



DUDE, Where's My Spot?

By Jovanny Perez, Anthony Barrios, Karissa Schick, Areli Muñoz, Matthew Cowans



Topics Being Covered

- Selection and Bubble Sort
- Hashing
- Keep Me Logged In feature
- Binary Searching
- Weather Api
- Marquee Event Scroller

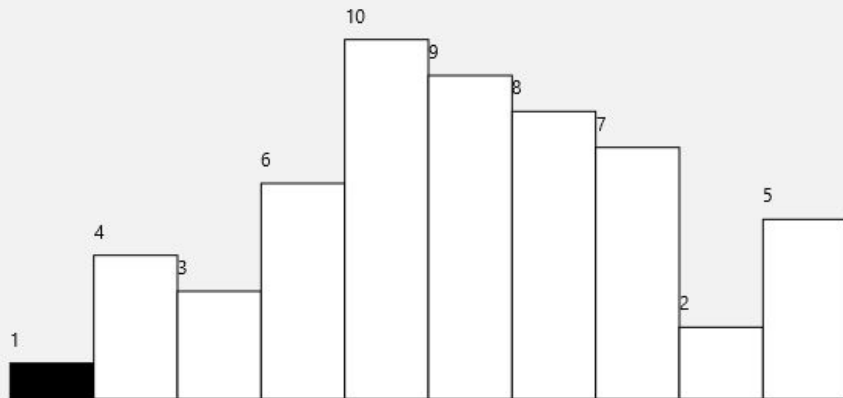
Selection Sort

```
// figuring out the next numbers being swapped in selection sort
public void nextStep() {
    passNumber++;
    if (passNumber >= searchSize) { // end of sorting
        sortStepBtn.setDisable(true);
        passNumber = -1;
    } else {
        int currentMin = random.get(passNumber); // current number and
        int currentMinIndex = passNumber;

        for (int j = passNumber + 1; j < random.size(); j++) {
            if (currentMin > random.get(j)) { // looking for a smaller num
                currentMin = random.get(j);
                currentMinIndex = j;
            }
        }
        if (currentMinIndex != passNumber) { // if the index of the sm
            random.set(currentMinIndex, random.get(passNumber)); // swap
            random.set(passNumber, currentMin);
        }
    }
}
```

Selection Sort

The selection sort algorithm sorts an array by repeatedly finding the minimum element from unsorted part and putting it at the beginning. In every iteration of selection sort, the minimum element (considering ascending order) from the unsorted subarray is picked and moved to the sorted subarray. Try it out for yourself below!



Start

Next Step

Reset

Bubble Sort

```
passNumber++;  
}  
}  
if (passNumber >= searchSize) {  
    bblStepBtn.setDisable(true);  
    passNumber = -1;  
}  
else {  
    if (endOfPass) {  
        needNextPass = false;  
        endOfPass = false;  
        index = 0;  
    }  
    if (!endOfPass) {  
        if (random.get(index) > random.get(index + 1)) {  
            // Swap list[i] with list[i + 1]  
            int temp = random.get(index);  
            random.set(index, random.get(index + 1));  
            random.set(index + 1, temp);  
            needNextPass = true; // Next pass still needed  
        }  
        index++;  
        if (!(index < random.size() - passNumber)) {  
            endOfPass = true;  
        }  
    }  
    if (endOfPass) {  
        if (!needNextPass) {  
            bblStepBtn.setDisable(true);  
            passNumber = -1;  
            index = -1;  
        }  
    }  
}
```

Bubble Sort

Bubble sort, sometimes referred to as sinking sort, is a simple sorting algorithm that repeatedly steps through the list to be sorted, compares each pair of adjacent items and swaps them if they are in the wrong order. This is repeated until no swaps are needed. Try it out for yourself below!



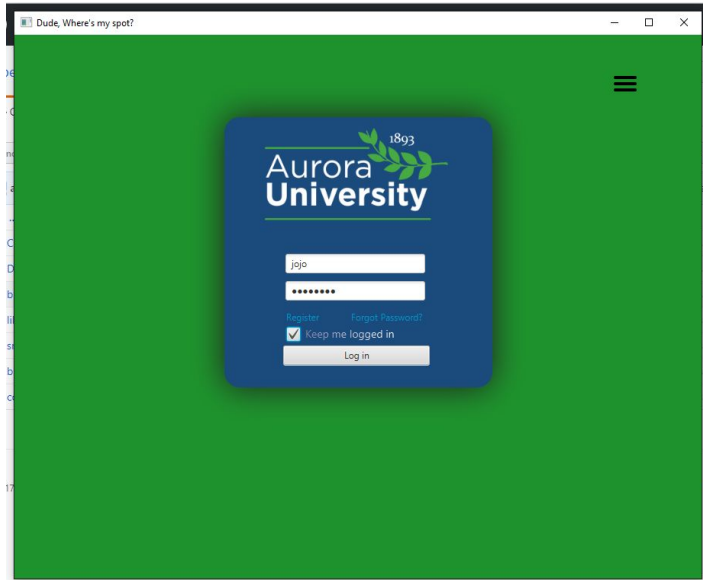
Hashing

```
//get text from field
String toHashString = HashFld.getText();
//Create instance of the hashing class using sha-256 algorithm
MessageDigest messageDigest = MessageDigest.getInstance("SHA-256");
//hash string
messageDigest.update(toHashString.getBytes());
//Convert to hashed string bytes to String
String encryptedString = new String(messageDigest.digest());
//Display hashed String
showHashedStringTxtArea.setText(encryptedString);
```

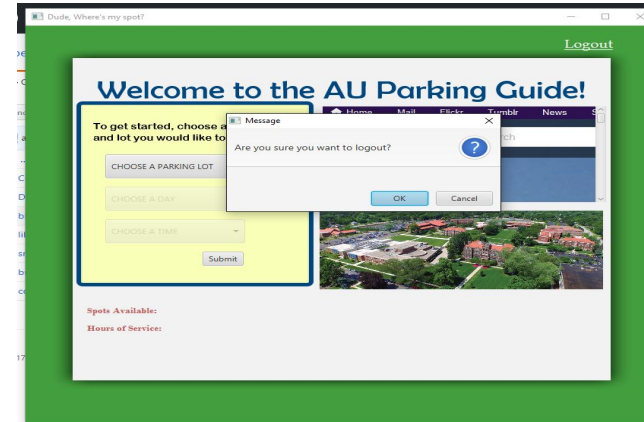
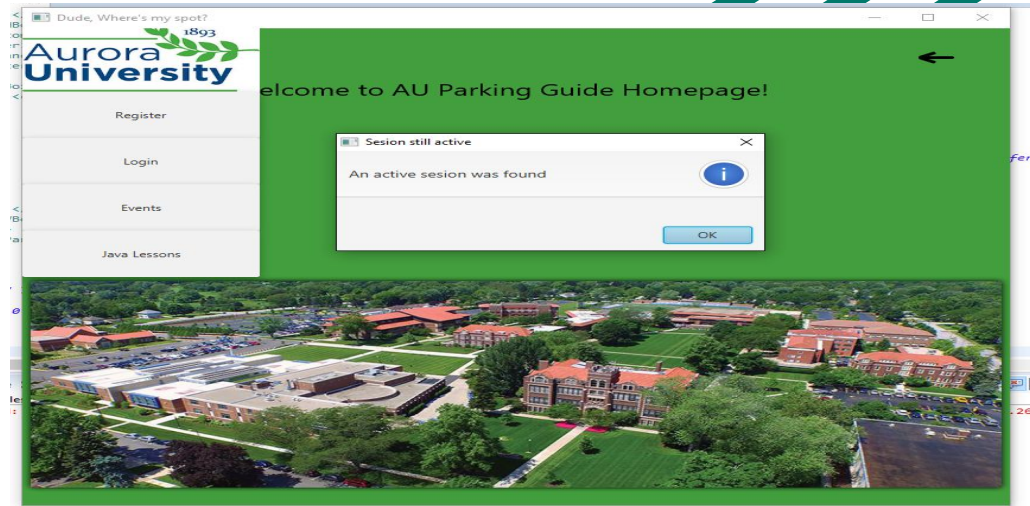
Enter a string to be hashed

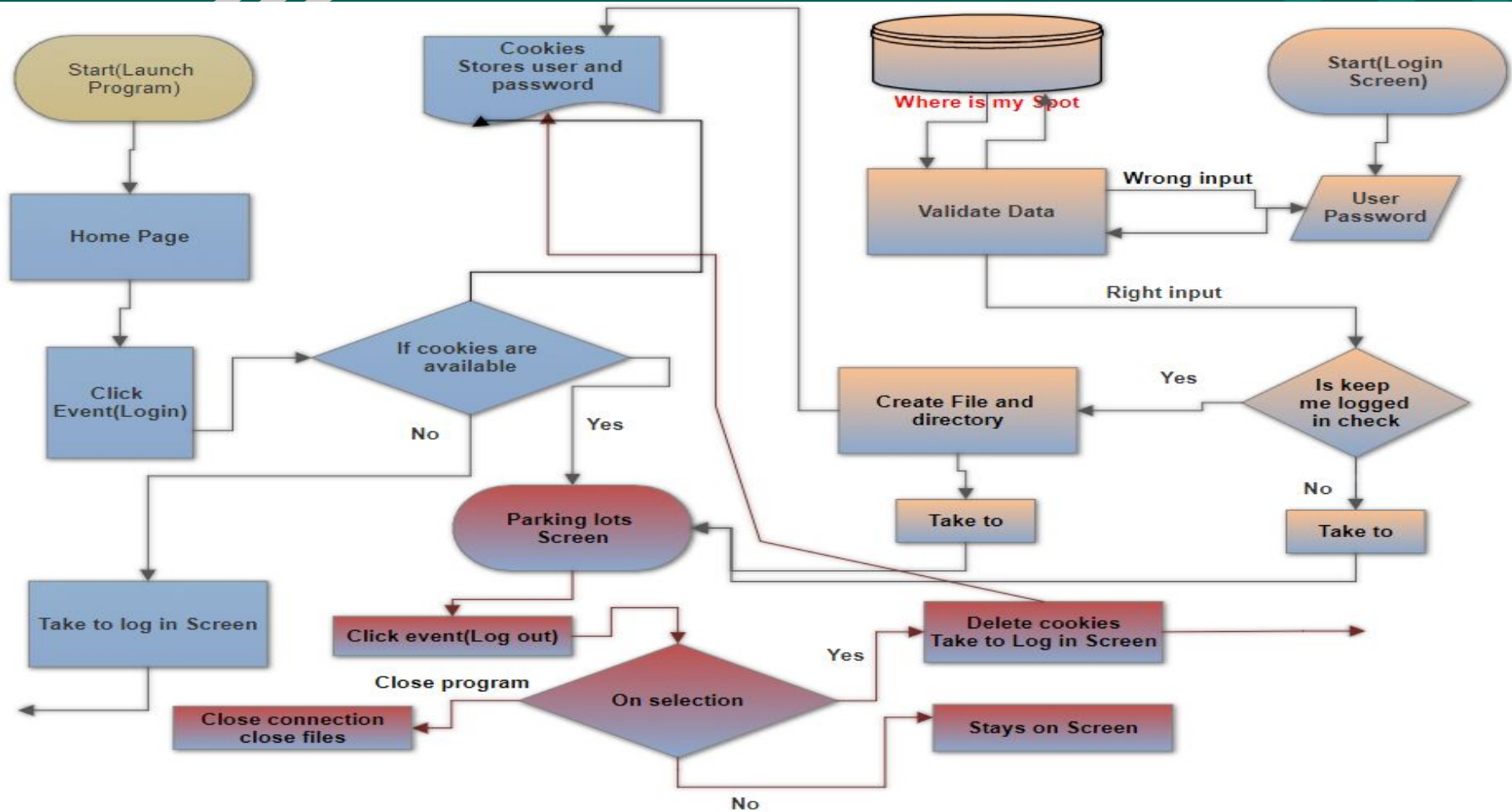
Your Hashed String

Keep Me Logged In



A screenshot of a web browser window titled "Dude, Where's my spot?". The page has a green background. In the center is a dark blue login box with the Aurora University logo (1893) at the top. Below the logo are two input fields: the first contains "jojo" and the second contains "*****". To the left of the password field are links for "Register" and "Forgot Password?". Below the password field is a checkbox labeled "Keep me logged in" which is checked. At the bottom of the box is a "Log in" button.





Binary Search

Front End

```
//When the run button is pushed
run.setOnAction( run->{
    int result;

    //Gather text from the textField
    String in = input.getText();
    int searchValue = Integer.parseInt(in);
    textArea.appendText("\n\nYou entered the number: " + in);

    //Display the current time in milliseconds when sorting started
    begin_time = System.currentTimeMillis();
    textArea.appendText("\nTime sorting started: " + begin_time);

    //Search for the value
    result = BinarySearch.search(numbers, searchValue);

    //Display the results
    if (result == -1)
    {
        //If the result is -1 then the value was not found
        textArea.appendText("\n" + searchValue + " value cannot be found");
        textArea.appendText("\nTime sorting ended: " + stop_time);
        textArea.appendText("\nSelection Sort Took " + (stop_time - begin_time));
    } else {
        //If the result is found then display what element it was found at
        textArea.appendText("\n" + searchValue + " was found at element " + result);
        stop_time = System.currentTimeMillis();
        textArea.appendText("\nTime sorting ended: " + stop_time);
        textArea.appendText("\nSelection Sort Took " + (stop_time - begin_time));
    }
});

//If reset button is pushed...
reset.setOnAction(reset-> {
    //Text area and input is cleared. Then the array is displayed again
    textArea.clear();
    input.clear();
    textArea.appendText(Arrays.toString(numbers));
});
```

- Use an interval that covers the whole array to determine the midpoint
- Starts by searching a sorted array by repeatedly dividing the interval in half
- If the search value is less than the value of the middle index item, narrow the interval to the lower half of the array
- If opposite, narrow it to the upper half
- Continue checking until the value is found in the interval, or the array does not contain the value searched

Binary Search

Back End

- The value entered is passed as a parameter to the function 'search', along with the array
- Perform a while loop to constantly search the array until the value is found or not found
- Calculates the midpoint of each half
- If the value equals the midpoint value, the variable 'found' becomes 'true' and exits the loop
- If the value is in the lower half, reduce the interval to the lower half only
- If the value is in the upper half, increase the interval to the upper half only
- If the value was not found, return the position as -1

```
//Set the initial values
first = 0;
last = array.length - 1;
position = -1;
found = false;

//Search for the value
while(!found && first <= last)
{
    //Calculate mid point
    middle = (first + last) / 2;

    //If value is found at midpoint...
    if(array[middle] == value)
    {
        found = true;
        position = middle;
    }
    //else if value is in lower half...
    else if (array[middle] > value)
        last = middle - 1;
    //else if value is in upper half...
    else first = middle + 1;
}

//Return the position of the item, or -1
//if it was not found
return position;
```

Why We Will Not Use Binary Search

In our project, we will not be using binary search. We have decided to use a database to store data and can search/pull from the database using queries.

Binary Search

Binary search is the other common search approach for a list of values. For binary search to work, the elements in the array must already be ordered. Assume that the array is in ascending order. The binary search first compares the key with the element in the middle of the array. If the value of the search key is less than the element in the middle of the array, narrow the array to the lower half. Otherwise, narrow it to the upper half. Repeatedly check until the value is found or the array is empty.

Enter a value to search for:

Run

Results:

[536, 289, 296, 429, 319, 142, 349, 101, 388, 147, 417, 199, 207, 222, 189, 30, 447, 521, 234, 600]
[30, 101, 142, 147, 189, 199, 207, 222, 234, 289, 296, 319, 349, 388, 417, 429, 447, 521, 536, 600]

Reset

Weather API

Shows the temperature in
fahrenheit

Shows the mode

<http://api.openweathermap.org/data/2.5/weather?q=<ENTER CITY>&units=imperial&mode=html&appid=<ENTER API KEY>>

Enter your city:
North Aurora, us

- Have your api key
- Create an html container
- Use <div> to insert weather information

```
@FXML
private WebView weather;
@FXML
private ImageView lotImage;
@FXML
private Label spotsAv;
@FXML
private Labeled hoursAv;

public void initialize() throws Exception {
    WebEngine webEngine = weather.getEngine();
    webEngine.load("http://api.openweathermap.org/data/2.5/weather?q=North%20Aurora,%20us&units=imperial&mode=html&appid=YOUR_API_KEY");
}
```

Weather API

<http://api.openweathermap.org/data/2.5/weather?q=North%20Aurora,%20us&units=imperial&mode=html&appid=1906951238f29646ce8ef804bea0b524>

```
<html lang="en">
<head>
<meta charset="utf-8">
<meta name="keywords" content="weather, world, openweathermap, weather, layer" />
<meta name="description" content="A layer with current weather conditions in cities for
world wide" />
<meta name="domain" content="openweathermap.org" />
<meta http-equiv="pragma" content="no-cache" />
<meta http-equiv="Expires" content="-1" />
</head>
<body>
<div style="font-size: medium; font-weight: bold; margin-bottom: 0px;">North Aurora</div>
<div style="float: left; width: 130px;">
<div style="display: block; clear: left;">
<div style="float: left;" title="Titel">

</div>
```

North Aurora
43.88°F



Clouds: 1%
Humidity: 45%
Wind: 5.73 m/h
Pressure: 1026hpa
[More..](#)

```
<div style="float: left;">
```

```
<div style="display: block; clear: left; font-size: medium; font-weight: bold; padding: 0pt 3pt,"  
title="Current Temperature">43.88°F</div>
```

```
<div style="display: block; width: 85px; overflow: visible;"></div>
```

```
</div>
```

```
</div>
```

```
<div style="display: block; clear: left; font-size: small;">Clouds: 1%</div>
```

```
<div style="display: block; clear: left; color: gray; font-size: x-small;" >Humidity: 45%</div>
```

```
<div style="display: block; clear: left; color: gray; font-size: x-small;" >Wind: 5.73 m/h</div>
```

```
<div style="display: block; clear: left; color: gray; font-size: x-small;" >Pressure: 1026hpa</div>
```

```
</div>
```

```
<div style="display: block; clear: left; color: gray; font-size: x-small;">
```

```
<a
```

```
href="http://openweathermap.org/city/4903818?utm\_source=openweathermap&utm\_medium=widge  
t&utm\_campaign=html\_old" target="_blank">More..</a>
```

```
</div>
```

```
<script>(function(i,s,o,g,r,a,m){i['GoogleAnalyticsObject']=r;i[r]=i[r]||function(){  
(i[r].q=i[r].q||[]).push(arguments)},i[r].l=1*new Date();a=s.createElement(o),  
m=s.getElementsByTagName(o)[0];a.async=1;a.src=g;m.parentNode.insertBefore(a,m)  
})(window,document,'script','/www.google-analytics.com/analytics.js','ga');ga('create',  
'UA-31601618-9', 'auto');ga('send', 'pageview');</script>
```

```
</body>
```

```
</html>
```

North Aurora
43.88°F



Clouds: 1%

Humidity: 45%

Wind: 5.73 m/h

Pressure: 1026hpa

[More..](#)

Jsoup and JavaFx Marquee

```
public String loop(String outPut) throws IOException {
```

```
    final Document document =
```

```
    Jsoup.connect("https://aurora.edu/academics/resources/academic-calendar.html#.WfjplNvMwYI").get();
```

```
    StringBuilder sb = new StringBuilder();
```

```
    for (Element row : document.select("table tr")){
```

```
        Elements title = row.select("td");
```

```
        if (title.size() == 2) {
```

```
            outPut = (title.get(1).text() + " " + title.get(0).text()+"\n");
```

```
            outPut = sb.append(outPut).toString();
```

```
        }
```

```
    }
```

```
    return outPut;
```

```
}
```

```
<table border="1" cellpadding="4" cellspacing="1"
summary="2017-2018 Academic Calendar">
```

```
<tbody>
```

```
<tr>
```

```
<td width="63%">
```

```
<p>Opening Week - Faculty Orientation/Meetings</p>
```

```
</td>
```

```
<td width="37%">
```

```
<p>August 21-25</p>
```

```
</td>
```

```
</tr>
```

```
<tr>
```

```
<td width="63%">
```

```
<p>New Student Orientation</p>
```

```
</td>
```

```
<td width="37%">
```

```
<p>August 24-27</p>
```

```
</td>
```

```
</tr>
```

```
<tr></tr>
```

```
<tr style="background-color: #cccccc;">
```

```
<td width="63%">
```

```
<p><strong>Fall Semester Classes Begin</strong></p>
```

```
</td>
```

```
<td width="37%">
```

```
<p><strong>August 28<br/></strong></p>
```

```
</td>
```

```
</tr>
```

```
txt.setText(loop(outPut));  
    double msgWidth = txt.getLayoutBounds().getWidth();  
  
    KeyValue initKeyValue = new KeyValue(txt.translateXProperty(),  
200);  
    KeyFrame initFrame = new KeyFrame(Duration.ZERO, initKeyValue);  
  
    KeyValue endKeyValue = new KeyValue(txt.translateXProperty(), -1.0  
    * msgWidth);  
    KeyFrame endFrame = new KeyFrame(Duration.seconds(3),  
endKeyValue);  
  
    Timeline timeline = new Timeline(initFrame, endFrame);  
  
    timeline.setCycleCount(Timeline.INDEFINITE);  
    timeline.play();
```

August 21-25 Opening Week - Faculty Orientation/Meetings
August 24-27 New Student Orientation
August 28 Fall Semester Classes Begin
September 2 End of Add for day classes; evening classes may be added prior to second class meeting; end of 100% refund for Fall Semester
August 28 – October 21 8-week Fall Module I
September 4 Labor Day - no classes
October 13 Founders Convocation (no traditional day classes after 1:05 p.m.)
October 7 Module I - last day to drop with automatic "W"
October 20-22 Fall Weekend - no traditional day classes
October 23 - December 16 8-Week Fall Module II
November 11 Last day to drop fall semester classes with automatic "W"
November 22-26 Thanksgiving Holiday

Problems Encountered

- >App-DB connection is not ideal(Not working)
- >App-WebService-DB is ideal

Summary

- As of now we are up to date and working very well as a team
- We are encountering very little problems
- “The sky's the limit” mentality

*Add gif image
here (haha)

What's Next?

- Demonstrate iterator and comparator
- (Possibility) → Add a tutorial video on how to use this app