Ans 1: Abstraction of the program nrooks.py includes:

1. Initial State: Initial state is the state from where no rooks are placed anywhere.
   In the board, all the position contains 0
   S_0 = [[0]*N]*N
   [[0,0,…N],[0,0,….N]…….Ntimes]
2. Set of valid states- Any arrangement of N rooks and an empty board on N*N chessboard such that no rooks attack each other.
3. The successor function: The successor function consists of the states that can be formed by placing 1st rook in the initial state and then based on this successor function, the positions where 2nd rook can be placed and so on..
4. Goal state definition: Goal state definition is the set of all the possible states which accomplishes the condition where N number of rooks are placed at N*N chessboard without attacking each other.
5. Cost function: Cost function is the function which tells us the least number of moves required to accomplish the goal according to the given N.

Ans 2: A more efficient successor function called successors2() which has been used in the nrooks.py file is as follows:

```
def successors2(board):
    for r in range(0,N):
        for c in range(0,N):
            if count_pieces(board)<N and count_on_row(board,r)<1 and
count_on_col(board,c)<1:
                return [ add_piece(board, r, c) ]
```

(the indent is not same as in the program. It got changed while copying from the editor)

Ans 3- Code for Bfs for nrooks has been implemented in the file – nrooks.py

A separate python file for this code has been attached .

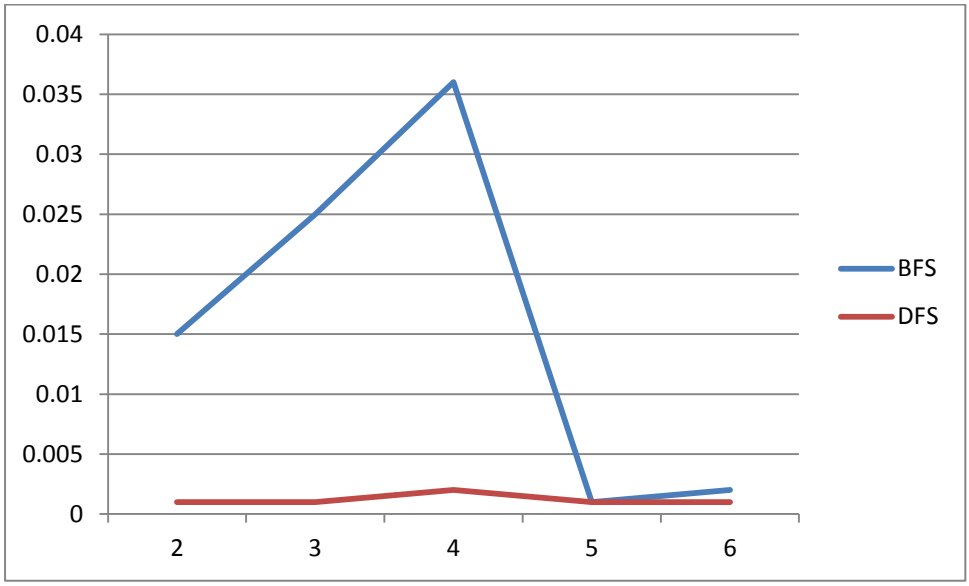
Ans 4: table showing how running time varies with N:

N	BFS	DFS
2	0.015	0.001
3	0.025	0.001
4	0.036	0.002
5	0.001	0.001
6	0.002	0.001

Running time is in milliseconds.

The two are different because as we can see in the table, running time of DFS is less than DFS.

In BFS, each node is traversed until the goal is reached whereas in DFS, it searches in the depth of the nodes where all the nodes need not be traversed.



Horizontal Axis – size of the board – N

Vertical Axis – time in milliseconds.