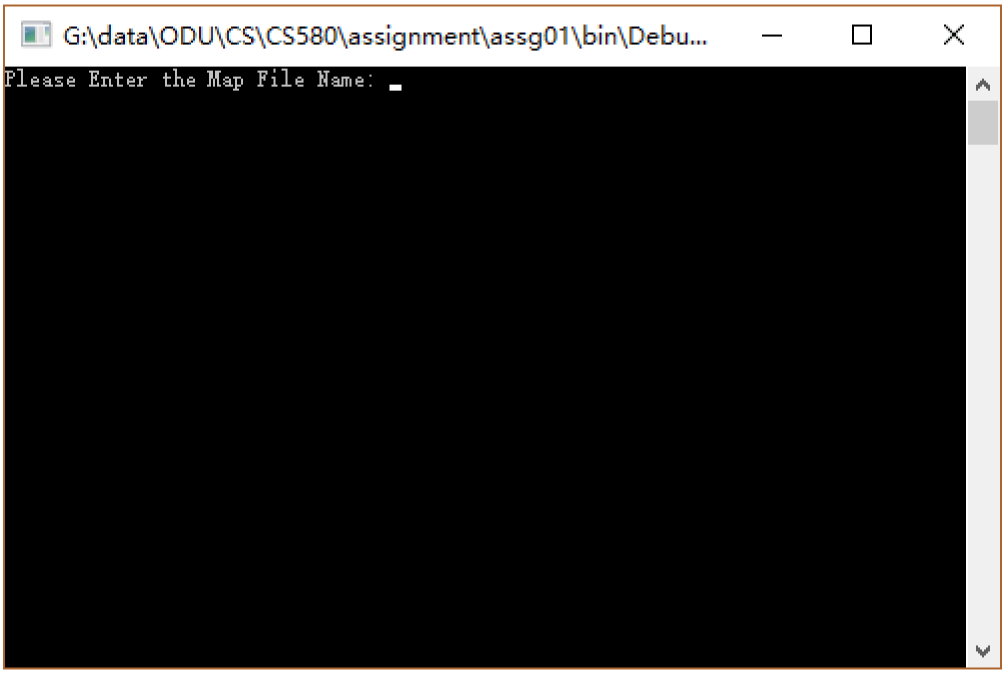
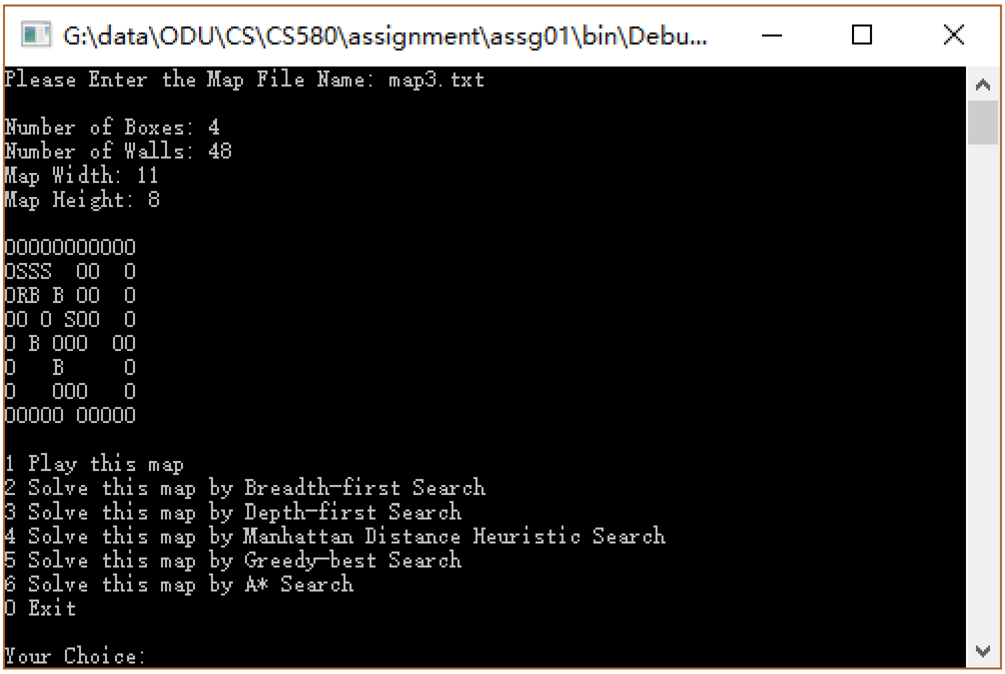
Execution

1. Please copy the map files into the program directory (\assg01\).
2. When you run the program, first you will be asked to input the name of map file. Please enter the full name of the map file (e.g. map.txt).



```
Please Enter the Map File Name: _
```

3. Once you enter the name of the map file, it will display the box number, wall number, width and height of the map. And the map will be displayed. Then you will be asked to choose what to do.

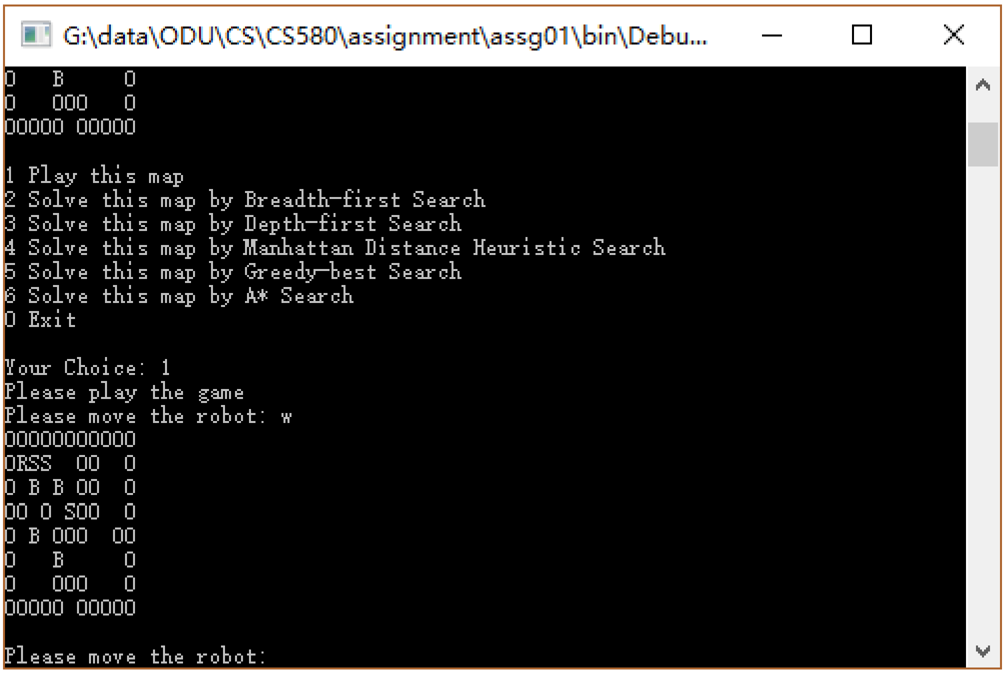


```
Please Enter the Map File Name: map3.txt

Number of Boxes: 4
Number of Walls: 48
Map Width: 11
Map Height: 8

00000000000
0SSS  00  0
0RB B 00  0
00 0 S00  0
0 B 000  00
0   B     0
0   000   0
00000 00000

1 Play this map
2 Solve this map by Breadth-first Search
3 Solve this map by Depth-first Search
4 Solve this map by Manhattan Distance Heuristic Search
5 Solve this map by Greedy-best Search
6 Solve this map by A* Search
0 Exit

Your Choice:
```

4. You can only choose the actions listed in the menu by entering the number in the

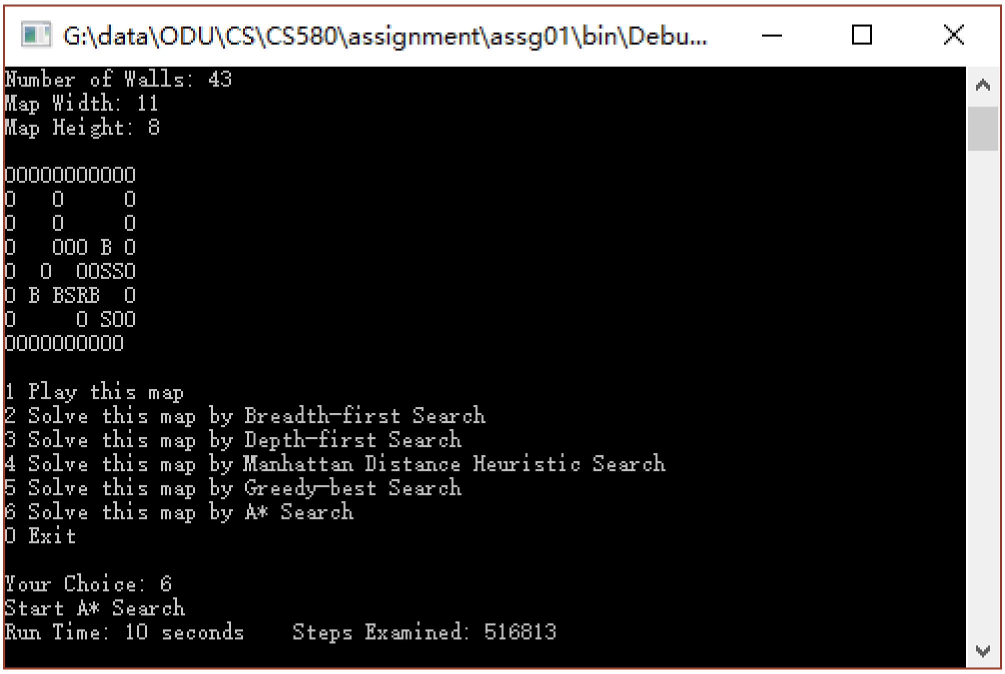
beginning.

4.1 If you choose to play this map, you may enter ‘w’ to move the robot up, ‘s’ down, ‘a’ left and ‘d’ right. After each step, it will display the map again. Note you are forbidden to push a box to a dead corner. If you want to give up, please enter ‘q’ to exit the program.



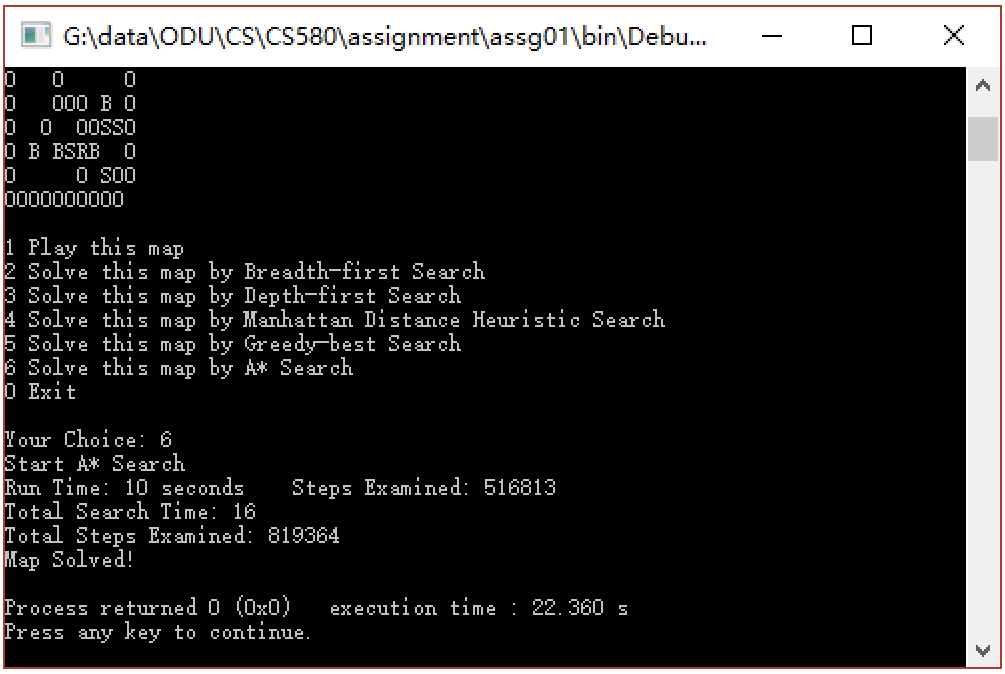
```
0   B     0
0   000   0
00000 00000

1 Play this map
2 Solve this map by Breadth-first Search
3 Solve this map by Depth-first Search
4 Solve this map by Manhattan Distance Heuristic Search
5 Solve this map by Greedy-best Search
6 Solve this map by A* Search
0 Exit

Your Choice: 1
Please play the game
Please move the robot: w
00000000000
0RSS  00  0
0 B B 00  0
00 0 S00  0
0 B 000  00
0   B     0
0   000   0
00000 00000

Please move the robot:
```

4.2 If you choose any algorithm to solve this map, it will output the progress every 10 seconds.



```
Number of Walls: 43
Map Width: 11
Map Height: 8

00000000000
0   0     0
0   0     0
0   000 B 0
0  0  00SS0
0 B BSRB  0
0     0 S00
0000000000

1 Play this map
2 Solve this map by Breadth-first Search
3 Solve this map by Depth-first Search
4 Solve this map by Manhattan Distance Heuristic Search
5 Solve this map by Greedy-best Search
6 Solve this map by A* Search
0 Exit

Your Choice: 6
Start A* Search
Run Time: 10 seconds    Steps Examined: 516813
```

5. When the solution has been found, it will output the total search time and steps. And the solution will be generated in the file ‘solution.txt’.



```
O   O     O
O   OOO B O
O  O  OOSSO
O B BSRB  O
O     O SOO
OOOOOOOOOO

1 Play this map
2 Solve this map by Breadth-first Search
3 Solve this map by Depth-first Search
4 Solve this map by Manhattan Distance Heuristic Search
5 Solve this map by Greedy-best Search
6 Solve this map by A* Search
0 Exit

Your Choice: 6
Start A* Search
Run Time: 10 seconds    Steps Examined: 516813
Total Search Time: 16
Total Steps Examined: 819364
Map Solved!

Process returned 0 (0x0)   execution time : 22.360 s
Press any key to continue.
```

Heuristic Greedy Algorithm Explanation

Manhattan distance heuristic assume that many boxes can be stored at one location. In my heuristic greedy algorithm, I force each box to be stored at different location and calculate the minimum sum distance (still use Manhattan distance and ignore the positions of obstacles) of all combinations. And I add the distance between the robot and its nearest box that has not been stored to the sum. So this sum distance will never be larger than the necessary steps needed to push all boxes to their positions.

Results

		Map1	Map2	Map3	Map4	Map5	Map6
Breadth First	steps	938	552	208781	9129	4463	441435
	time	<1s	<1s	2s	<1s	<1s	4s
Depth First	steps	787	428	163283	4914	3870	412693
	time	<1s	<1s	2s	<1s	<1s	3s
Manhattan Heuristic	steps	433	405	65438	6573	2732	145371
	time	<1s	<1s	1s	<1s	<1s	2s
Greedy Best	steps	401	332	134779	6907	2487	139172
	time	<1s	<1s	2s	<1s	<1s	2s
A*	steps	1238	795	322321	12686	7265	819364
	time	<1s	<1s	5s	<1s	<1s	14s