Hoang Vu – Porcelain Dolls Delivery

(616) 928-5794 – hoangvu8294@gmail.com

Objective

I am a nice old lady who delivers porcelain dolls on foot using my walker to residents in a neighborhood which does not sound spooky at all. Every day a nice young man who might be working for CIA or Satan, fills my handbag with porcelain dolls. He gives me a map of the neighborhood and an address of the neighbor to deliver the handbag of dolls to. The handbag is extremely heavy and I have a bad hip so I always try to minimize the walking distance from my beginning position to my destination. For most part, my job is to determine the shortest path to take to deliver my handbag of dolls. Great.

The most important task is writing a function in the Kotlin programming language when given:

- a starting location
- a target location
- a list of edges where each edge is represented as a map and connects two locations

Produces the shortest path with distance which:

- starts at the given starting location
- ends at the given target location

The function has a signature like this:

```
fun dijkstra(startLocation: String, endLocation: String, edges: List<Map<String, Any>>): Map<String, Any>
```

where an `edge` is a `Map` with keys for `startLocation`, `endLocation`, and `distance`. The return value should be a `Map` with keys for `distance` and `path`. I have included a set of executable high-level tests for my solution along with a sample GUI app for testing multiple cases and mostly for enjoyment while testing for client (you).

Technologies

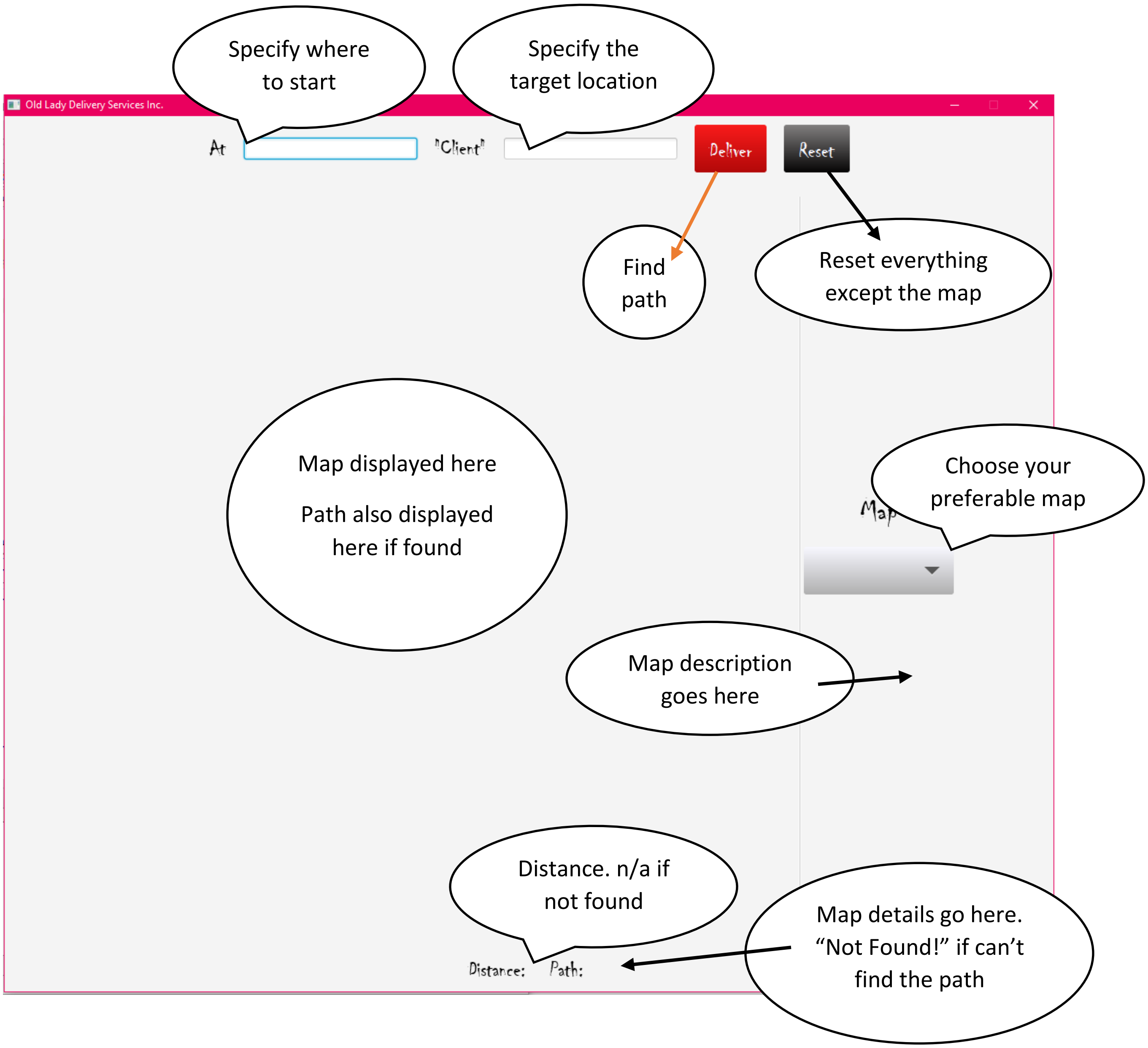
- IDE: IntelliJ IDEA 2016.3.5
- Language: **Kotlin**
- Libraries:
  - ➢ JavaFX lib : GUI design and implementation
  - ➢ TestNG : Unit testing
  - ➢ KotlinJavaRuntime: .jar file artifacts building
  - ➢ JDK 1.8

Design

doll-delivery-Kotlin

Package	Classes	Function
dd_gui	alert	Used as an alert box
	dd_app	Main class that handles all inputs from user
dd_map	map_painter	Provides methods that draw the map and the path
	map_processor	Loads in map for user input. Currently, user can only pick between two maps
dd_qa	original_map	Tests the map that is provided through GitHub
	small_map	Tests a made up small map

dd_app design



Specify where to start

Specify the target location

Find path

Reset everything except the map

Map displayed here

Path also displayed here if found

Choose your preferable map

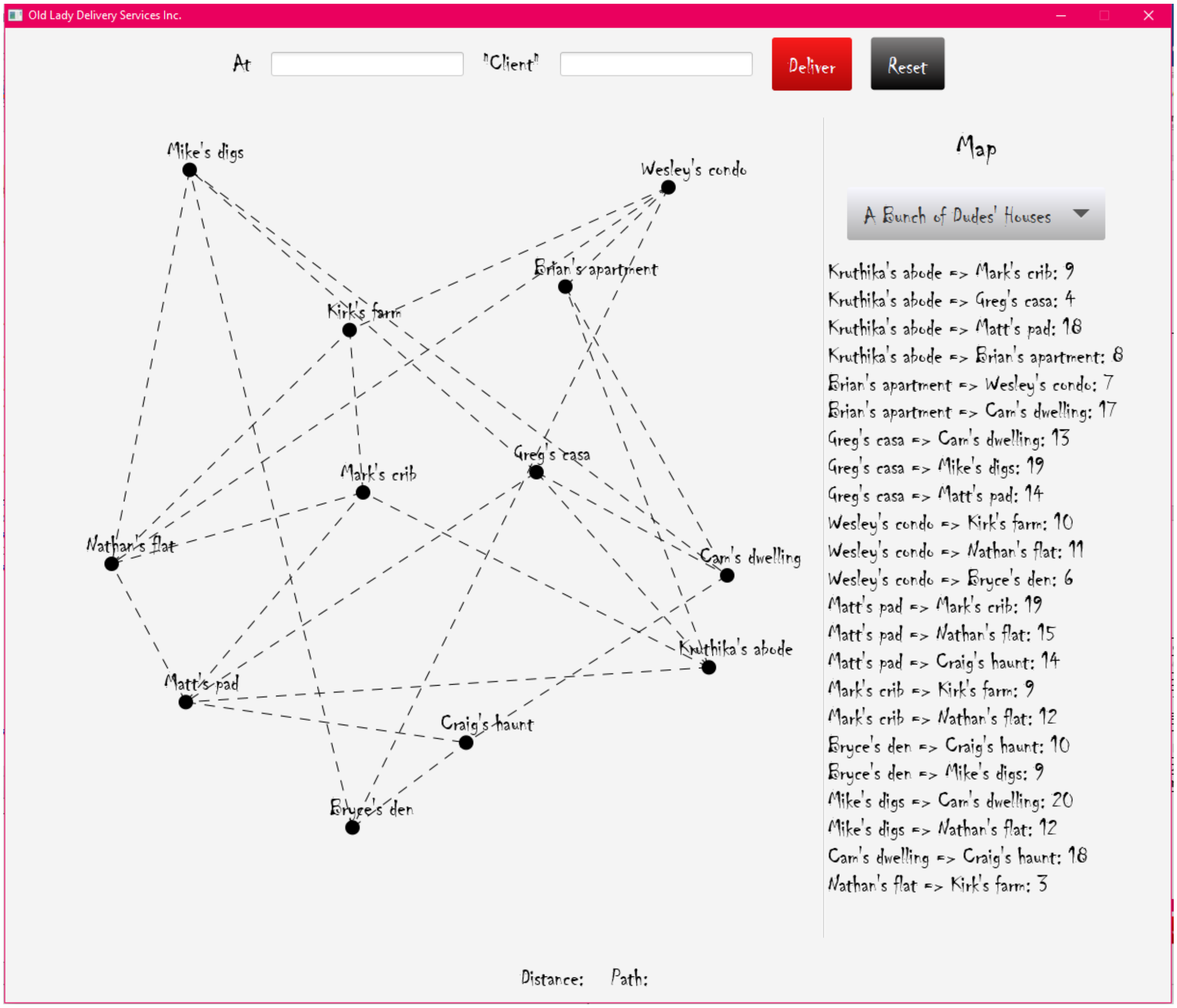
Map description goes here

Distance. n/a if not found

Map details go here. “Not Found!” if can’t find the path

Map preview

locations are generated randomly but guarantee the well distanced between locations.

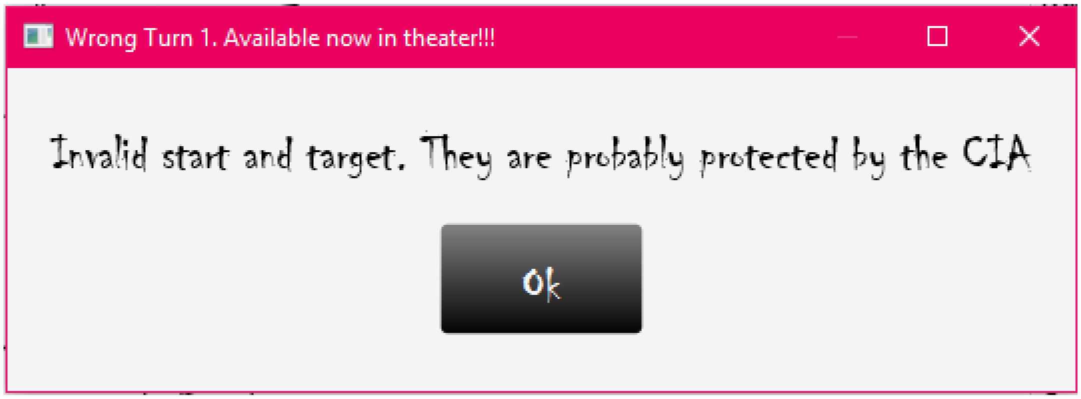


Kruthika's abode => Mark's crib: 9
Kruthika's abode => Greg's casa: 4
Kruthika's abode => Matt's pad: 18
Kruthika's abode => Brian's apartment: 8
Brian's apartment => Wesley's condo: 7
Brian's apartment => Cam's dwelling: 17
Greg's casa => Cam's dwelling: 13
Greg's casa => Mike's digs: 19
Greg's casa => Matt's pad: 14
Wesley's condo => Kirk's farm: 10
Wesley's condo => Nathan's flat: 11
Wesley's condo => Bryce's den: 6
Matt's pad => Mark's crib: 19
Matt's pad => Nathan's flat: 15
Matt's pad => Craig's haunt: 14
Mark's crib => Kirk's farm: 9
Mark's crib => Nathan's flat: 12
Bryce's den => Craig's haunt: 10
Bryce's den => Mike's digs: 9
Mike's digs => Cam's dwelling: 20
Mike's digs => Nathan's flat: 12
Cam's dwelling => Craig's haunt: 18
Nathan's flat => Kirk's farm: 3

Distance: Path:

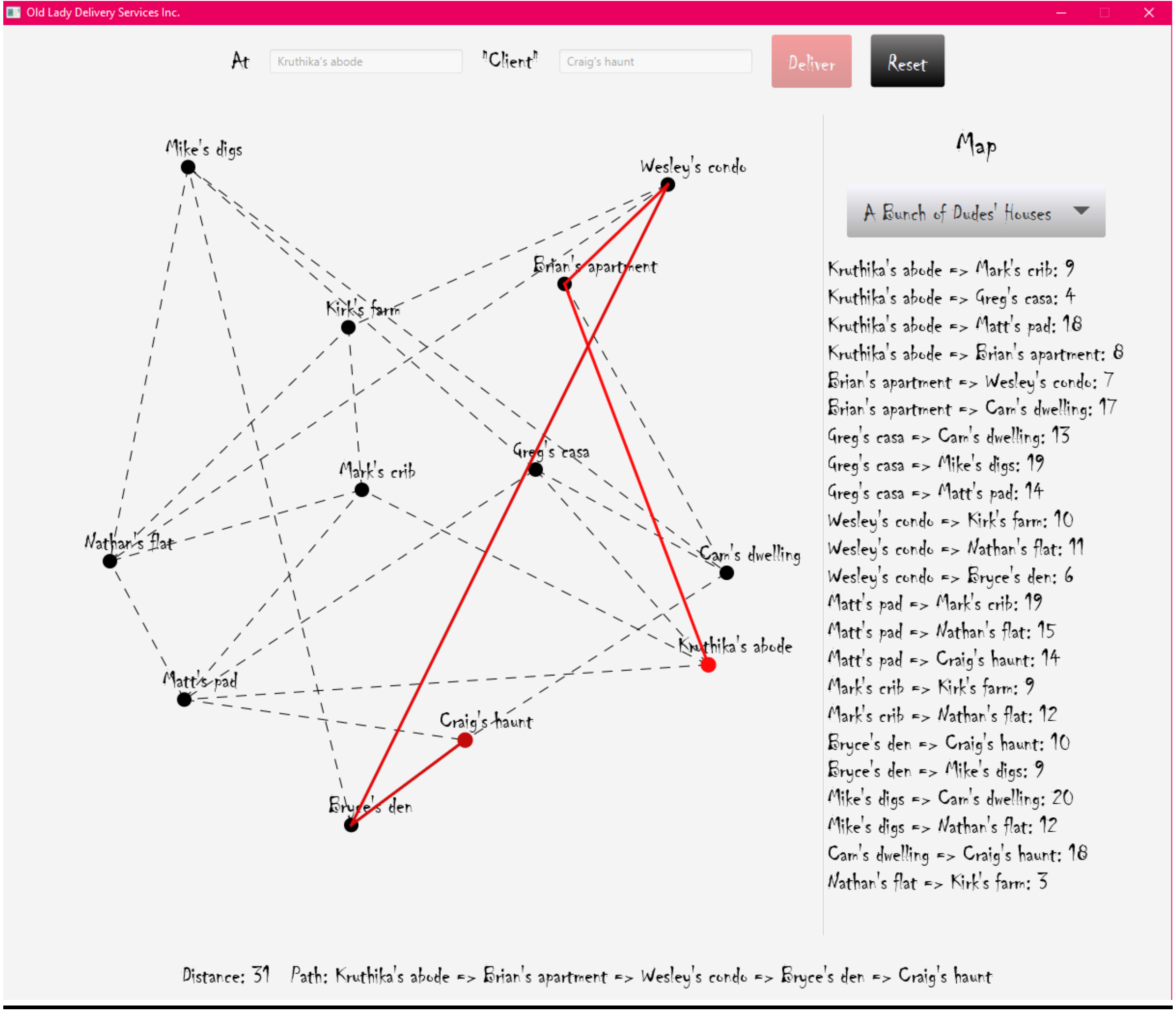
Solution preview

Deliver button yields alert box when the starting or target location is invalid.



Invalid start and target. They are probably protected by the CIA

Results will be shown in the bottom of the app GUI specifying distance and path details. On canvas, the path that is found will be highlighted in red, starting from desired location and end in target location. The color will slightly dim through the process. Buttons will be disabled during drawing the path. However, after that reset button will be enabled for making another path request.



At: Kruthika's abode

"Client": Craig's haunt

```
Kruthika's abode => Mark's crib: 9
Kruthika's abode => Greg's casa: 4
Kruthika's abode => Matt's pad: 18
Kruthika's abode => Brian's apartment: 8
Brian's apartment => Wesley's condo: 7
Brian's apartment => Cam's dwelling: 17
Greg's casa => Cam's dwelling: 13
Greg's casa => Mike's digs: 19
Greg's casa => Matt's pad: 14
Wesley's condo => Kirk's farm: 10
Wesley's condo => Nathan's flat: 11
Wesley's condo => Bryce's den: 6
Matt's pad => Mark's crib: 19
Matt's pad => Nathan's flat: 15
Matt's pad => Craig's haunt: 14
Mark's crib => Kirk's farm: 9
Mark's crib => Nathan's flat: 12
Bryce's den => Craig's haunt: 10
Bryce's den => Mike's digs: 9
Mike's digs => Cam's dwelling: 20
Mike's digs => Nathan's flat: 12
Cam's dwelling => Craig's haunt: 18
Nathan's flat => Kirk's farm: 3
```

```
Distance: 31    Path: Kruthika's abode => Brian's apartment => Wesley's condo => Bryce's den => Craig's haunt
```

Possible Improvement

- Make map_processor process maps from text files.
- Better map locations coordinate generator algorithm that provides better painting, eliminates map description section and shows distance between locations on the map canvas. Look forward to 3D map canvas
- Implement binary heap Dijkstra algorithm for better time complexity while working with larger maps
- More tests for GUI/Dijkstra algorithm
- More complicated alert to give instructions, assist users
- Use CSS implementation of design to reduce code duplication
- Export to runnable .jar file and probably .apk file

Time log

Day --- Time	Task
03/14/2017 --- 7pm – 11:59pm	• Set up IntelliJ, read Kotlin documentations • Implemented classic “Hello World”
03/15/2017 --- 12am – 11am	• Completed function Dijkstra (…) • Introduced class the_old_nice_lady • Test Dijkstra (…) in-class
03/15/2017 --- 7pm – 11:59pm	• Added TestNG, read TestNG documentations • Introduced two test classes: the original map test and the made up small map test
03/16/2017 --- 12am – 2am	• Wrapped up testing, completed both test classes, finished dd_qa package
03/16/2017 --- 11am – 5pm	• Added JavaFx lib, read documentations • Introduced dd_app with simple JavaFX stage scene and GUI • Test app
03/16/2017 --- 7pm – 12pm	• Realized been building on Java project skeleton with Kotlin • Migrated all code to legit Kotlin project • Continued working on dd_app
03/17/2017	TGIF
03/18/2017 --- 10am – 5pm	• dd_app GUI looked way too dissapointing, recreated dd_app GUI • Style up font, buttons • Read a lot about css
03/18/2017 --- 7pm – 12pm	• Introduced map_painter • Moved all map graphic activities to map_painter
03/19/2017 --- 9am – 3pm	• Completed map_painter however realized how much space a map initialization took • Threading paintPath method so that the solution is displayed in real time • Introduced map_processor
03/19/2017 --- 7pm – 12pm	• Completed map_processor • Completed sounds implementation • Completed alert box implementation
03/20/2017 --- 7pm – 9pm	• Wrote this report and pray