CMPE 300 ANALYSIS OF ALGORITHMS
PROJECT 3 - ANSWERS

PART 1

1.1) Fill the steps of one successful and one unsuccessful execution for each p value

1.1.1) Success - p=0.7

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

1.1.2) Unsuccessful - p=0.7

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

1.1.3) Success - p=0.8

	0	1	2	3	4	5	6	7
0	-1	-1	32	23	-1	35	26	-1
1	31	**-1**	**21**	**28**	**33**	**24**	**43**	**36**
2	-1	**-1**	**30**	**9**	**22**	**27**	**34**	**25**
3	-1	**10**	**1**	**20**	**29**	**44**	**37**	**42**
4	2	**49**	**-1**	**11**	**8**	**19**	**-1**	**45**
5	-1	**0**	**5**	**48**	**13**	**16**	**41**	**38**
6	50	**3**	**12**	**7**	**18**	**39**	**46**	**15**
7	-1	**6**	**51**	**4**	**47**	**14**	**17**	**40**

1.1.4) Unsuccessful - p=0.8

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

1.1.5) Success - p=0.85

	0	1	2	3	4	5	6	7
0	52	9	54	-1	40	-1	26	23
1	-1	**2**	**51**	**10**	**27**	**24**	**39**	**-1**
2	8	**53**	**6**	**1**	**50**	**41**	**22**	**25**
3	3	**0**	**47**	**28**	**11**	**20**	**49**	**38**
4	46	**7**	**30**	**5**	**48**	**37**	**42**	**21**
5	31	**4**	**15**	**12**	**29**	**34**	**19**	**36**
6	14	**45**	**32**	**17**	**-1**	**43**	**-1**	**-1**
7	-1	**16**	**13**	**44**	**33**	**18**	**35**	**-1**

1.1.6) Unsuccessful - p=0.85

	0	1	2	3	4	5	6	7
0	-1	-1	39	36	15	18	41	-1
1	10	**37**	**12**	**19**	**40**	**35**	**14**	**17**
2	5	**28**	**9**	**38**	**13**	**16**	**21**	**42**
3	-1	**11**	**4**	**29**	**20**	**43**	**34**	**-1**
4	3	**6**	**27**	**8**	**1**	**30**	**47**	**22**
5	26	**-1**	**2**	**51**	**48**	**23**	**44**	**33**
6	-1	**50**	**7**	**24**	**31**	**0**	**53**	**46**
7	-1	**25**	**-1**	**49**	**52**	**45**	**32**	**-1**

1.2) Fill the table and comment on it

1.2.1)

p	Number of Success	Number of Trials	Probability	Total Time of Execution
0.7	16012	100000	0.160	23.6
0.8	2230	100000	0.022	23.7
0.85	541	100000	0.005	24.0

1.2.1) Comments

Also answer these questions while commenting

1- How do changes on p affect total success probability and total execution time?

Answer: As p increases, since more squares are needed to be visited for success case, probability of success decreases. In terms of total execution time, as p increases, total execution time also increases. Since algorithm works randomly, total execution time is affected by the number of sufficient squares to be able to say a case is successful. For example, for lower values of p, lower number of squares are necessary for success case, and so it terminates earlier.

2- Define trade-offs of the algorithm.

Answer: As p increases, algorithm gives more correct probability values. However, in that case total execution time slightly increases. So, there is a tradeoff between selected p value and total execution time.

PART 2

2.1) Fill the tables and comment on them

2.1.1) p = 0.7

k	Number of Success	Number of Trials	Probability	Total Time
0	100000	100000	1	11.1

2	100000	100000	1	12.8
3	99942	100000	0.999	13.1

2.1.2) p = 0.8

k	Number of Success	Number of Trials	Probability	Total Time
0	100000	100000	1	13.6
2	100000	100000	1	15.1
3	99939	100000	0.999	15.5

2.1.3) p = 0.85

k	Number of Success	Number of Trials	Probability	Total Time
0	100000	100000	1	17.4
2	100000	100000	1	19.0
3	99929	100000	0.999	19.2

2.1.4) Comments

Also answer these questions while commenting

1- How does total time change with k?

Answer: Increasing k leads to a rise in total time, but the relationship is not linear. The time increment tends to slow down with larger k values. This may be due to the fact that there are fewer backtracking options as k increases.

2- How do total time change with p for a specific k value? How does this change different from the first part?

Answer: Increasing p leads to a rise in total time, but the relationship is again not linear. The increase gets faster as we increase p. Which is different from the first part.

3- Run this algorithm for each p value for k a value larger than 10 multiple times. What are your thoughts?

Answer: The success rates declined for every p value. This happens because we don't allow the algorithm to backtrack further than k steps. It might encounter a dead end in those k steps. So, there are less options to solve the algorithm as k increases.
Also, total time decreased for p=0.7 and increased for the rest of the p values. This happens because of the way that the algorithm is designed. The algorithm stops after finding 64*p moves. For p=0.7 we don't need to be very accurate with our moves and making more random moves results in faster execution time. For other p values, since the success rates declined, we can say that the algorithm failed more. In those failed attempts it tried every move it could and couldn't terminate as fast as the previous examples. This resulted in an increase in total time.

PART 3

In this part, you will compare Part1 and Part2 algorithms according to their ability to solve the actual Knight's Problem where p=1.

- Run Part1 algorithm with p=1.
- Run Part2 algorithm with p=1 and k=0.
- Run Part2 algorithm with p=1 and a k value you think will work well.

Clearly state your findings and comment on them. When would you choose Part1 algorithm and when would you choose the other?

3.1) Findings

Part1 (Random Approach) with p=1 case:

- Execution time: 23.8
- Number of successful tours: 0
- Number of trials: 100000
- Probability of a successful tour: 0.0

Part2 (Random + Backtracking) with p=1 and k=0 case:

- Runs for a very long time so there are no results to show.

Part2 (Random + Backtracking) with p=1 and k=37 case:

- Execution time: 258.2

- Number of successful tours: 9
- Number of trials: 100000
- Probability of a successful tour: 0.00009

3.2) Comments

Part1 (Random Approach) with p=1:

When running Part1 of the algorithm with a success threshold (p) of 1, it becomes evident that purely random moves are ineffective for achieving a complete Knight's Tour. Despite 100,000 attempts, the algorithm failed to complete a full tour even once. This outcome shows the low probability of success associated with random movements in the Knight's Tour problem. However, this approach is significantly faster in terms of execution time compared to the more complex algorithm in Part2.

Part2 (Random + Backtracking) with p=1 and k=0:

Implementing Part2 with p=1 and k=0 aims for a completely deterministic solution, seeking to cover every square of the chessboard. This setting, however, results in an extremely time-consuming process. When we ran the algorithm, it ran for over 8 hours without reaching termination, indicating that it's impractically slow. This suggests that while the algorithm is thorough, its use is limited by its impractical execution time in scenarios demanding a complete solution.

Part2 with p=1 and k=37:

By adjusting Part2 to start with 37 random moves (k=37) before employing backtracking, the algorithm demonstrates a balanced approach between randomness and systematic search. This strategy leads to a higher number of successful tours compared to the first scenario at a slower pace.
The trade-off here is the following: increasing the number of random moves (k) tends to speed up the process but at the expense of thoroughness, resulting in less comprehensive solutions. Conversely, reducing k enhances the likelihood of a complete solution, but significantly increases computation time.

When to Choose Part1 vs. Part2:

Choose the Part1 algorithm for scenarios where speed is crucial, and a partially complete solution (with p significantly less than 1) is acceptable. This approach is well-suited for instances where a quick and less thorough solution is sufficient.

Choose the Part2 algorithm when the priority is to find more complete solutions (with p close to 1), and the time it takes to arrive at these solutions is a secondary concern. This method is ideal for situations where the quality and comprehensiveness of the solution outweigh the need for speed.