Conclusion:

In this homework, I write four algorithms, BFS, DFS, A*, and IDA*, all algorithms are functional and produce corresponding output, except for the IDA* in arena3. It takes too long to complete the search.

Note:

Since when you run all the algorithms to arena3, the solution for BFS, DFS, A* will still be written to file, but it will stuck on IDA*.

Question 4.
(a) Cost of the path (step)

	BFS	DFS	A*	IDA*
Arena 1	10	19	10	10
Arena 2	74	431	74	74
Arena 3	229	1280	229	N/A

Cost: A* = BFS < IDA* < DFS

(b) Memory requirement (number of nodes stored)

	BFS	DFS	A*	IDA*
Arena 1	56	23	56	10
Arena 2	1425	1132	1425	74
Arena 3	3360	1765	3360	N/A

For BFS and A* algorithms, I used a 2D list to keep track of the nodes that are explored, so the size of memory is exactly the size of the map. For the DFS, we used one list to store the path, and another list to store the duplicate node, so the size of memory is the sum of the two lists. For IDA*, I only used a list to keep track of the path, so the size of memory is exactly the length of that path.

In memory usage: IDA* < DFS < A* < BFS

(c) Running time (ms)

	BFS	DFS	A*	IDA*
Arena 1	0.85	0.27	0.40	0.41
Arena 2	7.25	62.60	6.87	24.64
Arena 3	23.19	113.98	18.07	N/A

Running time: A* < BFS < IDA* < DFS

Summary: As you can see from the table, BFS and A* are optimal while IDA* and DFS are not. From memory aspect, BFS and A* both use lots memory, while DFS uses less and IDA* the least. For the running time, A* is slightly faster than BFS. IDA* and DFS are slower. IDA* is faster with simple maps, but slower with complicated maps. We can say that A* is strictly better than BFS, because they are tied in memory usage and optimality, but A* runs faster. DFS and IDA* can be seen as trading time for memory. In this situation, we should use IDA* if map is simple, DFS if map is complicated.

Question 5.

A* and IDA* will fail while not given goal position before starting. To calculate the cost, we need to know the heuristic function, which is acquired by estimating the cheapest cost from current position to goal position. If we don't know the position of the goal, then there will be no way to perform this estimation.

Question 6.

If the cost for moving from one position to neighboring position is not equal, heuristic search such as A* and IDA* will be the effective algorithms to find the solution. The most advantage of these algorithms is taking account into the heuristic function. When the distance to goal is unknown from the current state, they provide estimations for it to optimize the solutions. Compared to uninformed search i.e. DFS and BFS, which blindly expanding search while not knowing goal position, it may move in the reverse direction from the goal.

Question 7.

IDA* use less memory than the other three algorithms, but it takes lots of time to perform repeated search. Also it does not provide optimality.