**Report**

*What my code can do:* My code can find the shortest path and highlight the articulation points.

*What my code can’t do:* The challenge.

**A\* Algorithm pseudocode.**

Created a hashSet to store all the nodes that has been visited

Created a priority queue for the fringe

Initially all the nodes haven’t been visited yet and the fringe starts with the root node so I create a new A\*Node (start, null, 0, startFscore);

Offer the rootNode to the fringe.

while(!fringe.isEmpty()){

take the shortest heuristic out from the priority queue

check if(current node is not visited){

add that currentNode to the set

set the previousnode of the currentNode as the parent (currentNode.setParentnode(previousnode);

}

Check if (the currentNode reaches the end){ break;}

For( segment connected to the currentNode){

Gets the neighbouring nodes of the currentNode

Check if( the visited set doesn’t contain the neighbour) {

Add that neighbour into the set

Update gscore with the edge.length

Update fscore with new gScore + the fscore from the root and its neighbour

Set neighbour’s parent node to the current Node

Create new A\* object for the next node to be analysed

Push the next node into the fringe to be compared

}

}

**Pseudocode algorithm for articulation points iterative version**

Intialize stack as a single element (startnode, depth, root)

While stack is not empty {

Peek in stack( n\*, depth\*, parent\*);

If(depth value is to infinity){

Intializse depth and reach back depth(n\*) = depth, reachBack(n\*) = depth;

Children(n\*) = get all children nodes except the parent

}

Else if( !children is empty) {

Get a child from children list and remove

If( the depth is < infinity) {

Reach back = min (depth of child, reachBack)

}

Else push (child, depth+1, n) into the stack;

}

}

Else if no more child node in the children list

If n\* is not the start node {

Reachback of parent = min( reachback of currentNode, reachback of parent)

}

If reachback of current node > depth of parent then add that parent in the as articulation point

}

Then remove (n\*, depth\*, parent\*) from stack

Fin.

}

**Testing:**

For both algorithms I placed print statements in various sections to see if the algorithm was actually passing through that point. If it wasn’t then I would check that section and why it wasn’t working and eventually find the errors.