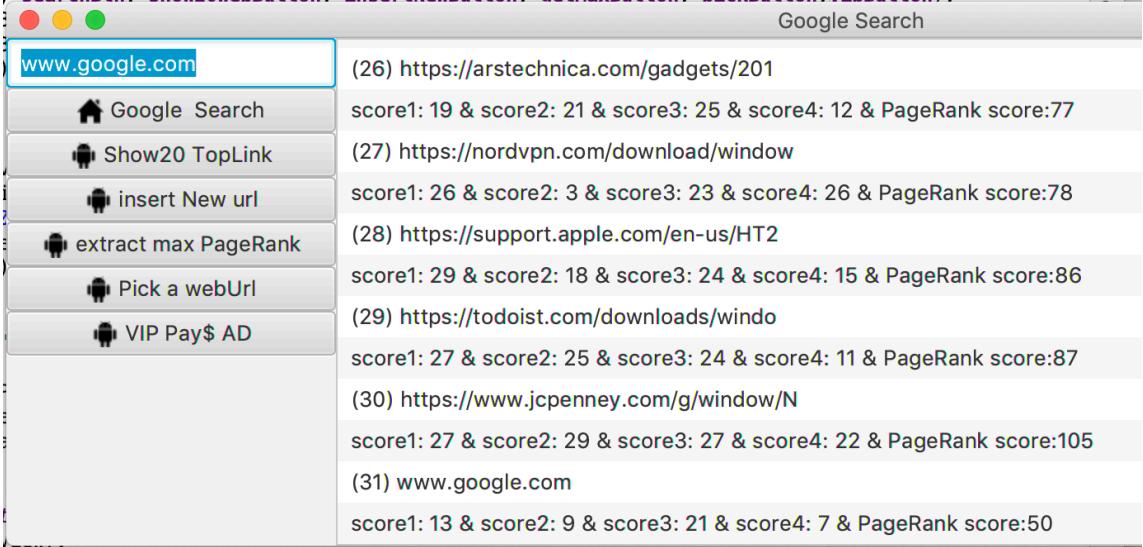
```

Rank	URL	Score1	Score2	Score3	Score4
1	https://www.papercut.com/support/re	16	24	23	2
2	https://www.techradar.com/reviews/p	5	21	4	7
3	https://www.pentagram.com/work/wind	7	2	12	10
4	https://nordvpn.com/download/window	26	3	23	26
5	https://www.yelp.com/search%3Fcfl%	15	25	3	11
6	https://github.com/Microsoft/calcul	15	23	19	6
7	https://todoist.com/downloads/windo	15	22	23	63
8	https://www.papercut.com/support/re	16	24	23	2
9	https://www.techradar.com/reviews/p	5	21	4	7
10	https://www.pentagram.com/work/wind	7	2	12	10
11	https://nordvpn.com/download/window	26	3	23	26
12	https://www.yelp.com/search%3Fcfl%	15	25	3	11
13	https://github.com/Microsoft/calcul	15	23	19	6
14	https://todoist.com/downloads/windo	15	22	23	63
15	https://www.papercut.com/support/re	16	24	23	2
16	https://www.techradar.com/reviews/p	5	21	4	7
17	https://www.pentagram.com/work/wind	7	2	12	10
18	https://www.airserver.com/WindowsDe	26	3	23	26
19	https://git-scm.com/download/win&s	1	22	2	0
20	https://www.reddit.com/r/windows/&s	10	1	8	1
21	https://www.reddit.com/r/windows/&s	1	3	12	4

- Testing show20webButton



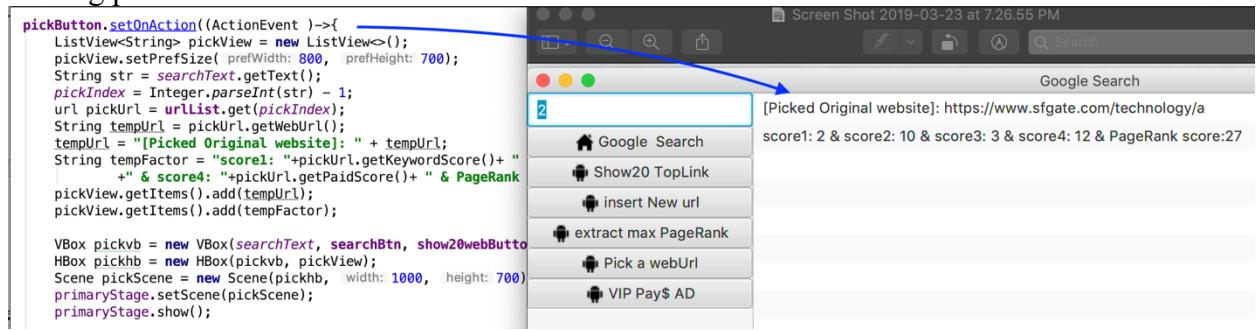
- Testing insertNewButton



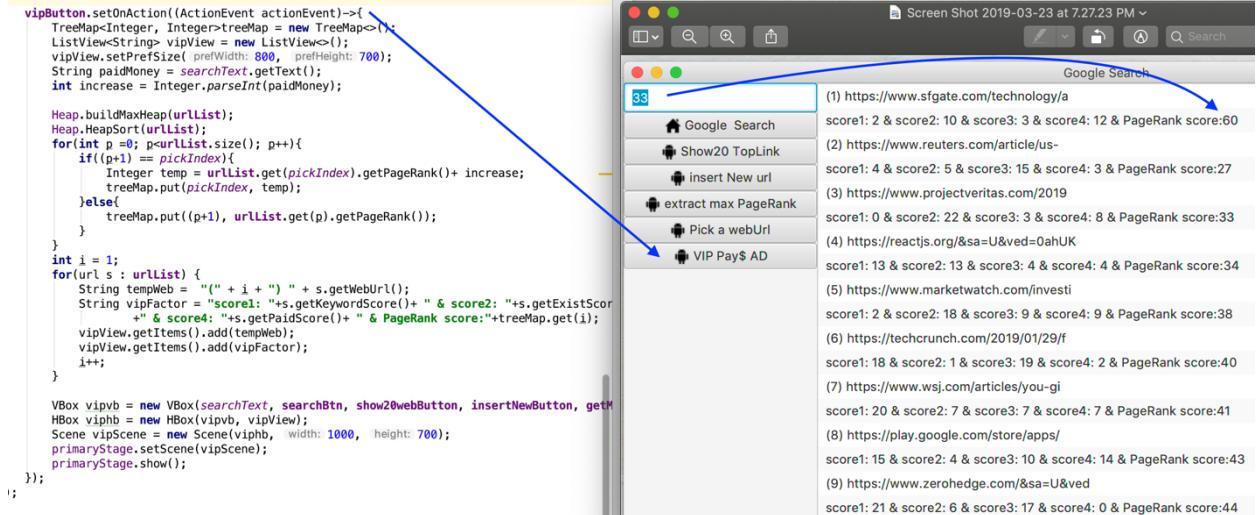
- Testing getMaxButton



- Testing pickButton



- Testing vipButton



6. Problems encountered during implementation

- Originally, I wrote all event method in Main class step by step
- → Firstly, I didn't know if the scope of searchButton will affect the following button events.
- → After struggling for more than 5 hours, I relized my mistake that the orther buttons Events must inside the searchButton's scope:

```
        HBox viphb = new HBox(vipvb, vipView);
        Scene vipScene = new Scene(viphb, width: 1000, height: 700);
        primaryStage.setScene(vipScene);
        primaryStage.show();
    });

};

VBox mainInterface = new VBox(searchText, searchBtn, show20webButton, insertNewButton, getMaxButton, pickButton,
    vipButton, searchBtn1, searchBtn2, searchBtn3, searchBtn4);
Scene scene = new Scene(mainInterface, width: 1000, height: 800);
primaryStage.setTitle("Google Search");
primaryStage.setScene(scene);
primaryStage.show();
}
```

- **vipButton** must be inside the **pickButton**'s scope, otherwise, vipButton cannot know which index that is user picked and required.

```
pickButton.setOnAction(ActionEvent actionEvent ->{
    ListView<String> pickView = new ListView<>();
    pickView.setPrefSize( prefWidth: 800, prefHeight: 700);
    String str = searchText.getText();
    pickIndex = Integer.parseInt(str) - 1;
    url pickUrl = urlList.get(pickIndex);
    String tempUrl = pickUrl.getWebUrl();
    tempUrl = "[Picked Original website]: " + tempUrl;
    String tempFactor = "score1: "+pickUrl.getKeywordScore() + " & score2: "+pickUrl.get
        +" & score4: "+pickUrl.getPaidScore() + " & PageRank score: "+pickUrl.getPageRan
    pickView.getItems().add(tempUrl);
    pickView.getItems().add(tempFactor);

    VBox pickvb = new VBox(searchText, searchBtn, show20webButton, insertNewButton, ge
    HBox pickhb = new HBox(pickvb, pickView);
    Scene pickScene = new Scene(pickhb, width: 1000, height: 700);
    primaryStage.setScene(pickScene);
    primaryStage.show();

vipButton.setOnAction(ActionEvent actionEvent ->{
    TreeMap<Integer, Integer>treeMap = new TreeMap<>();
    ListView<String> vipView = new ListView<>();
    vipView.setPrefSize( prefWidth: 800, prefHeight: 700);
    String paidMoney = searchText.getText();
    int increase = Integer.parseInt(paidMoney);

    Heap.buildMaxHeap(urlList);
    Heap.HeapSort(urlList);
    for(int p = 0; p < urlList.size(); p++){
        if((p+1) == pickIndex){
            Integer tempo = urlList.get(pickIndex).getPageRank() + increase;
        }
    }
}
```

- All buttons must be on Vbox, then the UI interface will show all buttons vertically.

```
VBox mainInterface = new VBox(searchText, searchBtn, show20webButton, insertNewButton, getMaxButton, pickButton,
    vipButton, searchBtn1, searchBtn2, searchBtn3, searchBtn4);
Scene scene = new Scene(mainInterface, width: 1000, height: 800);
primaryStage.setTitle("Google Search");
```

7. Leasson Learned

- While I encountered problems during the implementation, I learned to debug and create tested class for debugging:

```

import java.util.ArrayList;
public class debugHeap {
    public static void main(String[] args) {
        ArrayList<Integer> a = new ArrayList<>();
        a.add(10);
        a.add(20);
        a.add(33);
        a.add(44);
        a.add(111);
        a.add(222);
        a.add(33);
        a.add(19);
        a.add(55);
        a.add(132);
        HeapSortInteger testHeap = new HeapSortInteger(a);
        System.out.println("Original heap print:");
        System.out.println(a);
        testHeap.buildMaxHeap(a);
        System.out.println("Test build-max-heap");
        System.out.println("Expected build-max-heap: [222, 132, 111, 20, 33, 44, 55, 19, 10, 33]");
        System.out.println(a);
        System.out.println("Original heap list is: [222, 132, 111, 20, 33, 44, 55, 19, 10, 33]");
        System.out.println("Expected heap-sort: [10, 19, 20, 33, 44, 55, 111, 222, 132]");
        testHeap.HeapSort(a);
        System.out.println(a);
    }
}
```

- Then I find out the mistake: the arrayList must add a new key, then it will automatically Increasing it's size.

```

public void MaxHeapInsert(ArrayList<url> A, url key) {
    heapSize++;
    A.add(key);
    A.get(heapSize - 1).setPageRank((int) Double.NEGATIVE_INFINITY);
    heapIncreaseKey(A, heapSize, key); //then give the leaf real value
}
```

- Sometimes, professor Wu will help me to solve the problem that I have no idea:



Mike Wu

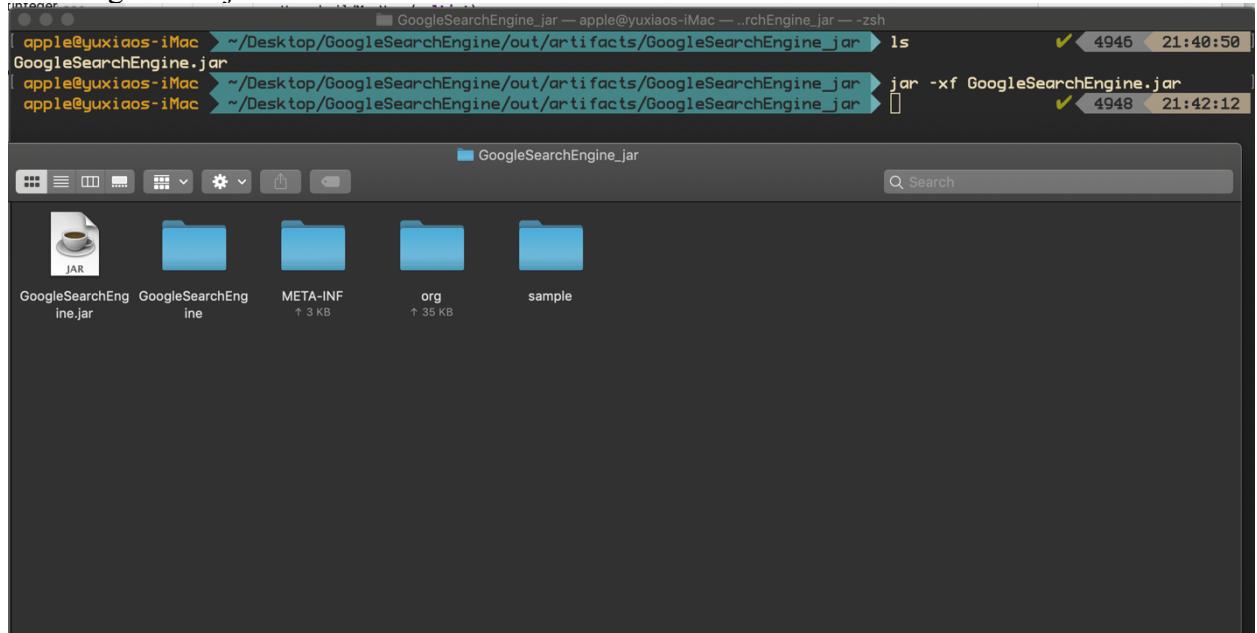
to me ▾

Allow users to pick anyone of the listed urls to increase the PageRank score and show the result.
Make sure all functional requirements are met.

...

- I learn to write a professional report in MS word fromat.

- Learning to test .jar file.

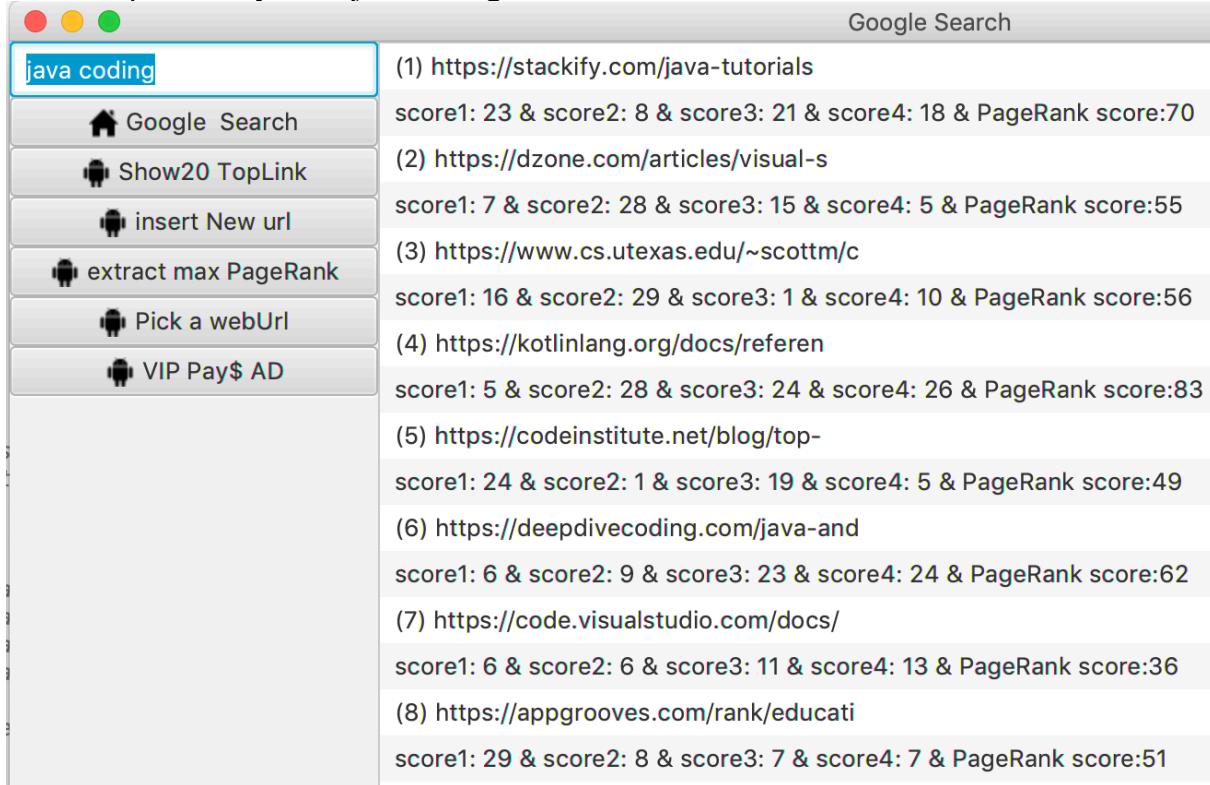


```
Last login: Sat Mar 23 21:30:02 on ttys000  ■ GoogleSearchEngine.jar — apple@yuxiaos-iMac — ..rChEngine.jar — zsh
[apple@yuxiaos-iMac ~] ~/Users/fish/Desktop/GoogleSearchEngine/out/artifacts/GoogleSearchEngine.jar ls
[apple@yuxiaos-iMac ~] ~/Desktop/GoogleSearchEngine/out/artifacts/GoogleSearchEngine.jar 1s
[apple@yuxiaos-iMac ~] ~/Desktop/GoogleSearchEngine/out/artifacts/GoogleSearchEngine.jar jar tvf GoogleSearchEngine.jar
[apple@yuxiaos-iMac ~] ~/Desktop/GoogleSearchEngine/out/artifacts/GoogleSearchEngine.jar jar tvf GoogleSearchEngine.jar
62 Sat Mar 23 21:29:40 PDT 2019 META-INF/MANIFEST.MF
0 Sat Mar 23 21:29:40 PDT 2019 GoogleSearchEngine/
293 Sat Mar 23 19:56:14 PDT 2019 GoogleSearchEngine/Controller.class
1633 Fri Mar 22 14:15:36 PDT 2019 GoogleSearchEngine/debugleap.class
1698 Sat Mar 23 21:29:40 PDT 2019 GoogleSearchEngine/extractUrl.class
297 Fri Mar 22 14:15:32 PDT 2019 GoogleSearchEngine/googleSearchEngine.fxml
3211 Sat Mar 23 19:56:14 PDT 2019 GoogleSearchEngine/heap.class
3198 Fri Mar 22 14:15:36 PDT 2019 GoogleSearchEngine/HeapSortInteger.class
12102 Sat Mar 23 21:29:40 PDT 2019 GoogleSearchEngine/Main.class
1862 Sat Mar 23 21:29:40 PDT 2019 GoogleSearchEngine/url.class
4128 Sat Mar 23 19:56:14 PDT 2019 GoogleSearchEngine/WebCrawler.class
0 Sat Mar 23 21:29:40 PDT 2019 META-INF/
16 Fri Mar 22 23:51:38 PDT 2019 META-INF/GoogleSearchEngine.kotlin_module
0 Sat Mar 23 21:29:40 PDT 2019 sample/
269 Fri Mar 22 14:15:36 PDT 2019 sample/Controller.class
12080 Fri Mar 22 14:15:36 PDT 2019 sample/Main.class
291 Fri Mar 22 14:15:32 PDT 2019 sample/sample.fxml
0 Sat Mar 23 21:29:40 PDT 2019 org/
0 Sat Mar 23 21:29:40 PDT 2019 org/jsoup/
0 Sat Mar 23 21:29:40 PDT 2019 org/jsoup/nodes/
0 Sat Mar 23 21:29:40 PDT 2019 org/jsoup/internal/
0 Sat Mar 23 21:29:40 PDT 2019 org/jsoup/parser/
0 Sat Mar 23 21:29:40 PDT 2019 org/jsoup/safety/
0 Sat Mar 23 21:29:40 PDT 2019 org/jsoup/select/
0 Sat Mar 23 21:29:40 PDT 2019 org/jsoup/helper/
0 Sat Mar 23 21:29:40 PDT 2019 META-INF/maven/
0 Sat Mar 23 21:29:40 PDT 2019 META-INF/maven/org.jsoup/
0 Sat Mar 23 21:29:40 PDT 2019 META-INF/maven/org.jsoup/jsoup/
753 Tue Mar 19 18:57:44 PDT 2019 org.jsoup.SerializationException.class
1757 Tue Mar 19 18:57:44 PDT 2019 org.jsoup.nodes.Element$1.class
9257 Tue Mar 19 18:57:44 PDT 2019 org.jsoup.nodes/Attributes.class
1616 Tue Mar 19 18:57:44 PDT 2019 org.jsoup.nodes/Node$OuterHtmlVisitor.class
1424 Tue Mar 19 18:57:44 PDT 2019 org.jsoup.nodes/ODataNode.class
1878 Tue Mar 19 18:57:44 PDT 2019 org.jsoup.nodes/PseudoTextElement.class
4228 Tue Mar 19 18:57:44 PDT 2019 org.jsoup.nodes/Document$OutputSettings.class
3865 Tue Mar 19 18:57:44 PDT 2019 org.jsoup.nodes/DocumentType.class
2663 Tue Mar 19 18:57:44 PDT 2019 org.jsoup.nodes/DataNode.class
1580 Tue Mar 19 18:57:44 PDT 2019 org.jsoup.Connection$Method.class
3034 Tue Mar 19 18:57:44 PDT 2019 org.jsoup.internal/ConstrainableInputStream.class
759 Tue Mar 19 18:57:44 PDT 2019 org.jsoup.internal/Normalizer.class
996 Tue Mar 19 18:57:44 PDT 2019 org.jsoup.parser/TokenizerState$15.class
1954 Tue Mar 19 18:57:44 PDT 2019 org.jsoup.parser/HtmlTreeBuilderState$17.class
1012 Tue Mar 19 18:57:44 PDT 2019 org.jsoup.parser/TokenizerState$33.class
1637 Tue Mar 19 18:57:44 PDT 2019 org.jsoup.parser/TokenizerState$56.class
1179 Tue Mar 19 18:57:44 PDT 2019 org.jsoup.parser/ParseError.class
197 Tue Mar 19 18:57:44 PDT 2019 org.jsoup.parser/Token$1.class
3204 Tue Mar 19 18:57:44 PDT 2019 org.jsoup.parser/HtmlTreeBuilderState$14.class
1613 Tue Mar 19 18:57:44 PDT 2019 org.jsoup.parser	TokenNameStartTag.class
1542 Tue Mar 19 18:57:44 PDT 2019 org.jsoup.parser/TokenizerState$29.class
4064 Tue Mar 19 18:57:44 PDT 2019 org.jsoup.parser/HtmlTreeBuilderState$16.class
```

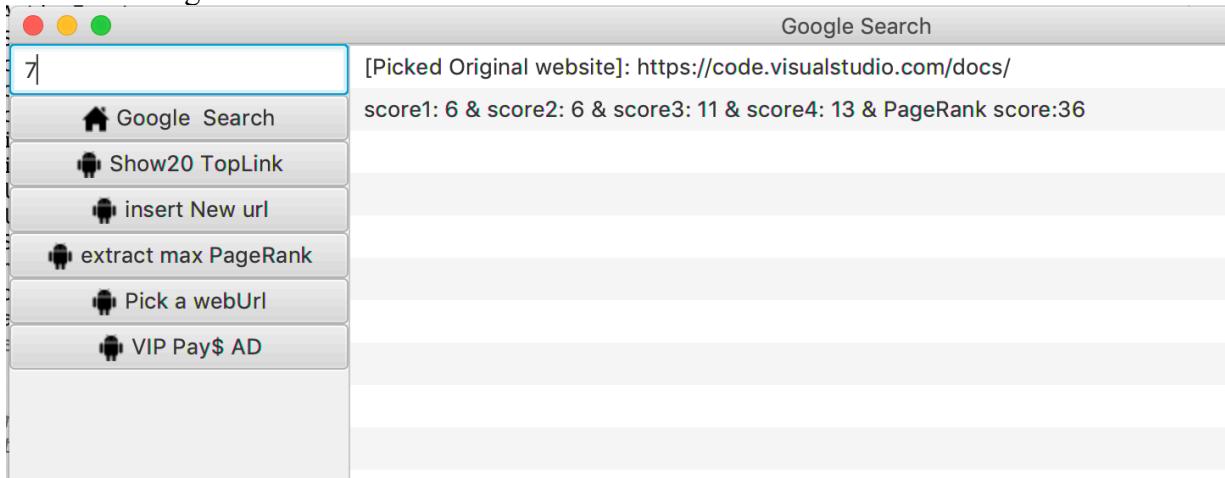
8. Updating (Debugging Report)

During last version, after increasing the score that the user picked, I forgot to resort the web url. This time I updating this new edition:

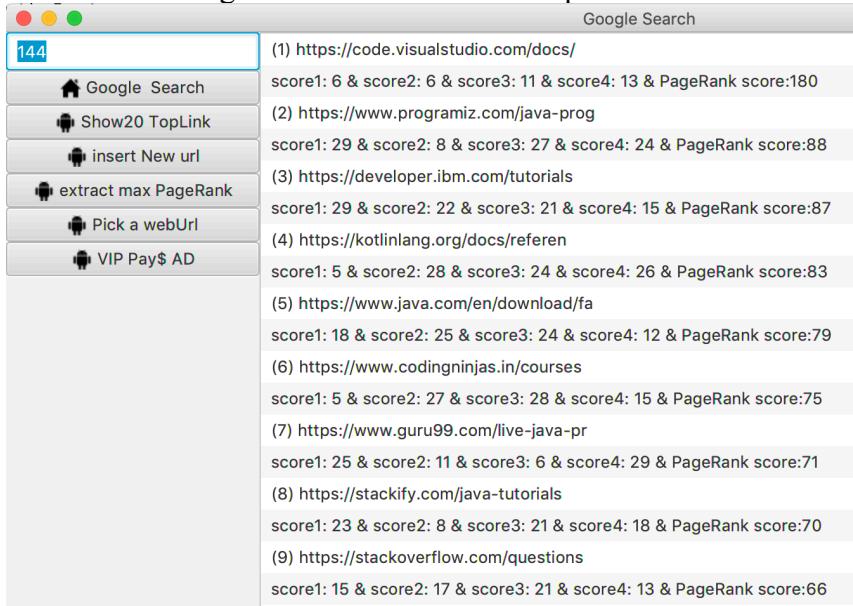
- Input the keyword: java coding



- Picking the 7th index



- Increasing the score that the user required.



- The below pictures are how to implement the requirement:

```

public url(String s){
    webUrl = s;
    linkScore = (int)(Math.random()* 30);
    keywordScore = (int)(Math.random()* 30);
    existScore = (int)(Math.random()* 30);
    score4 = (int)(Math.random()* 30);
    paidScore = 0;
}

public url(String theUrl, int s1, int s2, int s3, int s4, int s5){
    webUrl = theUrl;
    linkScore = s1;
    keywordScore = s2;
    existScore = s3;
    score4 = s4;
    paidScore = s5;
}

int i = 1;
for(int v = urlList.size()-1; v>=0; v--){
    String tempWeb = "(" + i + ")" + urlList.get(v).getWebUrl();
    String vipFactor = "score1: "+urlList.get(v).getKeywordScore()+" & score2: "+urlList.get(v).getExistScore()+" & score3: "+urlList.get(v).getLinkScore()+" & score4: "+urlList.get(v).getScore4()+" & PageRank score: "+urlList.get(v).getPageRank();
    vipView.getItems().add(tempWeb);
    vipView.getItems().add(vipFactor);
    i++;
}

VBox vipvb = new VBox(searchText, searchBtn, show20webButton, insertNewButton, getMaxButton, pickButton, vipButton);
HBox viphb = new HBox(vipvb, vipView);
Scene vipScene = new Scene(viphb, width: 1000, height: 700);
primaryStage.setScene(vipScene);
primaryStage.show();
...

```

Here, adding two constructors in order to implement the webUrl's resorted order.

This for loop will help resort the web Url