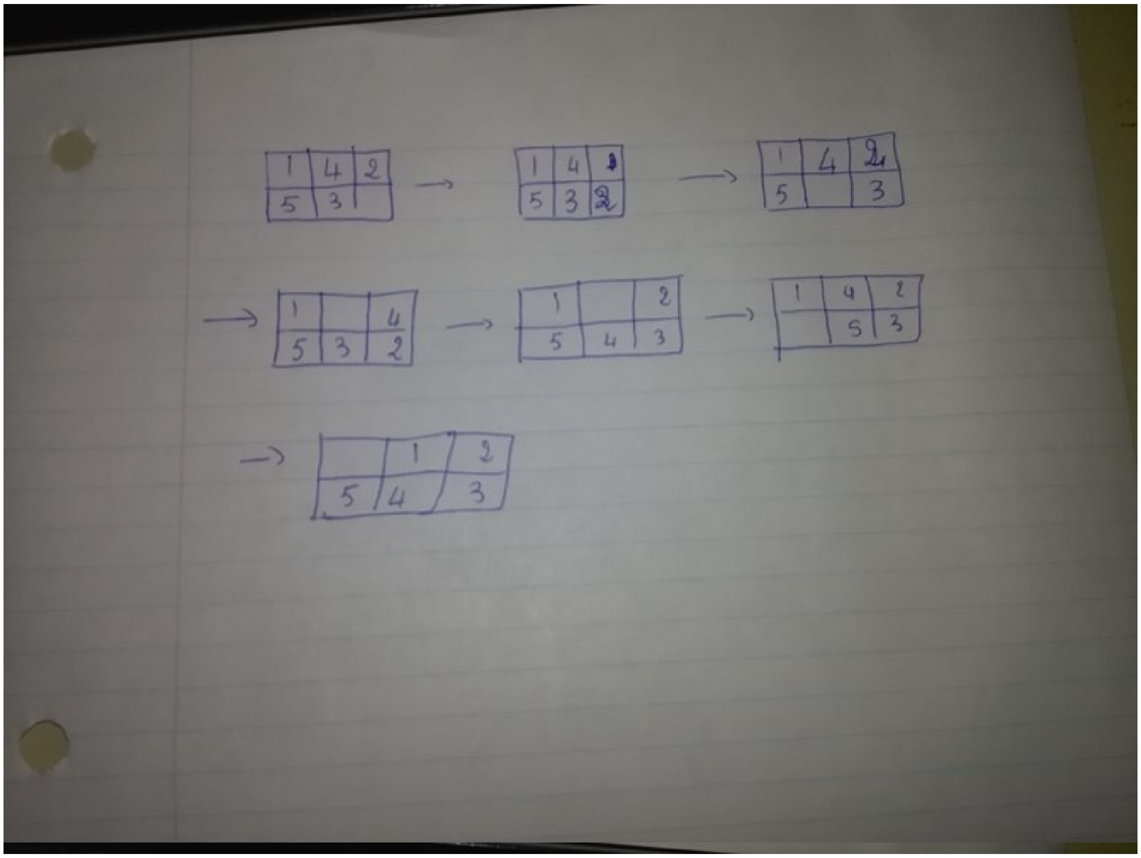
1. (a) i. Here is a tree of all the steps.



ii. Since all costs are the same, We get the same result as the bfs. (2)b)

iii.

iv.

(b)

(c)

(d)

2. (a) Suppose we have a domain where every state has a only one successor, the DFS will reach its goal in O(n) (n steps) where The ID will reach the goal in $O(n^2)$, (1+2+3+..+n).

(b) when when all costs are the same, (g(n) is just a multiple of depth(n)) uniform-cost and BFS will behave the same, in fact uniform-cost uses/reproduces BFS

(c) DFS is best-first search with f(n) = -depth(n)

(d) uniform cost search is just A* with h(n) = 0

3. (a) The code sent with this.

```
DX=0.01
[1.7400000000000013, 1.7400000000000007, 1.7399999999999998, 1.740000000000021, 3.960000000000001, 5.319999999999993, 6.3899999999999
92, 7.309999999999993, 8.119999999999997, 8.860000000000003, 10]

DX=0.02
[1.740000000000001, 1.7400000000000007, 1.7399999999999998, 1.7399999999999989, 3.96, 5.319999999999993, 6.379999999999992, 7.2999999
99999994, 8.119999999999997, 8.860000000000003, 10]

DX=0.03
[1.7400000000000013, 1.7500000000000007, 1.7299999999999998, 1.7400000000000062, 3.97, 5.330000000000003, 6.390000000000003, 7.300000
0000000025, 8.119999999999997, 8.850000000000003, 10]

DX=0.04
[1.7200000000000009, 1.7200000000000006, 1.7199999999999998, 1.7199999999999989, 3.96, 5.32, 6.4, 7.32, 8.119999999999997, 8.88000000
0000003, 10]

DX=0.05
[1.7500000000000009, 1.7500000000000007, 1.7499999999999998, 1.7500000000000033, 3.95, 5.299999999999999, 6.399999999999999, 7.299999
999999999, 8.100000000000001, 8.849999999999998, 10]

DX=0.06
[1.740000000000001, 1.7200000000000006, 1.7599999999999998, 1.7399999999999989, 3.94, 5.299999999999998, 6.359999999999998, 7.2999999
99999998, 8.120000000000001, 8.879999999999999, 10]

DX=0.07
[1.750000000000001, 1.7700000000000007, 1.7199999999999998, 1.740000000000002, 3.93, 5.350000000000001, 6.420000000000002, 7.28000000
0000001, 8.14, 8.86, 10]

DX=0.08
[1.7600000000000005, 1.7200000000000006, 1.7599999999999998, 1.7199999999999989, 3.92, 5.32, 6.4, 7.32, 8.16, 8.84, 10]

DX=0.09
[1.7100000000000004, 1.7200000000000006, 1.7299999999999998, 1.7400000000000013, 4, 5.359999999999999, 6.359999999999999, 7.27, 8.09,
 8.82, 10]

DX=0.1
[1.7000000000000004, 1.7000000000000006, 1.6999999999999997, 1.6999999999999988, 4, 5.299999999999999, 6.399999999999999, 7.299999999
999999, 8.1, 8.9, 10]
```

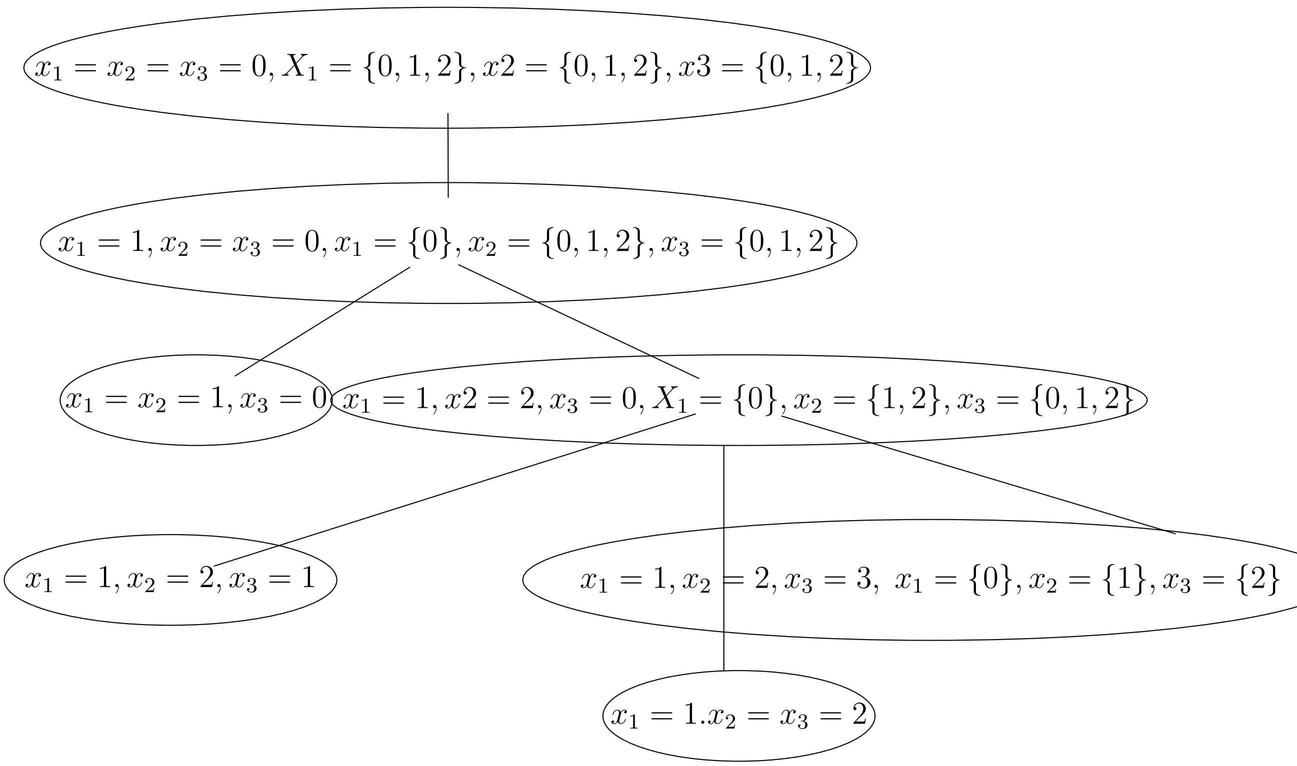
```
The steps of convergence are:
DX=0.01
[175, 75, 27, 127, 5, 33, 40, 32, 13, 15, 1]

DX=0.02
[88, 38, 14, 64, 3, 17, 20, 16, 7, 8, 1]

DX=0.03
[59, 26, 10, 43, 2, 12, 14, 11, 5, 6, 1]

DX=0.04
[44, 19, 8, 33, 2, 9, 11, 9, 4, 4, 1]

DX=0.05
[36, 16, 6, 26, 2, 7, 9, 7, 3, 4, 1]

DX=0.06
[30, 13, 5, 22, 2, 6, 7, 6, 3, 3, 1]

DX=0.07
[26, 12, 5, 19, 2, 6, 7, 5, 3, 3, 1]

DX=0.08
[23, 10, 4, 17, 2, 5, 6, 5, 3, 3, 1]

DX=0.09
[20, 9, 4, 15, 1, 5, 5, 4, 2, 3, 1]

DX=0.1
[18, 8, 4, 14, 1, 4, 5, 4, 2, 2, 1]
```

(b)

4. (a) Since no two rooks can attack each other, every rook must be on a different row and column. Let's assume they are on different colums and let x be the variable which is the row number (position). $S = \{x_1, x_2, .., x_k\}$ with $0 \leq x_i \leq n$ where

$x_i = 0$ if the rook i is not placed yet. Sot the constraint is that all $x_i \neq x_j$ for all $i \neq j$. (two diff rooks i and j can't be on the same row).

(b) here is the the search tree generated with applying backtracking search using backward checking for the problem with k = 3 and n = 3. (bottom right is the goal state.)



(c) here is the the search tree generated with applying backtracking search using forward checking for the problem with k = 3 and n = 3.

$$x_1 = x_2 = x_3 = 0X_1 = \{0, 1, 2\}, x_2 = \{0, 1, 2\}, x_3 = \{0, 1, 2\}$$

$$\downarrow$$

$$x_1 = 1, x_2 = x_3 = 0, x_1 = \{0\}, x_2 = \{1, 2\}, x_3 = \{1, 2\}$$

$$\downarrow$$

$$x_1 = 1, x_2 = 2, x_3 = 0, x_1 = \{0\}, x2 = \{1\}, x3 = \{2\}$$

$$\downarrow$$

$$x_1 = 1, x_2 = 2, x_3 = 3, x_1 = \{0\}, x_2 = \{1\}, x_3 = \{2\}$$