**Name: Dannasri Srinivasan**
**ID: 1001698730**
**Programming Language used: Java 8**
**Software Required: JDK 11, JRE 11**

**Commands to execute:**

1. Note: Go to the code folder <assignment1_code_dxs8730> on to the terminal. The folder should contain find_route.java, input1.txt, and h_kassed.txt files.
2. The input and heuristic text file should contain END OF INPUT at the end of the text file, else it leads to array index out of bound exception.
3. **Uniform Cost Search:**
   - Compile command: javac find_route.java
   - After compiling, it should generate 4 class file such as find_route.class (main class), find_route$HeuristicTypes.class(Inner class), find_route$InputTypes.class(Inner class), find_route$PathCalc.class(Inner class).
   - Execute command: java find_route <input txt file> <source> <destination>
   - Execute command Example: java find_route input1.txt Bremen Kassel
4. **A* search:**
   - Compile command: javac find_route.java
   - After compiling, it should generate 4 class file such as find_route.class (main class), find_route$HeuristicTypes.class(Inner class), find_route$InputTypes.class(Inner class), find_route$PathCalc.class(Inner class).
   -
   - Execute command: java find_route <input txt file> <source> <destination> <heuristic file>
   - Execute command Example: java find_route input1.txt Bremen Kassel h_kassel.txt

The Code is Structured as follows:

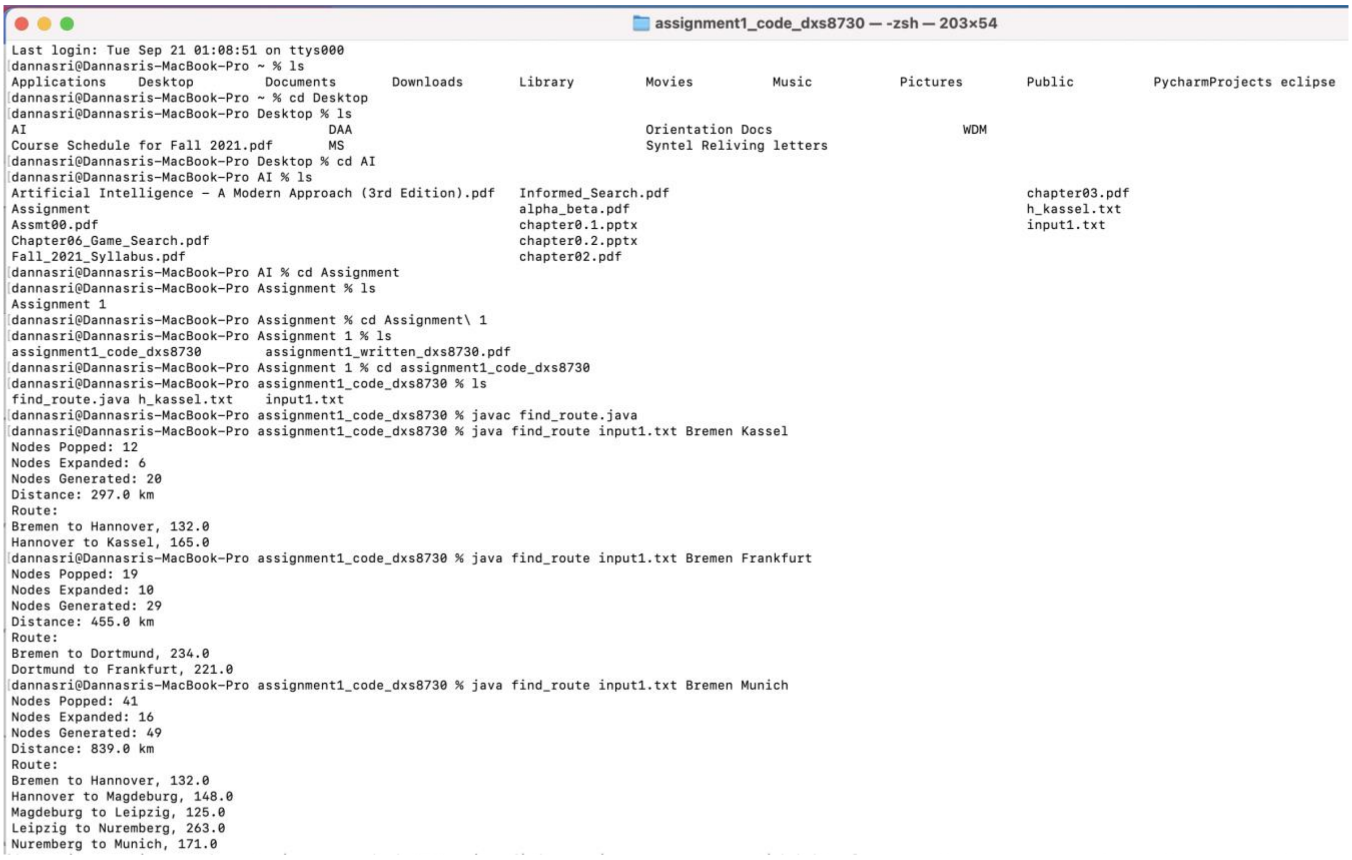
1. The code consists of three function UCS, A* and file reading for input and heuristic txt file.
2. Based on the input argument, main class calls UCS function if it has input txt file, source and destination else it calls A* function if the input argument has input text file, source, destination and heuristic file.
3. My class contains 3 more sub classes in order to store input file, heuristic file and the path of all expanded nodes.
4. UCS function:
   a. My UCS function contains following variable,
      i. Input – ArrayList<src, dest, cost>
      ii. Fringe – sortedMap< nodeCost , node>
      iii. ClosedSet – ArrayList<node>
      iv. NodesPoped – ArrayList<node>
      v. NodesGenerated – ArrayList<node>
      vi. NodesExpanded – ArrayList<node>
      vii. Path of all expanded Nodes – ArrayList<src, dest, cost, srcDepth, destDepth>

   b. My loop will continue until my current node is equal to goal node, initially my current node will be my source/start node, currentCost will be 0. After each iteration current node will be replaced by node which is present first in fringe, currentCost will be replaced by cumulative cost of source and destination cost.
   c. My currentNode will check for the successors based on the input file and add those value into Fringe.

d. In Every iteration my fringe will checked, if its empty, then return no route exits else continue with the iteration.
e. After finding the goal node, print NodesGenerated, NodesPoped, NodeExpanded, Distance and the route

5. A* function:
    a. My A* function contains following variable,
        i. Input – ArrayList<src, dest, cost>
        ii. HeuristicInput – ArrayList<node, heuristicCost>
        iii. Fringe – sortedMap<heuristicCost, Map< nodeCost, node>
        iv. ClosedSet – ArrayList<node>
        v. NodesPoped – ArrayList<node>
        vi. NodesGenerated – ArrayList<node>
        vii. NodesExpanded – ArrayList<node>
        viii. Path of all expanded Nodes – ArrayList<src, dest, cost, srcDepth, destDepth>
    b. My loop will continue until my current node is equal to goal node, initially my current node will be my source/start node, currentCost and currentHeuristicCost will be 0. After each iteration current node will be replaced by node which is present first in fringe, currentCost will be replaced by cumulative cost of source and destination cost and finally currentHeruristicCost will be replaced by cumulative cost of source, destination and HeruristicCost.
    c. My currentNode will check for the successors based on the input file and add those value into Fringe.
    d. In Every iteration my fringe will checked, if its empty, then return no route exits else continue with the iteration.
    e. After finding the goal node, print NodesGenerated, NodesPoped, NodeExpanded, Distance and the route

Sample Execution:



assignment1_code_dxs8730 — -zsh — 203×54

```
Last login: Tue Sep 21 01:08:51 on ttys000
dannasri@Dannasris-MacBook-Pro ~ % ls
Applications    Desktop         Documents       Downloads       Library         Movies          Music           Pictures        Public          PycharmProjects eclipse
dannasri@Dannasris-MacBook-Pro ~ % cd Desktop
dannasri@Dannasris-MacBook-Pro Desktop % ls
AI                                      DAA                                     Orientation Docs                        WDM
Course Schedule for Fall 2021.pdf       MS                                      Syntel Reliving letters
dannasri@Dannasris-MacBook-Pro Desktop % cd AI
dannasri@Dannasris-MacBook-Pro AI % ls
Artificial Intelligence - A Modern Approach (3rd Edition).pdf   Informed_Search.pdf                                     chapter03.pdf
Assignment                                                      alpha_beta.pdf                                          h_kassel.txt
Assmt00.pdf                                                     chapter0.1.pptx                                         input1.txt
Chapter06_Game_Search.pdf                                       chapter0.2.pptx
Fall_2021_Syllabus.pdf                                          chapter02.pdf
dannasri@Dannasris-MacBook-Pro AI % cd Assignment
dannasri@Dannasris-MacBook-Pro Assignment % ls
Assignment 1
dannasri@Dannasris-MacBook-Pro Assignment % cd Assignment\ 1
dannasri@Dannasris-MacBook-Pro Assignment 1 % ls
assignment1_code_dxs8730        assignment1_written_dxs8730.pdf
dannasri@Dannasris-MacBook-Pro Assignment 1 % cd assignment1_code_dxs8730
dannasri@Dannasris-MacBook-Pro assignment1_code_dxs8730 % ls
find_route.java h_kassel.txt    input1.txt
dannasri@Dannasris-MacBook-Pro assignment1_code_dxs8730 % javac find_route.java
dannasri@Dannasris-MacBook-Pro assignment1_code_dxs8730 % java find_route input1.txt Bremen Kassel
Nodes Popped: 12
Nodes Expanded: 6
Nodes Generated: 20
Distance: 297.0 km
Route:
Bremen to Hannover, 132.0
Hannover to Kassel, 165.0
dannasri@Dannasris-MacBook-Pro assignment1_code_dxs8730 % java find_route input1.txt Bremen Frankfurt
Nodes Popped: 19
Nodes Expanded: 10
Nodes Generated: 29
Distance: 455.0 km
Route:
Bremen to Dortmund, 234.0
Dortmund to Frankfurt, 221.0
dannasri@Dannasris-MacBook-Pro assignment1_code_dxs8730 % java find_route input1.txt Bremen Munich
Nodes Popped: 41
Nodes Expanded: 16
Nodes Generated: 49
Distance: 839.0 km
Route:
Bremen to Hannover, 132.0
Hannover to Magdeburg, 148.0
Magdeburg to Leipzig, 125.0
Leipzig to Nuremberg, 263.0
Nuremberg to Munich, 171.0
```