Question 1

Uniform Cost Search (superscript represents distance from start node)		
Expanded Node	*Open Queue*	*Closed Queue*
	1^{0}	
1^{0}	5^{5} 8^{24}	1^{0}
5^{5}	8^{24} 6^{40}	1^{0} 5^{5}
8^{24}	10^{39} 6^{40} 3^{47}	1^{0} 5^{5} 8^{24}
10^{39}	6^{40} 3^{47} 9^{65}	1^{0} 5^{5} 8^{24} 10^{39}
6^{40}	3^{47} 9^{65} 2^{78}	1^{0} 5^{5} 8^{24} 10^{39} 6^{40}
3^{47}	4^{54} 9^{65} 2^{78}	1^{0} 5^{5} 8^{24} 10^{39} 6^{40} 3^{47}
4^{54}	9^{65} 2^{78}	1^{0} 5^{5} 8^{24} 10^{39} 6^{40} 3^{47} 4^{54}
9^{65}	2^{78} 7^{100}	1^{0} 5^{5} 8^{24} 10^{39} 6^{40} 3^{47} 4^{54} 9^{65}
2^{78}	7^{100}	1^{0} 5^{5} 8^{24} 10^{39} 6^{40} 3^{47} 4^{54} 9^{65} 2^{78}
7^{100}		1^{0} 5^{5} 8^{24} 10^{39} 6^{40} 3^{47} 4^{54} 9^{65} 2^{78} 7^{100}

Path: 1 8 10 9 7
Cost: 110

Greedy Best First Search (superscript represents heuristic value)		
Expanded Node	*Open Queue*	*Closed Queue*
	1^{78}	
1^{78}	8^{60} 5^{75}	1^{78}
8^{60}	3^{37} 10^{57} 5^{75}	1^{78} 8^{60}
3^{37}	4^{30} 10^{57} 5^{75}	1^{78} 8^{60} 3^{37}
4^{30}	9^{35} 10^{57} 5^{75}	1^{78} 8^{60} 3^{37} 4^{30}
9^{35}	7^{0} 2^{32} 10^{57} 6^{60} 5^{75}	1^{78} 8^{60} 3^{37} 4^{30} 9^{35}
7^{0}	2^{32} 10^{57} 6^{60} 5^{75}	1^{78} 8^{60} 3^{37} 4^{30} 9^{35} 7^{0}

Path: 1 8 3 4 9 7

Cost: 107

A* Search (superscript represents distance from start node plus heuristic value)		
Expanded Node	*Open Queue*	*Closed Queue*
	1^{0}	
1^{0}	5^{80} 8^{84}	1^{0}
5^{80}	8^{84} 6^{100}	1^{0} 5^{80}
8^{84}	3^{84} 10^{96} 6^{100}	1^{0} 5^{80} 8^{84}
3^{84}	4^{84} 10^{96} 6^{100}	1^{0} 5^{80} 8^{84} 3^{84}
4^{84}	10^{96} 6^{100} 9^{107}	1^{0} 5^{80} 8^{84} 3^{84} 4^{84}
10^{96}	6^{100} 9^{107}	1^{0} 5^{80} 8^{84} 3^{84} 4^{84} 10^{96}
6^{100}	9^{107} 2^{110}	1^{0} 5^{80} 8^{84} 3^{84} 4^{84} 10^{96} 6^{100}
9^{107}	7^{107} 2^{110}	1^{0} 5^{80} 8^{84} 3^{84} 4^{84} 10^{96} 6^{100} 9^{107}
7^{107}	2^{110}	1^{0} 5^{80} 8^{84} 3^{84} 4^{84} 10^{96} 6^{100} 9^{107} 7^{107}

Path: 1 8 3 4 9 7
Cost: 107

Question 2

a. The order in which neighboring nodes are explored affects the result. Since no order was specified, I explored the node above, to the right, below, then to the left.

   BFS:

   Each node is a tuple of its x and y index/coordinate. At the start of the search, I create **openList** and **closedList**. **openList** is a <u>queue</u> of paths, each path being a list of nodes starting at the start node and ending at the current node. **closedList** list of nodes.

   At each iteration, a path is removed from **openList**, and its last node is the node being expanded. The node is ensured to be within the maze, not a wall, and not a previously visited node. If the node is the end node, the algorithm ends. Otherwise, the expanded node is added to **closedList**, and 4 new paths (one for each direction) are created and added to **openList**. Each new path appends a neighboring node to the current path.

Once the end node is found, the path is available as the current/end node is taken from the end of the path. The cost is the length of the path. The number of explored nodes is the length of **closedList**.

DFS:

Similar to BFS, except this uses a <u>stack</u> (LIFO queue) instead.

A*:

openList is a min heap of tuples. The first element of the tuple is the value from the heuristic function, the second value is a path of nodes. Python's *heapq* sorts tuples by their first value, so the heap will return the tuple with the smallest heuristic value.

The heuristic function returns the distance between 2 nodes. It uses the x and y indices/coordinates and calculates the distance using $\mathbf{d=\sqrt{(x1 - x2)^2+(y1 - y2)^2}}$. This function is admissible because it never overestimates the cost to move between 2 nodes. If nodes are in the same row/column with no walls between them (the best case scenario), this function returns the exact cost. If the nodes are diagonal to each other or if there is a wall between them, the actual cost will be greater than the heuristic value. Along with being admissible, this heuristic function is also reasonable because it makes sense to move to nodes that are closer to the end node.

The remainder of this algorithm is the same as BFS and DFS.

b. and c. For graphics, the path begins at the number 2 and increments for each step

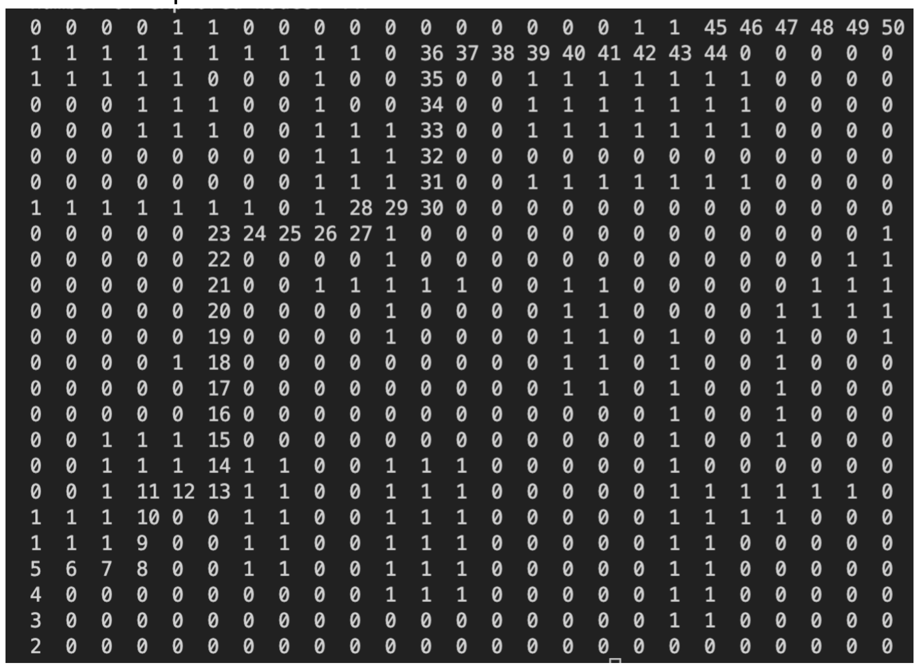
BFS	**S to E1:** path: [(11, 2), (12, 2), (13, 2), (14, 2), (15, 2), (16, 2), (16, 3), (16, 4), (16, 5), (16, 6), (16, 7), (16, 8), (16, 9), (17, 9), (17, 10), (17, 11), (18, 11), (19, 11), (19, 12), (19, 13), (19, 14), (19, 15), (19, 16), (19, 17), (19, 18), (19, 19), (19, 20), (19, 21), (19, 22), (19, 23)] cost: 30 number of explored nodes: 375

```
0  0  0  0  1  1  0  0  0  0  0  0  0  0  0  0  0  1  1  0  0  0  0  0  0
1  1  1  1  1  1  1  1  1  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
1  1  1  1  1  0  0  0  1  0  0  0  0  0  1  1  1  1  1  1  1  0  0  0  0
0  0  0  1  1  1  0  0  1  0  0  0  0  0  1  1  1  1  1  1  1  0  0  0  0
0  0  0  1  1  1  0  0  1  1  1  0  0  0  1  1  1  1  1  1  1  0  0  0  0
0  0  0  0  0  0  0  0  1  1  1  19 20 21 22 23 24 25 26 27 28 29 30 31 0
0  0  0  0  0  0  0  0  1  1  1  18 0  0  1  1  1  1  1  1  1  0  0  0  0
1  1  1  1  1  1  1  0  1  15 16 17 0  0  0  0  0  0  0  0  0  0  0  0  0
0  0  7  8  9  10 11 12 13 14 1  0  0  0  0  0  0  0  0  0  0  0  0  0  1
0  0  6  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  1  1
0  0  5  0  0  0  0  0  1  1  1  1  1  0  0  1  1  0  0  0  0  0  1  1  1
0  0  4  0  0  0  0  0  0  0  1  0  0  0  0  1  1  0  0  0  0  1  1  1  1
0  0  3  0  0  0  0  0  0  0  1  0  0  0  0  1  1  0  1  0  0  1  0  0  1
0  0  2  0  1  0  0  0  0  0  0  0  0  0  0  1  1  0  1  0  0  1  0  0  0
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  1  0  1  0  0  1  0  0  0
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  1  0  0  0
0  0  1  1  1  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  1  0  0  0
0  0  1  1  1  0  1  1  0  0  1  1  1  0  0  0  0  0  1  0  0  0  0  0  0
0  0  1  0  0  0  1  1  0  0  1  1  1  0  0  0  0  0  1  1  1  1  1  1  0
1  1  1  0  0  0  1  1  0  0  1  1  1  0  0  0  0  0  1  1  1  1  0  0  0
1  1  1  0  0  0  1  1  0  0  1  1  1  0  0  0  0  0  1  1  0  0  0  0  0
0  0  0  0  0  0  1  1  0  0  1  1  1  0  0  0  0  0  1  1  0  0  0  0  0
0  0  0  0  0  0  0  0  0  0  1  1  1  0  0  0  0  0  1  1  0  0  0  0  0
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  1  0  0  0  0  0
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
```

S to E2:
path: [(11, 2), (12, 2), (13, 2), (14, 2), (15, 2), (16, 2), (16, 3), (16, 4), (16, 5), (16, 6), (16, 7), (17, 7), (18, 7), (19, 7), (19, 6), (19, 5), (19, 4), (19, 3), (19, 2), (20, 2), (21, 2)]
cost: 21
number of explored nodes: 249

```
0  0  0  0  1  1  0  0  0  0  0  0  0  0  0  0  0  1  1  0  0  0  0  0  0
1  1  1  1  1  1  1  1  1  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
1  1  1  1  1  0  0  0  1  0  0  0  0  0  1  1  1  1  1  1  1  0  0  0  0
0  0  22 1  1  1  0  0  1  0  0  0  0  0  1  1  1  1  1  1  1  0  0  0  0
0  0  21 1  1  1  0  0  1  1  1  0  0  0  1  1  1  1  1  1  1  0  0  0  0
0  0  20 19 18 17 16 15 1  1  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0
0  0  0  0  0  0  0  14 1  1  1  0  0  0  1  1  1  1  1  1  1  0  0  0  0
1  1  1  1  1  1  1  13 1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
0  0  7  8  9  10 11 12 0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  1
0  0  6  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  1  1
0  0  5  0  0  0  0  0  1  1  1  1  1  0  0  1  1  0  0  0  0  0  1  1  1
0  0  4  0  0  0  0  0  0  0  1  0  0  0  0  1  1  0  0  0  0  1  1  1  1
0  0  3  0  0  0  0  0  0  0  1  0  0  0  0  1  1  0  1  0  0  1  0  0  1
0  0  2  0  1  0  0  0  0  0  0  0  0  0  0  1  1  0  1  0  0  1  0  0  0
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  1  0  1  0  0  1  0  0  0
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  1  0  0  0
0  0  1  1  1  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  1  0  0  0
0  0  1  1  1  0  1  1  0  0  1  1  1  0  0  0  0  0  1  0  0  0  0  0  0
0  0  1  0  0  0  1  1  0  0  1  1  1  0  0  0  0  0  1  1  1  1  1  1  0
1  1  1  0  0  0  1  1  0  0  1  1  1  0  0  0  0  0  1  1  1  1  0  0  0
1  1  1  0  0  0  1  1  0  0  1  1  1  0  0  0  0  0  1  1  0  0  0  0  0
0  0  0  0  0  0  1  1  0  0  1  1  1  0  0  0  0  0  1  1  0  0  0  0  0
0  0  0  0  0  0  0  0  0  0  1  1  1  0  0  0  0  0  1  1  0  0  0  0  0
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  1  0  0  0  0  0
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
```

(0, 0) to (24, 24):
path: [(0, 0), (1, 0), (2, 0), (3, 0), (3, 1), (3, 2), (3, 3), (4, 3), (5, 3), (6,

<table>
<tr><td></td><td>3), (6, 4), (6, 5), (7, 5), (8, 5), (9, 5), (10, 5), (11, 5), (12, 5), (13, 5), (14, 5), (15, 5), (16, 5), (16, 6), (16, 7), (16, 8), (16, 9), (17, 9), (17, 10), (17, 11), (18, 11), (19, 11), (20, 11), (21, 11), (22, 11), (23, 11), (23, 12), (23, 13), (23, 14), (23, 15), (23, 16), (23, 17), (23, 18), (23, 19), (24, 19), (24, 20), (24, 21), (24, 22), (24, 23), (24, 24)]
cost: 49
number of explored nodes: 447

0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 1 1 45 46 47 48 49 50
1 1 1 1 1 1 1 1 1 1 0 36 37 38 39 40 41 42 43 44 0 0 0 0 0
1 1 1 1 1 0 0 0 1 0 0 35 0 0 1 1 1 1 1 1 1 0 0 0 0
0 0 0 1 1 1 0 0 1 0 0 34 0 0 1 1 1 1 1 1 1 0 0 0 0
0 0 0 1 1 1 0 0 1 1 1 33 0 0 1 1 1 1 1 1 1 0 0 0 0
0 0 0 0 0 0 0 0 1 1 1 32 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 1 1 1 31 0 0 1 1 1 1 1 1 1 0 0 0 0
1 1 1 1 1 1 1 0 1 28 29 30 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 23 24 25 26 27 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1
0 0 0 0 0 22 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 1
0 0 0 0 0 21 0 0 1 1 1 1 1 0 0 1 1 0 0 0 0 0 1 1 1
0 0 0 0 0 20 0 0 0 0 1 0 0 0 0 1 1 0 0 0 0 1 1 1 1
0 0 0 0 0 19 0 0 0 0 1 0 0 0 0 1 1 0 1 0 0 1 0 0 1
0 0 0 0 1 18 0 0 0 0 0 0 0 0 0 1 1 0 1 0 0 1 0 0 0
0 0 0 0 0 17 0 0 0 0 0 0 0 0 0 1 1 0 1 0 0 1 0 0 0
0 0 0 0 0 16 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0
0 0 1 1 1 15 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0
0 0 1 1 1 14 1 1 0 0 1 1 1 0 0 0 0 0 1 0 0 0 0 0 0
0 0 1 11 12 13 1 1 0 0 1 1 1 0 0 0 0 0 1 1 1 1 1 1 0
1 1 1 10 0 0 1 1 0 0 1 1 1 0 0 0 0 0 1 1 1 1 0 0 0
1 1 1 9 0 0 1 1 0 0 1 1 1 0 0 0 0 0 1 1 0 0 0 0 0
5 6 7 8 0 0 1 1 0 0 1 1 1 0 0 0 0 0 1 1 0 0 0 0 0
4 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 1 1 0 0 0 0 0
3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0
2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0</td></tr>
<tr><td>DFS</td><td>S to E1:
path: [(11, 2), (12, 2), (13, 2), (14, 2), (15, 2), (16, 2), (16, 3), (16, 4), (16, 5), (16, 6), (16, 7), (16, 8), (16, 9), (17, 9), (17, 10), (17, 11), (18, 11), (19, 11), (20, 11), (21, 11), (22, 11), (23, 11), (24, 11), (24, 12), (24, 13), (24, 14), (24, 15), (24, 16), (23, 16), (23, 17), (23, 18), (23, 19), (24, 19), (24, 20), (24, 21), (24, 22), (24, 23), (24, 24), (23, 24), (22, 24), (21, 24), (20, 24), (19, 24), (18, 24), (17, 24), (17, 23), (18, 23), (19, 23)]
cost: 48
number of explored nodes: 77</td></tr>
</table>

```
0  0  0  0  1  1  0  0  0  0  0  24 25 26 27 28 29 1  1  34 35 36 37 38 39
1  1  1  1  1  1  1  1  1  1  0  23 0  0  0  0  30 31 32 33 0  0  0  0  40
1  1  1  1  1  0  0  0  1  0  0  22 0  0  1  1  1  1  1  1  1  0  0  0  41
0  0  0  1  1  1  0  0  1  0  0  21 0  0  1  1  1  1  1  1  1  0  0  0  42
0  0  0  1  1  1  0  0  1  1  1  20 0  0  1  1  1  1  1  1  1  0  0  0  43
0  0  0  0  0  0  0  0  1  1  1  19 0  0  0  0  0  0  0  0  0  0  0  49 44
0  0  0  0  0  0  0  0  1  1  1  18 0  0  1  1  1  1  1  1  1  0  0  48 45
1  1  1  1  1  1  1  0  1  15 16 17 0  0  0  0  0  0  0  0  0  0  0  47 46
0  0  7  8  9  10 11 12 13 14 1  0  0  0  0  0  0  0  0  0  0  0  0  0  1
0  0  6  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  1  1
0  0  5  0  0  0  0  0  1  1  1  1  1  0  0  1  1  0  0  0  0  0  1  1  1
0  0  4  0  0  0  0  0  0  0  1  0  0  0  0  1  1  0  0  0  0  1  1  1  1
0  0  3  0  0  0  0  0  0  0  1  0  0  0  0  1  1  0  1  0  0  1  0  0  1
0  0  2  0  1  0  0  0  0  0  0  0  0  0  0  1  1  0  1  0  0  1  0  0  0
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  1  0  1  0  0  1  0  0  0
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  1  0  0  0
0  0  1  1  1  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  1  0  0  0
0  0  1  1  1  0  1  1  0  0  1  1  1  0  0  0  0  0  1  0  0  0  0  0  0
0  0  1  0  0  0  1  1  0  0  1  1  1  0  0  0  0  0  1  1  1  1  1  1  0
1  1  1  0  0  0  1  1  0  0  1  1  1  0  0  0  0  0  1  1  1  1  0  0  0
1  1  1  0  0  0  1  1  0  0  1  1  1  0  0  0  0  0  1  1  0  0  0  0  0
0  0  0  0  0  0  1  1  0  0  1  1  1  0  0  0  0  0  1  1  0  0  0  0  0
0  0  0  0  0  0  0  0  0  0  1  1  1  0  0  0  0  0  1  1  0  0  0  0  0
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  1  0  0  0  0  0
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
```

S to E2:
path: [(11, 2), (12, 2), (13, 2), (14, 2), (15, 2), (16, 2), (16, 3), (16, 4), (16, 5), (16, 6), (16, 7), (17, 7), (18, 7), (19, 7), (20, 7), (21, 7), (22, 7), (22, 6), (21, 6), (20, 6), (19, 6), (18, 6), (18, 5), (19, 5), (19, 4), (18, 4), (18, 3), (19, 3), (19, 2), (20, 2), (21, 2)]
cost: 31
number of explored nodes: 30

```
0  0  0  0  1  1  0  0  0  0  0  0  0  0  0  0  0  1  1  0  0  0  0  0  0
1  1  1  1  1  1  1  1  1  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
1  1  1  1  1  0  19 18 1  0  0  0  0  0  1  1  1  1  1  1  1  0  0  0  0
0  0  32 1  1  1  20 17 1  0  0  0  0  0  1  1  1  1  1  1  1  0  0  0  0
0  0  31 1  1  1  21 16 1  1  1  0  0  0  1  1  1  1  1  1  1  0  0  0  0
0  0  30 29 26 25 22 15 1  1  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0
0  0  0  28 27 24 23 14 1  1  1  0  0  0  1  1  1  1  1  1  1  0  0  0  0
1  1  1  1  1  1  1  13 1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
0  0  7  8  9  10 11 12 0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  1
0  0  6  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  1  1
0  0  5  0  0  0  0  0  1  1  1  1  1  0  0  1  1  0  0  0  0  0  1  1  1
0  0  4  0  0  0  0  0  0  0  1  0  0  0  0  1  1  0  0  0  0  1  1  1  1
0  0  3  0  0  0  0  0  0  0  1  0  0  0  0  1  1  0  1  0  0  1  0  0  1
0  0  2  0  1  0  0  0  0  0  0  0  0  0  0  1  1  0  1  0  0  1  0  0  0
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  1  0  1  0  0  1  0  0  0
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  1  0  0  0
0  0  1  1  1  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  1  0  0  0
0  0  1  1  1  0  1  1  0  0  1  1  1  0  0  0  0  0  1  0  0  0  0  0  0
0  0  1  0  0  0  1  1  0  0  1  1  1  0  0  0  0  0  1  1  1  1  1  1  0
1  1  1  0  0  0  1  1  0  0  1  1  1  0  0  0  0  0  1  1  1  1  0  0  0
1  1  1  0  0  0  1  1  0  0  1  1  1  0  0  0  0  0  1  1  0  0  0  0  0
0  0  0  0  0  0  1  1  0  0  1  1  1  0  0  0  0  0  1  1  0  0  0  0  0
0  0  0  0  0  0  0  0  0  0  1  1  1  0  0  0  0  0  1  1  0  0  0  0  0
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  1  0  0  0  0  0
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
```

(0, 0) to (24, 24):

<table>
<tr>
<td></td>
<td>path: [(0, 0), (1, 0), (2, 0), (3, 0), (3, 1), (3, 2), (3, 3), (4, 3), (5, 3), (6, 3), (6, 4), (6, 5), (7, 5), (8, 5), (9, 5), (10, 5), (11, 5), (12, 5), (13, 5), (14, 5), (15, 5), (16, 5), (16, 6), (16, 7), (16, 8), (16, 9), (17, 9), (17, 10), (17, 11), (18, 11), (19, 11), (20, 11), (21, 11), (22, 11), (23, 11), (24, 11), (24, 12), (24, 13), (24, 14), (24, 15), (24, 16), (23, 16), (23, 17), (23, 18), (23, 19), (24, 19), (24, 20), (24, 21), (24, 22), (24, 23), (24, 24)]
cost: 51
number of explored nodes: 80

0 0 0 0 1 1 0 0 0 0 0 37 38 39 40 41 42 1 1 47 48 49 50 51 52
1 1 1 1 1 1 1 1 1 1 0 36 0 0 0 0 43 44 45 46 0 0 0 0 0
1 1 1 1 1 0 0 0 1 0 0 35 0 0 1 1 1 1 1 1 1 0 0 0 0
0 0 0 1 1 1 0 0 1 0 0 34 0 0 1 1 1 1 1 1 1 0 0 0 0
0 0 0 1 1 1 0 0 1 1 1 33 0 0 1 1 1 1 1 1 1 0 0 0 0
0 0 0 0 0 0 0 0 1 1 1 32 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 1 1 1 31 0 0 1 1 1 1 1 1 1 0 0 0 0
1 1 1 1 1 1 1 0 1 28 29 30 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 23 24 25 26 27 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1
0 0 0 0 0 22 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 1
0 0 0 0 0 21 0 0 1 1 1 1 1 0 0 1 1 0 0 0 0 0 1 1 1
0 0 0 0 0 20 0 0 0 0 1 0 0 0 0 1 1 0 0 0 0 1 1 1 1
0 0 0 0 0 19 0 0 0 0 1 0 0 0 0 1 1 0 1 0 0 1 0 0 1
0 0 0 0 1 18 0 0 0 0 0 0 0 0 0 1 1 0 1 0 0 1 0 0 0
0 0 0 0 0 17 0 0 0 0 0 0 0 0 0 1 1 0 1 0 0 1 0 0 0
0 0 0 0 0 16 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0
0 0 1 1 1 15 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0
0 0 1 1 1 14 1 1 0 0 1 1 1 0 0 0 0 0 1 0 0 0 0 0 0
0 0 1 11 12 13 1 1 0 0 1 1 1 0 0 0 0 0 1 1 1 1 1 1 0
1 1 1 10 0 0 1 1 0 0 1 1 1 0 0 0 0 0 1 1 1 1 0 0 0
1 1 1 9 0 0 1 1 0 0 1 1 1 0 0 0 0 0 1 1 0 0 0 0 0
5 6 7 8 0 0 1 1 0 0 1 1 1 0 0 0 0 0 1 1 0 0 0 0 0
4 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 1 1 0 0 0 0 0
3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0
2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>A*</td>
<td>S to E1:
path: [(11, 2), (11, 3), (12, 3), (12, 4), (12, 5), (12, 6), (12, 7), (12, 8), (12, 9), (11, 9), (11, 10), (11, 11), (11, 12), (11, 13), (11, 14), (12, 14), (13, 14), (14, 14), (15, 14), (15, 15), (15, 16), (15, 17), (15, 18), (15, 19), (15, 20), (16, 20), (16, 21), (17, 21), (17, 22), (18, 22), (18, 23), (19, 23)]
cost: 32
number of explored nodes: 32</td>
</tr>
</table>

```
0  0  0  0  1  1  0  0  0  0  0  0  0  0  0  0  0  1  1  0  0  0  0  0  0
1  1  1  1  1  1  1  1  1  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
1  1  1  1  1  0  0  0  1  0  0  0  0  0  1  1  1  1  1  1  1  0  0  0  0
0  0  0  1  1  1  0  0  1  0  0  0  0  0  1  1  1  1  1  1  1  0  0  0  0
0  0  0  1  1  1  0  0  1  1  1  0  0  0  1  1  1  1  1  1  1  0  0  0  0
0  0  0  0  0  0  0  0  1  1  1  0  0  0  0  0  0  0  0  0  0  0  0  33 0
0  0  0  0  0  0  0  0  1  1  1  0  0  0  1  1  1  1  1  1  1  0  31 32 0
1  1  1  1  1  1  1  0  1  0  0  0  0  0  0  0  0  0  0  0  0  29 30 0  0
0  0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  27 28 0  0  1
0  0  0  0  0  0  0  0  0  0  1  0  0  0  20 21 22 23 24 25 26 0  0  1  1
0  0  0  0  0  0  0  0  1  1  1  1  1  0  19 1  1  0  0  0  0  0  1  1  1
0  0  0  0  0  0  0  0  0  0  1  0  0  0  18 1  1  0  0  0  0  1  1  1  1
0  0  0  4  5  6  7  8  9  10 1  0  0  0  17 1  1  0  1  0  0  1  0  0  1
0  0  2  3  1  0  0  0  0  11 12 13 14 15 16 1  1  0  1  0  0  1  0  0  0
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  1  0  1  0  0  1  0  0  0
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  1  0  0  0
0  0  1  1  1  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  1  0  0  0
0  0  1  1  1  0  1  1  0  0  1  1  1  0  0  0  0  0  1  0  0  0  0  0  0
0  0  1  0  0  0  1  1  0  0  1  1  1  0  0  0  0  0  1  1  1  1  1  1  0
1  1  1  0  0  0  1  1  0  0  1  1  1  0  0  0  0  0  1  1  1  1  0  0  0
1  1  1  0  0  0  1  1  0  0  1  1  1  0  0  0  0  0  1  1  0  0  0  0  0
0  0  0  0  0  0  1  1  0  0  1  1  1  0  0  0  0  0  1  1  0  0  0  0  0
0  0  0  0  0  0  0  0  0  0  1  1  1  0  0  0  0  0  1  1  0  0  0  0  0
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  1  0  0  0  0  0
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
```

S to E2:
path: [(11, 2), (12, 2), (13, 2), (14, 2), (15, 2), (16, 2), (16, 3), (16, 4), (16, 5), (16, 6), (16, 7), (17, 7), (18, 7), (18, 6), (18, 5), (18, 4), (19, 4), (19, 3), (19, 2), (20, 2), (21, 2)]
cost: 21
number of explored nodes: 29

```
0  0  0  0  1  1  0  0  0  0  0  0  0  0  0  0  0  1  1  0  0  0  0  0  0
1  1  1  1  1  1  1  1  1  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
1  1  1  1  1  0  0  0  1  0  0  0  0  0  1  1  1  1  1  1  1  0  0  0  0
0  0  22 1  1  1  0  0  1  0  0  0  0  0  1  1  1  1  1  1  1  0  0  0  0
0  0  21 1  1  1  0  0  1  1  1  0  0  0  1  1  1  1  1  1  1  0  0  0  0
0  0  20 19 18 0  0  0  1  1  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0
0  0  0  0  17 16 15 14 1  1  1  0  0  0  1  1  1  1  1  1  1  0  0  0  0
1  1  1  1  1  1  1  13 1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
0  0  7  8  9  10 11 12 0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  1
0  0  6  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  1  1
0  0  5  0  0  0  0  0  1  1  1  1  1  0  0  1  1  0  0  0  0  0  1  1  1
0  0  4  0  0  0  0  0  0  0  1  0  0  0  0  1  1  0  0  0  0  1  1  1  1
0  0  3  0  0  0  0  0  0  0  1  0  0  0  0  1  1  0  1  0  0  1  0  0  1
0  0  2  0  1  0  0  0  0  0  0  0  0  0  0  1  1  0  1  0  0  1  0  0  0
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  1  0  1  0  0  1  0  0  0
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  1  0  0  0
0  0  1  1  1  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  1  0  0  0
0  0  1  1  1  0  1  1  0  0  1  1  1  0  0  0  0  0  1  0  0  0  0  0  0
0  0  1  0  0  0  1  1  0  0  1  1  1  0  0  0  0  0  1  1  1  1  1  1  0
1  1  1  0  0  0  1  1  0  0  1  1  1  0  0  0  0  0  1  1  1  1  0  0  0
1  1  1  0  0  0  1  1  0  0  1  1  1  0  0  0  0  0  1  1  0  0  0  0  0
0  0  0  0  0  0  1  1  0  0  1  1  1  0  0  0  0  0  1  1  0  0  0  0  0
0  0  0  0  0  0  0  0  0  0  1  1  1  0  0  0  0  0  1  1  0  0  0  0  0
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  1  0  0  0  0  0
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
```

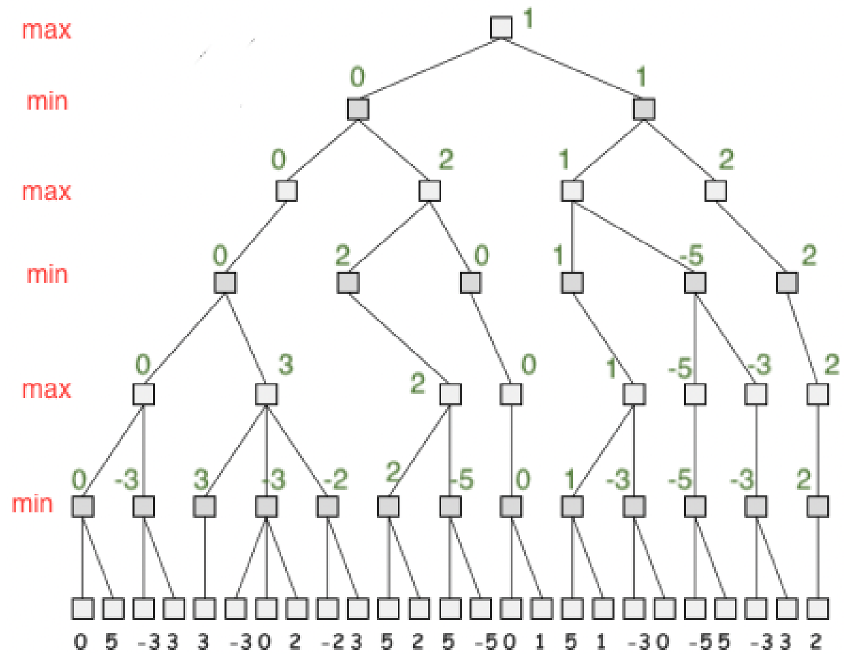
(0, 0) to (24, 24):
path: [(0, 0), (0, 1), (1, 1), (1, 2), (2, 2), (2, 3), (3, 3), (3, 4), (4, 4), (4,

5), (5, 5), (6, 5), (7, 5), (8, 5), (8, 6), (8, 7), (8, 8), (8, 9), (9, 9), (9, 10), (10, 10), (10, 11), (11, 11), (11, 12), (12, 12), (12, 13), (13, 13), (13, 14), (14, 14), (15, 14), (15, 15), (15, 16), (16, 16), (16, 17), (17, 17), (17, 18), (17, 19), (17, 20), (17, 21), (18, 21), (19, 21), (20, 21), (21, 21), (21, 22), (22, 22), (22, 23), (23, 23), (23, 24), (24, 24)]
cost: 49
number of explored nodes: 48

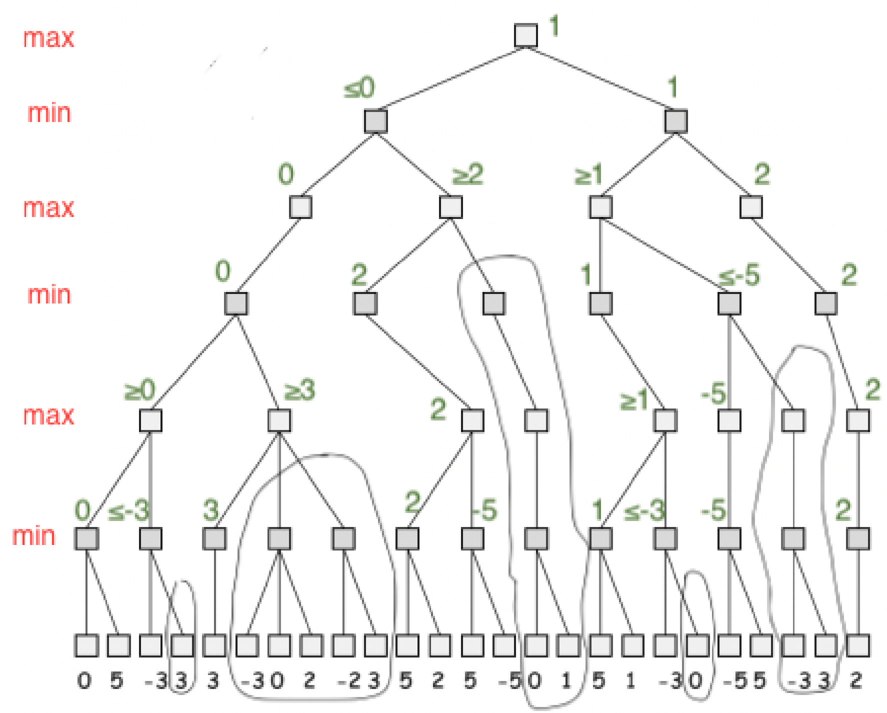
```
0  0  0  0  1  1  0  0  0  0  0  0  0  0  0  0  0  1  1  0  0  0  0  0  50
1  1  1  1  1  1  1  1  1  1  0  0  0  0  0  0  0  0  0  0  0  0  0  48 49
1  1  1  1  1  0  0  0  1  0  0  0  0  0  1  1  1  1  1  1  1  0  46 47 0
0  0  0  1  1  1  0  0  1  0  0  0  0  0  1  1  1  1  1  1  1  44 45 0  0
0  0  0  1  1  1  0  0  1  1  1  0  0  0  1  1  1  1  1  1  1  43 0  0  0
0  0  0  0  0  0  0  0  1  1  1  0  0  0  0  0  0  0  0  0  0  42 0  0  0
0  0  0  0  0  0  0  0  1  1  1  0  0  0  1  1  1  1  1  1  1  41 0  0  0
1  1  1  1  1  1  1  0  1  0  0  0  0  0  0  0  0  36 37 38 39 40 0  0  0
0  0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  34 35 0  0  0  0  0  0  1
0  0  0  0  0  0  0  0  0  0  1  0  0  0  31 32 33 0  0  0  0  0  0  1  1
0  0  0  0  0  0  0  0  1  1  1  1  1  0  30 1  1  0  0  0  0  0  1  1  1
0  0  0  0  0  0  0  0  0  0  1  0  0  28 29 1  1  0  0  0  0  1  1  1  1
0  0  0  0  0  0  0  0  0  0  1  0  26 27 0  1  1  0  1  0  0  1  0  0  1
0  0  0  0  1  0  0  0  0  0  0  24 25 0  0  1  1  0  1  0  0  1  0  0  0
0  0  0  0  0  0  0  0  0  0  22 23 0  0  0  1  1  0  1  0  0  1  0  0  0
0  0  0  0  0  0  0  0  0  20 21 0  0  0  0  0  0  0  1  0  0  1  0  0  0
0  0  1  1  1  15 16 17 18 19 0  0  0  0  0  0  0  0  1  0  0  1  0  0  0
0  0  1  1  1  14 1  1  0  0  1  1  1  0  0  0  0  0  1  0  0  0  0  0  0
0  0  1  0  0  13 1  1  0  0  1  1  1  0  0  0  0  0  1  1  1  1  1  1  0
1  1  1  0  0  12 1  1  0  0  1  1  1  0  0  0  0  0  1  1  1  1  0  0  0
1  1  1  0  10 11 1  1  0  0  1  1  1  0  0  0  0  0  1  1  0  0  0  0  0
0  0  0  8  9  0  1  1  0  0  1  1  1  0  0  0  0  0  1  1  0  0  0  0  0
0  0  6  7  0  0  0  0  0  0  1  1  1  0  0  0  0  0  1  1  0  0  0  0  0
0  4  5  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  1  0  0  0  0  0
2  3  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
```

Question 3

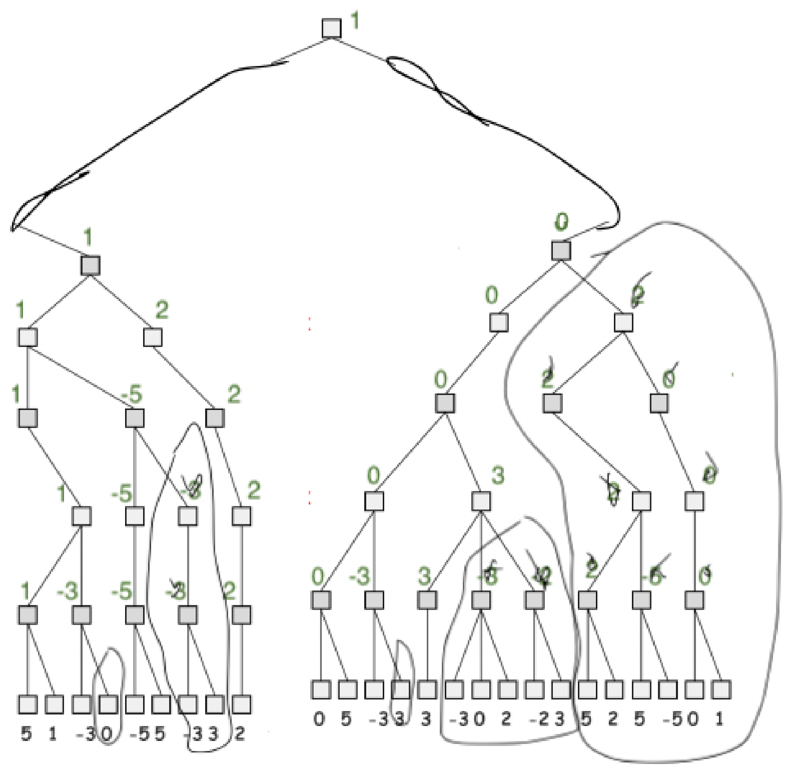
a.



b. Pruned nodes are circled in black. All necessary calculations are shown in green.



c. Restructure the tree by swapping the root's subtrees. This makes it so that the alpha value of the root is ≥1 after traversing the left subtree. The beta value of the root's right child is ≤0 after traversing the right child's left subtree. This enables the right child's right subtree to be pruned, increasing the number of pruned nodes and branches.



Question 4

a. Problem Formulation: Find the coordinate (x1 and x2 values) at the global minimum, which minimizes the cost.
    1. State Space: points (x1, x2) such that x1 and x2 are between (inclusive) -100 and 100
    2. Initial State: best (lowest cost) point out of 10 randomly generated points
    3. Goal state and goal test: any point (x1, x2) with cost < -0.99
    4. Concept of cost: the output of the Easom function
    5. Set of actions:

a. Move from the current point to another point in its neighborhood, as long as both coordinates of the point is between -100 and 100
b. Decrease temperature

Neighborhood function: Given a value for **range**, the neighborhood of a point at (x, y) consists of points with coordinates (x ± **range**, y ± **range**). It's essentially a square around the point. Range is set to 2 by default.

Cost function: The Easom function, the cost of a point at (x, y) is the value of the Easom function with x = x1 and y = x2. I convert the values of x and y from degree to radian

b.

i. 10 random initial points sorted by cost:

(-1.730430538553419e-36, (11, 7))
(-3.3614738847391093e-105, (17, 10))
(0.0, (-83, -16))
(-0.0, (-69, 58))
(0.0, (-52, -81))
(0.0, (-42, -76))
(-0.0, (-42, -23))
(0.0, (-42, 99))
(-0.0, (23, -92))
(-0.0, (78, -61))

The best of the 10 is (11, 7) with cost of ~-1.73e-36

Note: Points are randomly generated each iteration, one iteration was chosen as an example

ii. I ranged the initial temperature from 500 to 1000. Assuming an average decrease of 1 per iteration (linear annealing scheduling with alpha = 1), and the average distance between a node and another node in its neighborhood being roughing **range** / 2 (which is 1 using range = 2), this range makes it possible for exploration through the entire graph. At the same time, not too much exploration.

Note: Initial temperatures are randomly generated each iteration. The question says to use the best temperature for future steps but I need to select an annealing schedule to run the SA algorithm. However, to test annealing schedules, I need to use the best temperature. This seems like circular dependency to me, so tried all 10 temperatures instead of choosing the best one with each of the 9 annealing schedules (90 runs of SA in total).

iii. I chose linear with alpha values 0.5, 1, 2. Exponential with alpha values 0.9, 0.99, 0.999. Slow with alpha/beta values 0.001, 0.0005, 0.0001.

All 3 schedule functions use equations from class:

```
def linear(temp, a):
    return temp - a


def geometric(temp, a):
    return temp * a


def slowDecrease(temp, b):
    return temp / (1 + b * temp)
```

c. The best solution I found was (3.166210661468341, 3.157544962342315) with a cost of -0.9987100130402456. The SA settings were:
   - Annealing schedule: slow
   - Annealing alpha: 0.0001
   - Initial Temperature: ~689
   - Initial Solution: (-97.12295396468289, 10.589869468049756) with cost 0

   This setting was better than other settings using a linear or exponential schedule because it spends more time exploring (more time at a high temperature). A very small area around the global minimum is decreasing towards the minimum. Meaning it's harder to find a point that can lead to the global minimum at low temperatures. The additional exploration time increases the probability of finding a point near the global minimum before the temperature gets low.

   The alpha value was 0.0001 which increases the total number of iterations with a high temperature compared to the other possible values (0.001 and 0.0005). More iterations at a high temperature increases the amount of exploration and probability of finding a promising region.

   The initial Temperature was 564 and affect the number of iterations. It's closer to the smaller end of the range (500 compared to 1000). This likely didn't make a big difference as increasing the temperature from 500 to 1000 slightly increases the number of iterations with the same settings.

The initial solution’s x2 was close to the final point, which makes finding the global minimum easier than initial solutions that are further away.