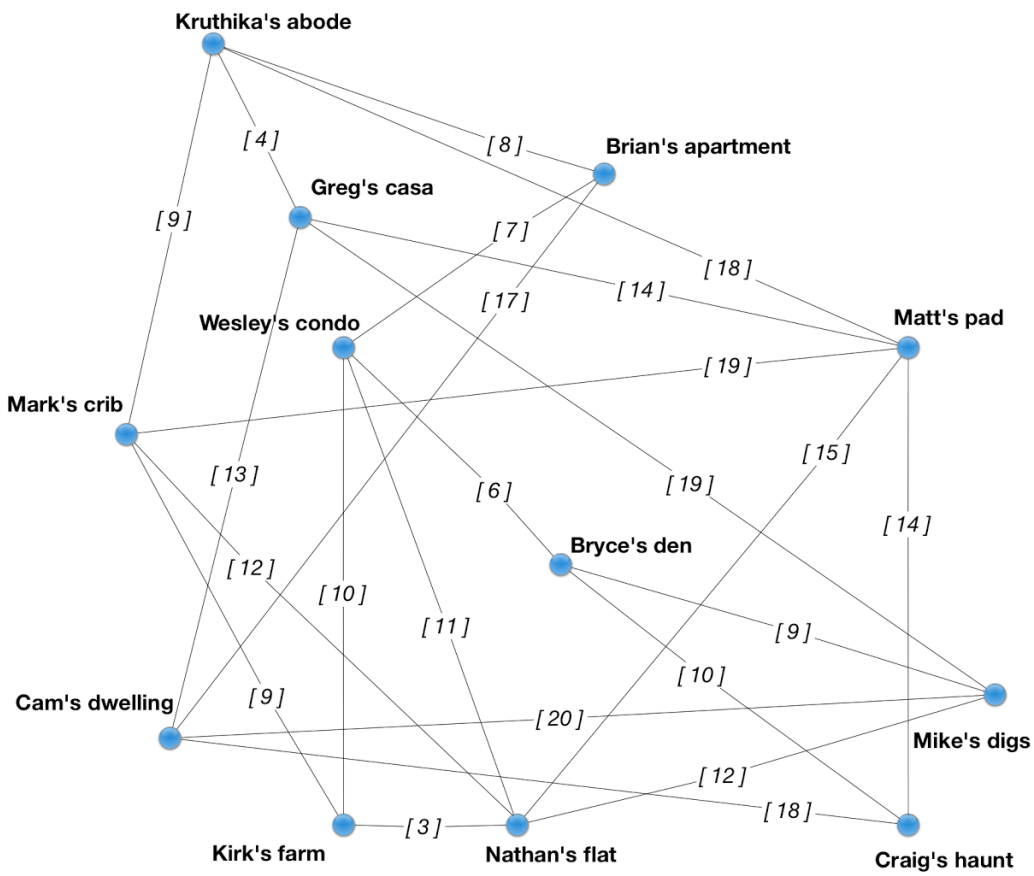
William Koppelberger
Dijkstra's Algorithm in Scala
2/2/15

- **The Problem**
  - https://github.com/postnati/doll-delivery
    - Find shortest path from Start to End using the List of edges(as Maps)
    - Given:
      - StartLocation *String*
      - EndLocation *String*
      - *List* of all edges(as *Maps*)



- **Problems to look out for**
    - How to keep track of the path
        - Use Linked List?
    - When checking for neighboring edges, you must check both starting and ending locations because the edges are not duplicated as shown below:
        - ***Map("startLocation" -> "Kruthika's abode", "endLocation" -> "Mark's crib", "distance" -> 9),***
          Map("startLocation" -> "Kruthika's abode", "endLocation" -> "Greg's casa", "distance" -> 4),
          Map("startLocation" -> "Kruthika's abode", "endLocation" -> "Matt's pad", "distance" -> 18),
          Map("startLocation" -> "Kruthika's abode", "endLocation" -> "Brian's apartment", "distance" -> 8),
          Map("startLocation" -> "Brian's apartment", "endLocation" -> "Wesley's condo", "distance" -> 7),
          Map("startLocation" -> "Brian's apartment", "endLocation" -> "Cam's dwelling", "distance" -> 17),
          Map("startLocation" -> "Greg's casa", "endLocation" -> "Cam's dwelling", "distance" -> 13),
          Map("startLocation" -> "Greg's casa", "endLocation" -> "Mike's digs", "distance" -> 19),
          Map("startLocation" -> "Greg's casa", "endLocation" -> "Matt's pad", "distance" -> 14),

- **Dijkstra's Algorithm**
    - http://en.wikipedia.org/wiki/Dijkstra%27s_algorithm
    - https://www.youtube.com/watch?v=gdmfOwyQlcI

- **What is Scala?**
    - http://www.scala-lang.org/documentation/
    - Compare different languages: http://rosettacode.org/wiki/Tic-tac-toe
    - Scala is an *OOP* but has functional programming abilities that makes it powerful
        - We can make Java-ish pseudo code and convert to Scala

- **Understanding Dijkstra's in Java**
  - **Dijkstra's Concept**
    - Want to make pseudo code first so we can look up what we need to know/learn in scala to complete this task. Take what we know(Scala is similar to Java) and apply it here.

  - **Dijkstra's in Java-Pseudo-Code**

```
Foreach node set distance[node] = HIGH
SettledNodes = empty
UnSettledNodes = empty

Add sourceNode to UnSettledNodes
distance[sourceNode]= 0

while (UnSettledNodes is not empty) {
  evaluationNode = getNodeWithLowestDistance(UnSettledNodes)
  remove evaluationNode from UnSettledNodes
    add evaluationNode to SettledNodes
    evaluatedNeighbors(evaluationNode)
}

getNodeWithLowestDistance(UnSettledNodes){
  find the node with the lowest distance in UnSettledNodes and return it
}

evaluatedNeighbors(evaluationNode){                    AND which is not in SettledNodes {
  Foreach destinationNode which can be reached via an edge from evaluationNode
    edgeDistance = getDistance(edge(evaluationNode, destinationNode))
    newDistance = distance[evaluationNode] + edgeDistance
    if (distance[destinationNode]  > newDistance) {
      distance[destinationNode]  = newDistance
      add destinationNode to UnSettledNodes
    }
  }
}
```

  - **Dijkstra's in Java**
    - http://www.vogella.com/tutorials/JavaAlgorithmsDijkstra/article.html#dijkstra_overview
    - http://www.algolist.com/code/java/Dijkstra%27s_algorithm

- **Understanding Scala**
  - My Google Doc "*Learn Scala*": https://docs.google.com/a/mail.gvsu.edu/document/d/1ONlgzD3oAp3eN7jVR6qdfjzFg_mWYT5mQpbzQDIVkbY/edit
  - Follow tutorials
    - http://www.tutorialspoint.com/scala/index.htm
  - YouTube videos

- **Converting/Write Java/Pseudo-Code to Scala**
    - Learn about Scala: http://www.scala-lang.org/documentation/
        - CheatSheet: http://docs.scala-lang.org/cheatsheets/?_ga=1.135774769.1642426883.1421702470
        - *List*
            - *http://www.tutorialspoint.com/scala/scala_lists.htm*
        - *Map*
            - *https://www.youtube.com/watch?v=6v-_jnN807A*
            - *http://www.scala-lang.org/api/current/index.html#scala.collection.immutable.Map*

- **Follow Scala-Style-Guide Code**
    - http://docs.scala-lang.org/style/?_ga=1.135774769.1642426883.1421702470

- **Largest problems encountered**
    - Understanding Classes and Objects
    - How to iterate Lists/Maps and pull data out from them
    - Converting data types (mainly the 'Any' type)