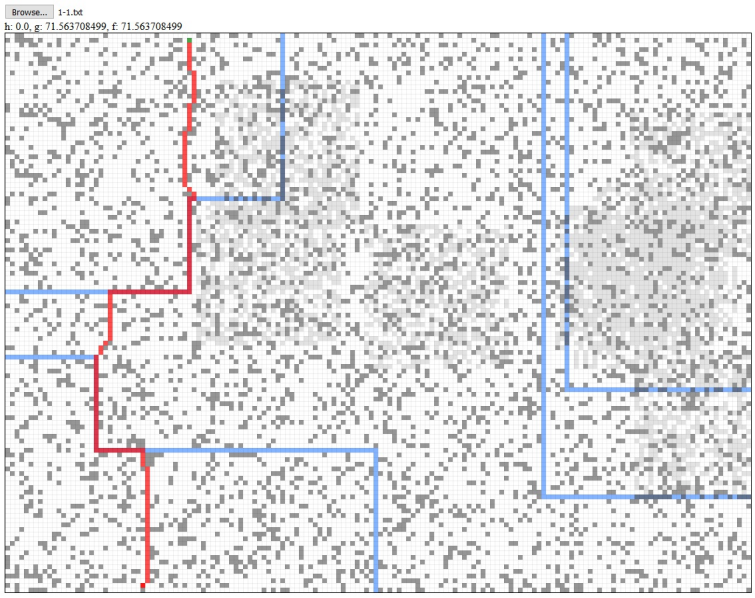
Heuristic Search Assignment

a)

front-end folder



b)

search.py has the abstract heuristic algorithm
SearchClasses.py has the various instantiations of it

c)

The implementation was optimized by using grids with O(1) efficiency instead of iterating through a list of the vertices. Grids were used to implement the closed list, so looking up if a vertex was alread visited is simply a random acces call to the grid: grid[vertex]. The g values were also held in a grid with default values set to pythons maximum integer value.

d)

Best admissible/consistent heuristic: To implement an admissible/consistent heuristic, we used a variation of the Manhattan distance. The heuristic simply divides the Manhattan distance by 4 in order to simulate the best case senario in which the goal and start vertices are directly connected by a highway. This makes the heuristic consistent since h(s') + c(s,s') will never be lower than h(s).

Manhattan: Manhattan distance from the point to the goal

Custom1: This heuristic weights the two components of the manhattan distance to account for the possibility of highways. It multiplies the difference in the row numbers of the start and goal vertices by the number of rows and dides everything by (number of rows + number of columns). It does the same with columns.

Custom2: This is similar to custom1 but instead of scaling the components of the Manhattan distance, it scales the components of the euclidean distance. As can be seen, this heuristic along with custom1 perform very well, trading off a small amount of cost, while considerably lowering the number of expansions.

Custom3: Maximum of distance from goal in the x-axis and distance from goal in the y-axis

e)

Benchmark data collected for each heuristic and the following weights [0,1,1.5,2.5]:

```
- Admissible, w: 0
average path cost:       111.84385927461732
average expanded nodes:  103047.46
average run time:        1.0151276588439941
average visited/optimum: 1.481836334744668
- Admissible, w: 1
average path cost:       111.84385927461732
average expanded nodes:  78096.7
average run time:        0.8967481184005738
average visited/optimum: 1.481836334744668
- Admissible, w: 1.5
average path cost:       115.50471856198911
average expanded nodes:  35331.06
average run time:        0.4995976734161377
average visited/optimum: 1.3325063384040943
- Admissible, w: 2.5
average path cost:       176.2280337783883
average expanded nodes:  1429.0
average run time:        0.07152163028717042
average visited/optimum: 1.0739431016619791
- Manhattan, w: 0
average path cost:       111.84385927461732
average expanded nodes:  103047.46
average run time:        1.0491833257675172
average visited/optimum: 1.481836334744668
- Manhattan, w: 1
average path cost:       139.80380944789016
average expanded nodes:  7422.84
average run time:        0.12337670803070068
average visited/optimum: 1.2126789475216173
- Manhattan, w: 1.5
average path cost:       183.72135122493907
average expanded nodes:  1081.66
average run time:        0.04007392406463623
average visited/optimum: 1.0638882634163012
- Manhattan, w: 2.5
average path cost:       186.26661384514657
average expanded nodes:  1047.16
average run time:        0.03606216430664062
average visited/optimum: 1.0623821824725703
```

```
- Custom1, w: 0
average path cost:       111.84385927461732
average expanded nodes:  103047.46
average run time:        0.950502758026123
average visited/optimum: 1.481836334744668
- Custom1, w: 1
average path cost:       114.05131669041234
average expanded nodes:  41443.94
average run time:        0.6460857582092285
average visited/optimum: 1.3750903480649506
- Custom1, w: 1.5
average path cost:       155.10482817641238
average expanded nodes:  4278.12
average run time:        0.085530686378479
average visited/optimum: 1.1136014256167401
- Custom1, w: 2.5
average path cost:       183.4517610314042
average expanded nodes:  1042.0
average run time:        0.05516284465789795
average visited/optimum: 1.0488947512356441
- Custom2, w: 0
average path cost:       111.84385927461732
average expanded nodes:  103047.46
average run time:        1.0684921503067017
average visited/optimum: 1.481836334744668
- Custom2, w: 1
average path cost:       115.52862116419536
average expanded nodes:  37782.74
average run time:        0.5155617475509644
average visited/optimum: 1.3789415862135757
- Custom2, w: 1.5
average path cost:       155.08399339882607
average expanded nodes:  1345.1
average run time:        0.0807570219039917
average visited/optimum: 1.0664133543542191
- Custom2, w: 2.5
average path cost:       169.07112252341875
average expanded nodes:  1011.66
average run time:        0.07502131938934326
average visited/optimum: 1.0259286752138546
- Custom3, w: 0
average path cost:       111.84385927461732
average expanded nodes:  103047.46
average run time:        1.042763123512268
average visited/optimum: 1.481836334744668
- Custom3, w: 1
average path cost:       125.29175608280576
average expanded nodes:  26403.3
average run time:        0.39967933654785154
average visited/optimum: 1.3113612389657283
- Custom3, w: 1.5
average path cost:       168.65461748553926
```

```
average expanded nodes:  1211.78
average run time:        0.04547203540802002
average visited/optimum: 1.0771243343236003
- Custom3, w: 2.5
average path cost:       173.18652440774997
average expanded nodes:  1080.38
average run time:        0.053687214851379395
average visited/optimum: 1.077040049702986
```

f)

Uniform search is of course the slowest of the algorithms since it is unguided by any heuristic, but it always returns the optimal path. The admissible heuristic we used will also alwyas return the optimal path, and it does so while exapanding fewer vertices. The reduction in the number of vertices expanded is more than 20% lower. But by increasing the weight to 1.5, the weighted search cuts down the number of expanded nodes by 65% compared to uniform cost search. But the increase in cost is a negligible ~3%. The Manhattan heuristic is the fastest search with A* that averages a decent path cost of 140 with only 7400 nodes expanded. However, the custom2 heuristic performs very well with A*, tieing with the admissible heuristic weighted at 1.5. Custom1 performs similarly, but slightly worse. The Custom3 heuristic also works very well with A*, but is very unoptimal with the weighted search.

g)

NOTE: The experiment here was run on a much slower laptop with a dual core laptop running at 1.8 Ghz (at 2.5 Ghz turbo during experiment), as opposed to the previous experiment, which was run on a quad core processor clocked at around 4 Ghz.

```
- Sequential, w1: 1 w2: 1
average path cost:       111.84385927461732
average expanded nodes:  13533.02
average run time:        3.1547413301467895
average visited/optimum: 1.481836334744668
- Sequential, w1: 1 w2: 1.2
average path cost:       112.0085396489327
average expanded nodes:  17744.68
average run time:        4.274209499359131
average visited/optimum: 1.4746543021389815
- Sequential, w1: 1 w2: 1.5
average path cost:       116.9803224716757
average expanded nodes:  19239.04
average run time:        4.484409561157227
average visited/optimum: 1.4167194276142416
- Sequential, w1: 1.5 w2: 1
average path cost:       112.02028063085463
average expanded nodes:  7849.54
average run time:        1.8071447658538817
average visited/optimum: 1.4807034067708518
- Sequential, w1: 1.5 w2: 1.2
```

```
average path cost:       112.51422507388192
average expanded nodes:  8735.8
average run time:        2.224102749824524
average visited/optimum: 1.4599195108502503
- Sequential, w1: 1.5 w2: 1.5
average path cost:       122.83878493810467
average expanded nodes:  8562.22
average run time:        2.0019946670532227
average visited/optimum: 1.3009197583383183
- Sequential, w1: 2 w2: 1
average path cost:       112.6952488785469
average expanded nodes:  5403.66
average run time:        1.4194062757492065
average visited/optimum: 1.4342973164708794
- Sequential, w1: 2 w2: 1.2
average path cost:       112.7615230485266
average expanded nodes:  5553.5
average run time:        1.45022723197937
average visited/optimum: 1.4342973164708794
- Sequential, w1: 2 w2: 1.5
average path cost:       116.08156272296823
average expanded nodes:  5483.4
average run time:        1.4014669179916381
average visited/optimum: 1.3772204126993024
```

h)

The implementation of sequential A* search was indeed much more efficient than the more straightforward, single-heuristic approaches. Note that the average time statistic is misleading as the sequential search was run on a much weaker processor. But looking at the number of overall expanded nodes, the sequential search is much more efficient that the simpler searches. The sequential search run with w1 = 2 and w2 = 1, has an average path cost of around 112.7, less than 1 point higher than the actual optimum of 111.8. But the number of expanded nodes is vastly fewer, at 5400, as opposed to the over 100,000 expanded by uniform cost search. The search is very efficient since all of the checks and calls are O(1) due to the use of grids instead of lists.