CS314 Software Engineering
Sprint 3 - Web Server, SQL, 2-opt

Dave Matthews
Sprint 3 Epics

- Epics from previous Sprints
- Map background
- Web Server
  - Load application page from web server
  - Application interacts with server to get map and itinerary
  - User selects destinations from a large data source (SQL).
- 2-opt
  - apply to all nearest neighbor tours, choose the shortest.
Web server - stage 1

• Enter a word, click search button. Examples:
  – heliport
  – canyon
  – denver
• Display the map and itinerary with all destinations that match the word.
• Repeat
Web server - stage 1

Data | Server | Interface | Application
--- | --- | --- | ---
| Data | Itinerary | Interface | Application

SQL

- itinerary
- nn
- 2-opt

parseJson

createJson

search

search

render
Web server - stage 2

- Enter word, click search button.
- Add destinations from search results. Remove selected destinations.
- Plan the trip using selected destinations.
- Display the map and itinerary for the selected destinations.
- Repeat search/select/plan.
Web server - stage 2
Demo server

- http://sparkjava.com
```sql
SELECT id, name, municipality, type, latitude, longitude FROM airports LIMIT 20;
```

<table>
<thead>
<tr>
<th>id</th>
<th>name</th>
<th>municipality</th>
<th>type</th>
<th>latitude</th>
<th>longitude</th>
</tr>
</thead>
<tbody>
<tr>
<td>KBJC</td>
<td>Rocky Mountain Metropolitan Airport</td>
<td>Denver</td>
<td>medium_airport</td>
<td>39.0879822</td>
<td>-105.116996</td>
</tr>
<tr>
<td>KBKF</td>
<td>Buckley Air Force Base</td>
<td>Aurora</td>
<td>medium_airport</td>
<td>39.701698303200004</td>
<td>-104.751998901</td>
</tr>
<tr>
<td>KCOS</td>
<td>City of Colorado Springs Municipal Airport</td>
<td>Colorado Springs</td>
<td>large_airport</td>
<td>38.805801391602</td>
<td>-104.7009639893</td>
</tr>
<tr>
<td>KDEN</td>
<td>Denver International Airport</td>
<td>Denver</td>
<td>large_airport</td>
<td>38.861698150635</td>
<td>-104.672996521</td>
</tr>
<tr>
<td>KEGE</td>
<td>Eagle County Regional Airport</td>
<td>Eagle</td>
<td>medium_airport</td>
<td>39.64260101</td>
<td>-106.9179993</td>
</tr>
<tr>
<td>KJGT</td>
<td>Grand Junction Regional Airport</td>
<td>Grand Junction</td>
<td>medium_airport</td>
<td>39.1223983765</td>
<td>-108.527000427</td>
</tr>
<tr>
<td>KPUB</td>
<td>Pueblo Memorial Airport</td>
<td>Pueblo</td>
<td>small_airport</td>
<td>38.29100646972656</td>
<td>-104.4970016479422</td>
</tr>
<tr>
<td>00CO</td>
<td>Cass Field</td>
<td>Briggsdale</td>
<td>small_airport</td>
<td>40.6222001220703</td>
<td>-104.34400177001953</td>
</tr>
<tr>
<td>01CO</td>
<td>St Vincent General Hospital Heliport</td>
<td>Leadville</td>
<td>heliport</td>
<td>37.64329910279999</td>
<td>-106.24600219726562</td>
</tr>
<tr>
<td>02CO</td>
<td>Mc Cullough Airport</td>
<td>Monte Vista</td>
<td>small_airport</td>
<td>40.2125015259</td>
<td>-106.744003296</td>
</tr>
<tr>
<td>03CO</td>
<td>Kugel-Strong Airport</td>
<td>Platteville</td>
<td>small_airport</td>
<td>38.0990833</td>
<td>-104.1743889</td>
</tr>
<tr>
<td>04V</td>
<td>Saguache Municipal Airport</td>
<td>Saguache</td>
<td>small_airport</td>
<td>40.2149839</td>
<td>-104.9844228</td>
</tr>
<tr>
<td>05CO</td>
<td>Rancho De Aereo Airport</td>
<td>Mead</td>
<td>small_airport</td>
<td>37.41109848022461</td>
<td>-105.552001953125</td>
</tr>
<tr>
<td>05V</td>
<td>Blanca Airport</td>
<td>Blanca</td>
<td>small_airport</td>
<td>37.38750076293945</td>
<td>-103.69100189208984</td>
</tr>
<tr>
<td>06CO</td>
<td>Jecan Airport</td>
<td>Branson</td>
<td>small_airport</td>
<td>39.2635993575195</td>
<td>-104.427001953125</td>
</tr>
<tr>
<td>07CO</td>
<td>Comanche Creek Airport</td>
<td>Kiowa</td>
<td>small_airport</td>
<td>38.73249816894531</td>
<td>-104.84100036621094</td>
</tr>
<tr>
<td>08CO</td>
<td>Terra Firma Airport</td>
<td>Rush</td>
<td>small_airport</td>
<td>38.055599212646484</td>
<td>-103.65299987792969</td>
</tr>
<tr>
<td>09CO</td>
<td>Cottonwood Field</td>
<td>Swink</td>
<td>small_airport</td>
<td>38.7407989502</td>
<td>-108.652001953</td>
</tr>
<tr>
<td>0CD0</td>
<td>Delta County Memorial Hospital Heliport</td>
<td>Delta</td>
<td>heliport</td>
<td>40.2610917</td>
<td>-103.7963389</td>
</tr>
<tr>
<td>0CD1</td>
<td>Colorado Plains Medical Center Heliport</td>
<td>Fort Morgan</td>
<td>heliport</td>
<td>40.2610917</td>
<td>-103.7963389</td>
</tr>
</tbody>
</table>

20 rows in set (0.00 sec)
SQL (https://mariadb.org/learn/)

# connect to the database from a shell using your eID
mysql -u eID -D cs314 -h faure -p
# once in SQL...
# see a list of tables
show tables;
# see the columns in a table
show columns from airports;
# count the number of records in the table
select count(*) from airports;
# see the first 5 entries in the airports table
select * from airports limit 5;
# see selected columns
select id,name,municipality from airports limit 5;
# show the heliports
select name from airports where type = 'heliport';

# show all of the airports (large, medium, small)
select name from airports where type like '%airport%';

# show all records that refer to denver sorted by name
select id, name, municipality, type from airports where name like '%denver%' or municipality like '%denver%' order by name;

# select airports by ids
select id, name, municipality, type from airports where id in ('19CO', '26CO', '77CO', 'C023', 'C024', 'K00V', 'KFN'L', 'KDEN');

# finished
quit;
import java.sql.Connection; // https://docs.oracle.com/javase/tutorial/jdbc/basics/index.html
import java.sql.DriverManager; // https://www.tutorialspoint.com/jdbc/
import java.sql.SQLException;
import java.sql.Statement;
import java.sql.ResultSet;

public class Sprint3Example1 {
    public static void main(String[] args) { // command line args contain username and password
        String myDriver = "com.mysql.jdbc.Driver"; // add dependencies in pom.xml
        String myUrl = "jdbc:mysql://faure.cs.colostate.edu/cs314";
        try { // connect to the database
            Class.forName(myDriver);
            Connection conn = DriverManager.getConnection(myUrl, args[0], args[1]);
            try { // create a statement
                Statement st = conn.createStatement();
                try { // submit a query
                    String query = "SELECT * FROM airports LIMIT 10";
                    ResultSet rs = st.executeQuery(query);
                    try { // iterate through the query results and print selected columns
                        while (rs.next()) {
                            String id = rs.getString("id");
                            String name = rs.getString("name");
                            System.out.printf("%s,%s\n", id, name);
                        }
                    } finally { rs.close(); }
                } finally { st.close(); }
            } finally { conn.close(); }
        } catch (Exception e) { // catches all exceptions in the nested try's
            System.err.printf("Exception: ");
            System.err.println(e.getMessage());
        }
    }
}
2-opt
2-opt inversions

route  23  15  3  9  7  ...  21  11  5  4  23
index  0  1  2  3  ...  n-3  n-2  n-1  n

\[0 \leq i < i+1 < k < k+1 \leq n\]
2-opt (from Wikipedia) - very slow

2optSwap(route, i, k) {
  1. take route[1] to route[i-1] and add them in order to new_route
  2. take route[i] to route[k] and add them in reverse order to new_route
  3. take route[k+1] to end and add them in order to new_route
  return new_route;
}

repeat until no improvement is made {
  start_again:
  best_distance = calculateTotalDistance(existing_route)
  for (i = 0; i < number of nodes eligible to be swapped - 1; i++) {
    for (k = i + 1; k < number of nodes eligible to be swapped; k++) {
      new_route = 2optSwap(existing_route, i, k)
      new_distance = calculateTotalDistance(new_route)
      if (new_distance < best_distance) {
        existing_route = new_route
        goto start_again
      }
    }
  }
}
2-opt (my version)

2optSwap(route, i1, k) { // swap in place
    while(i1 < k) {
        temp = route[i1]
        route[i1] = route[k]
        route[k] = temp
        i1++; k--
    }
}

improvement = true
while improvement {
    improvement = false
    for (i = 0; i <= n-3; i++) { // check n>4
        for (k = i + 2; k <= n-1; k++) {
            delta = -dis(route,i,i+1)-dis(route,k,k+1)+dis(route,i,k)+dis(route,i+1,k+1)
            if (delta < 0) { //improvement?
                2optSwap(route, i+1, k)
                improvement = true
            }
        }
    }
}