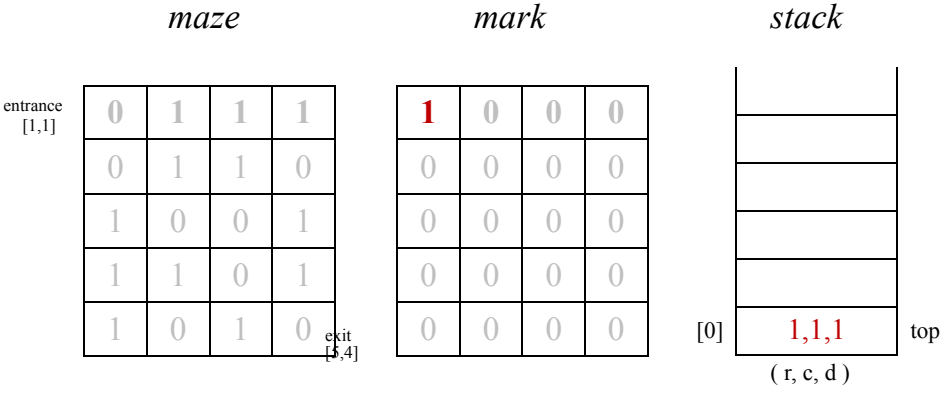
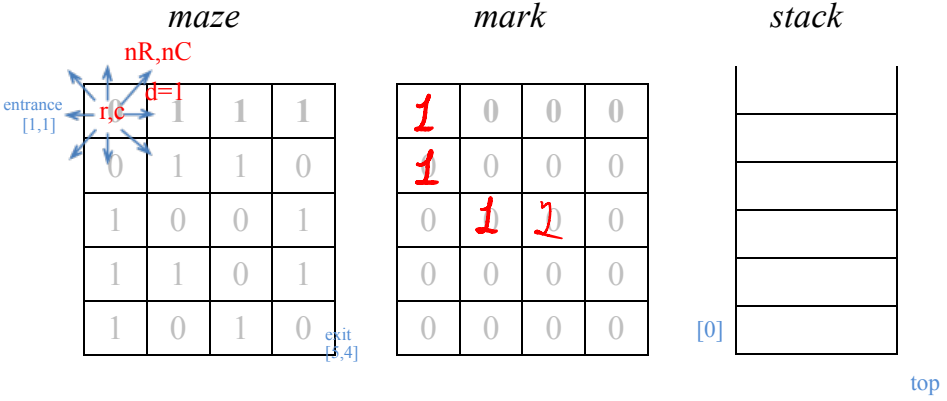
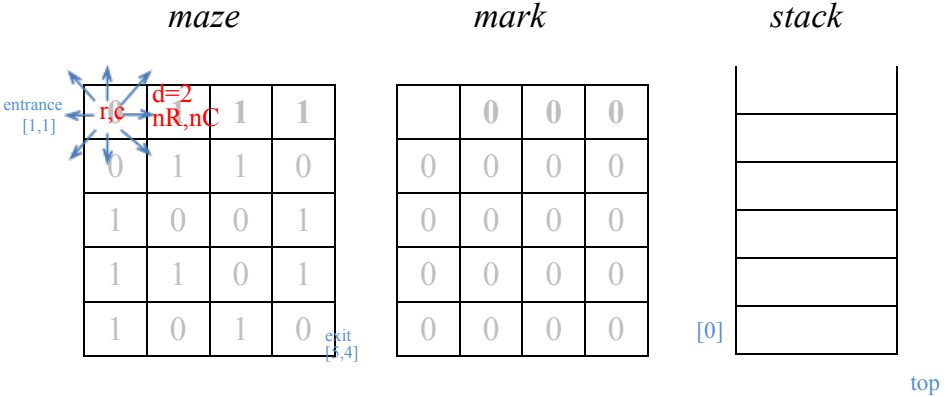
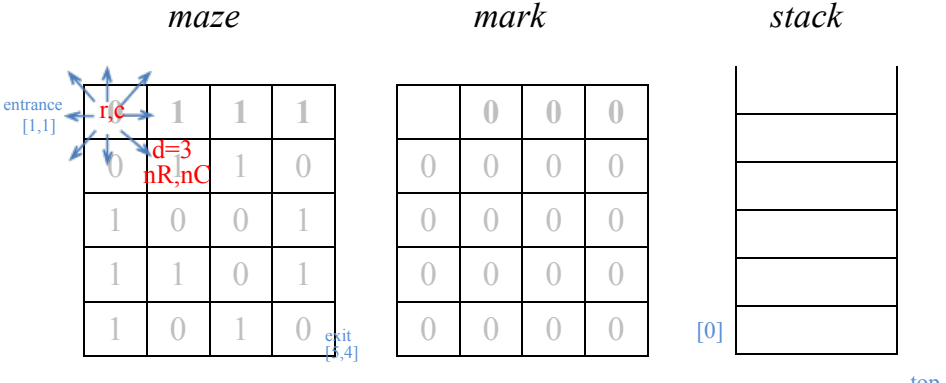
Ex: Maz

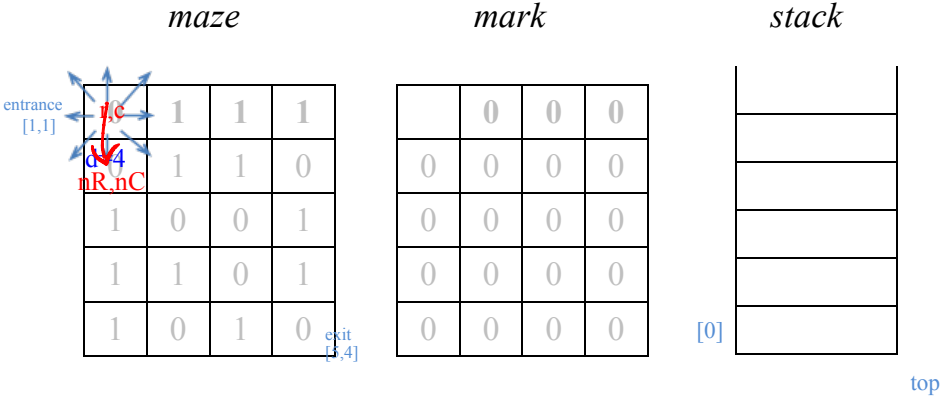


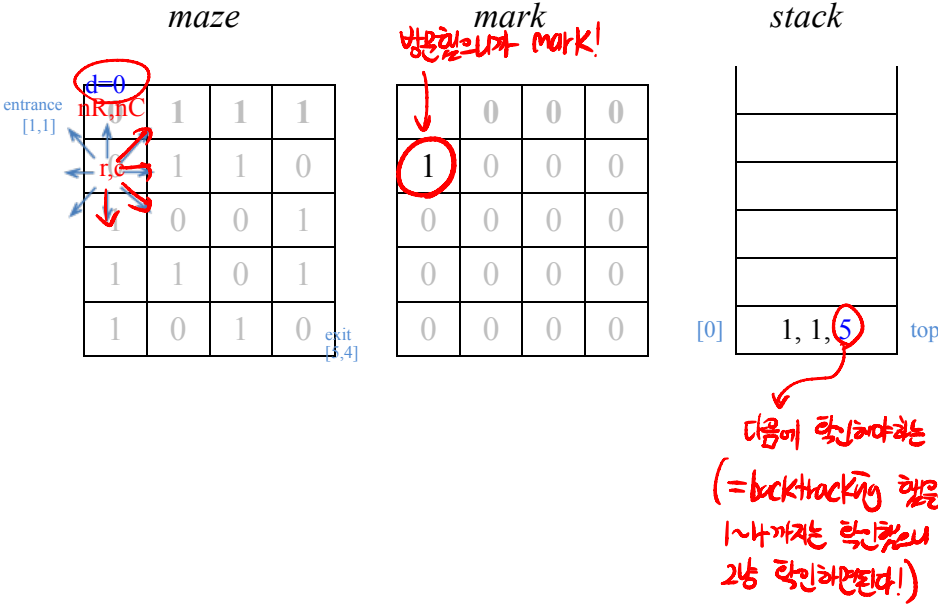

```
void path(void)   {
  int i, row, col, nextRow, nextCol, dir, found=FALSE;
  element position;
  mark[1][1]=1;   top=0;
  stack[0].row=1;   stack[0].col=1;   stack[0].dir=1;
```


maze
mark
stack
nR,nC
d=1
r,c
entrance
[1,1]
exit
[3,4]
[0]
top

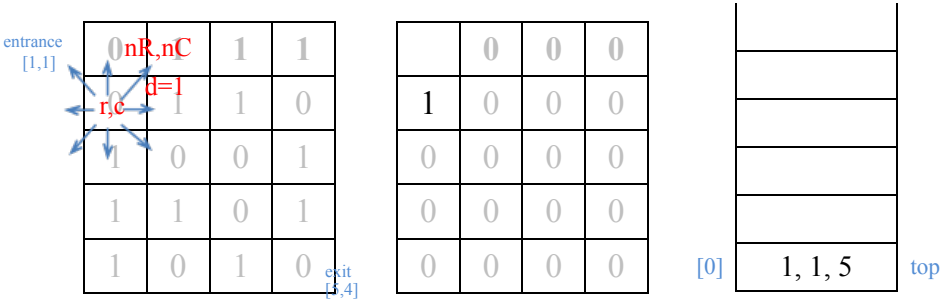

maze
mark
stack
entrance
[1,1]
r,c
d=2
nR,nC
1 1
0 1 1 0
1 0 0 1
1 1 0 1
1 0 1 0
exit
[5,4]
0 0 0
0 0 0 0
0 0 0 0
0 0 0 0
0 0 0 0
[0]
top


maze
mark
stack
entrance
[1,1]
r,c
d=3
nR,nC
1 1 1
0 1 0
1 0 0 1
1 1 0 1
1 0 1 0
exit
[3,4]
0 0 0
0 0 0 0
0 0 0 0
0 0 0 0
0 0 0 0
[0]
top

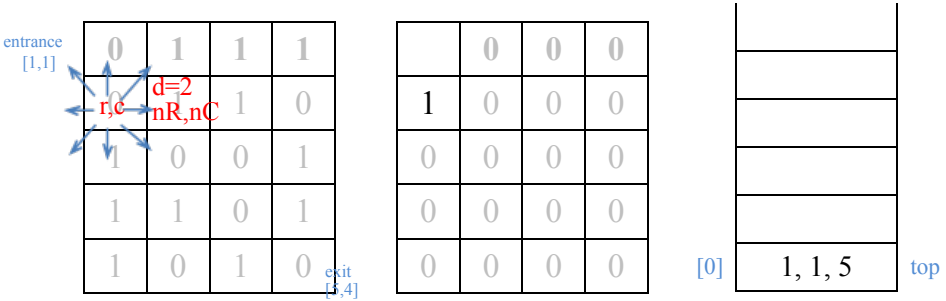

maze
mark
stack
entrance
[1,1]
r,c
d=4
nR,nC
exit
[3,4]
[0]
top


maze
mark
stack
방문했으니까 mark!
d=0
nR,nC
r,c
entrance
[1,1]
exit
[5,4]
1 1 1
1 1 0
0 0 1
1 1 0 1
1 0 1 0
0 0 0
1 0 0 0
0 0 0 0
0 0 0 0
0 0 0 0
[0]
1, 1, 5
top
다음에 확인해야하는 dir?
(=backtracking 했을때는
1~4까지는 확인했으니 5부터
2등 확인하면돼!)

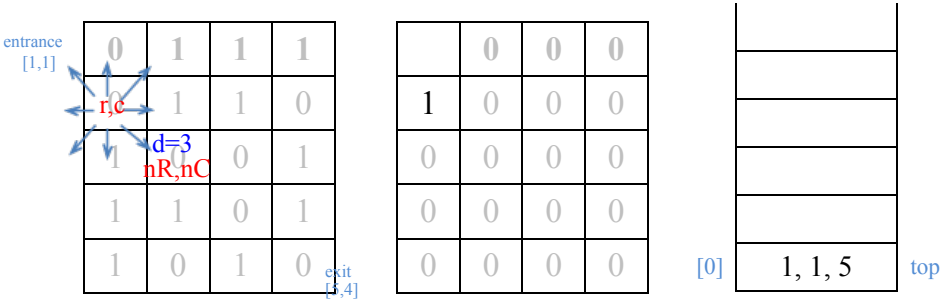
maze *mark* *stack*



maze *mark* *stack*



maze *mark* *stack*



maze

entrance
[1,1]

0	1	1	1
0	0	1	0
1	1	0	1
1	1	0	1
1	0	1	0

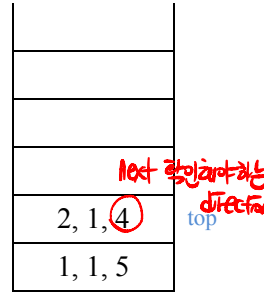
d=0
nR=nC
r,c

exit
[3,4]

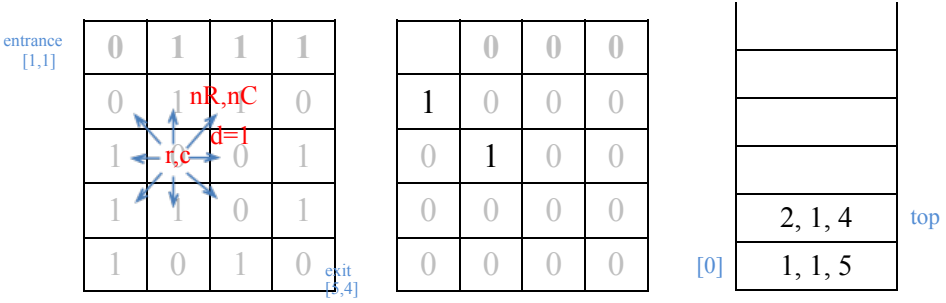
mark

	0	0	0
1	0	0	0
0	1	0	0
0	0	0	0
0	0	0	0

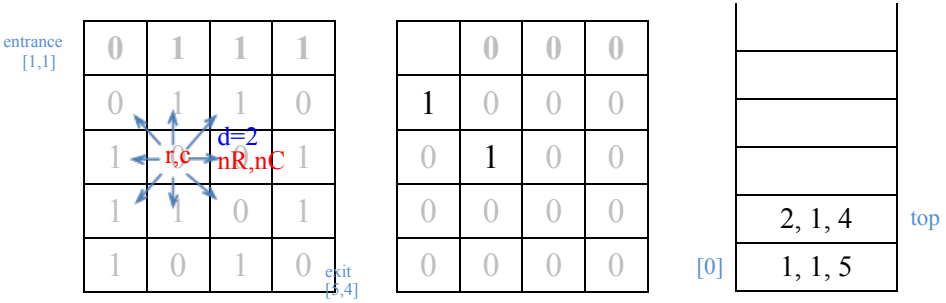
stack



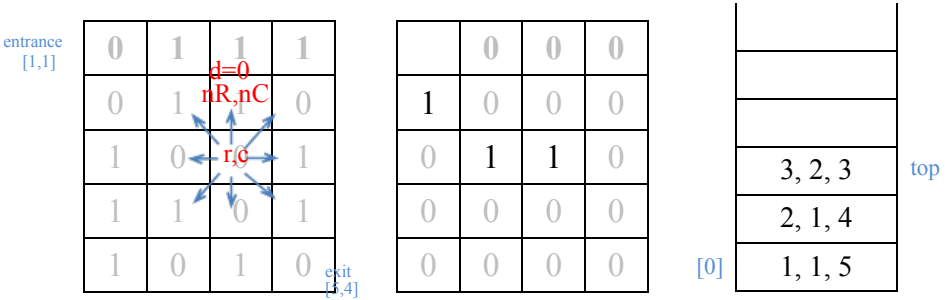
maze *mark* *stack*



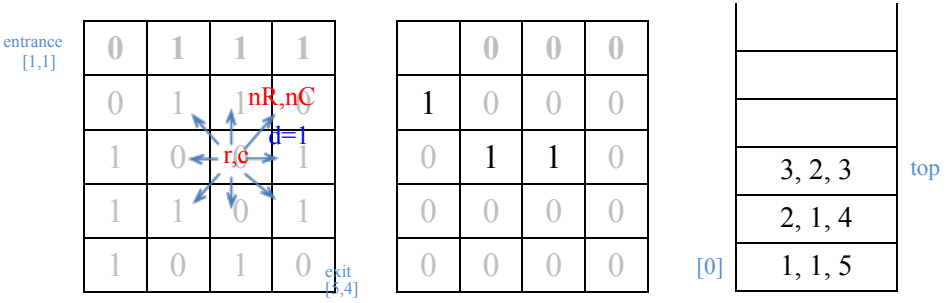
maze *mark* *stack*

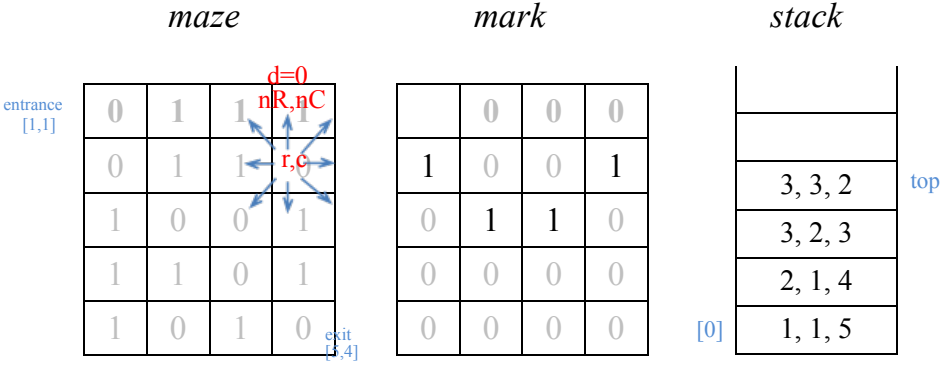


maze *mark* *stack*

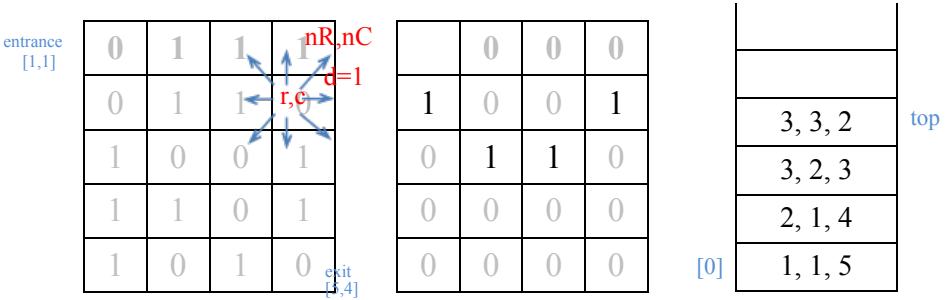


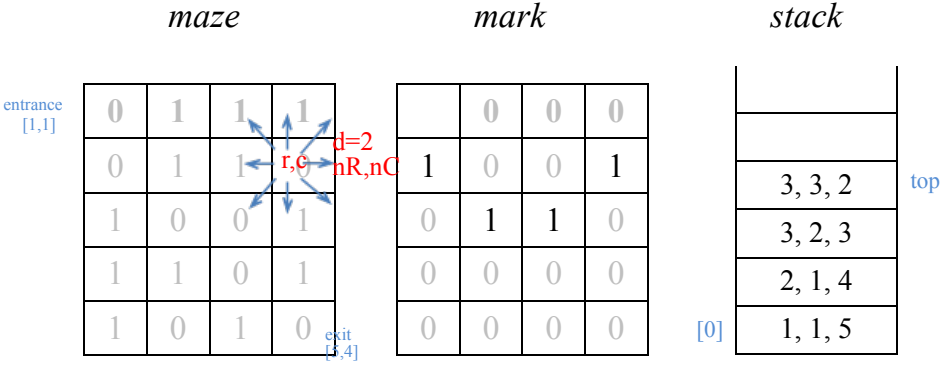
maze *mark* *stack*

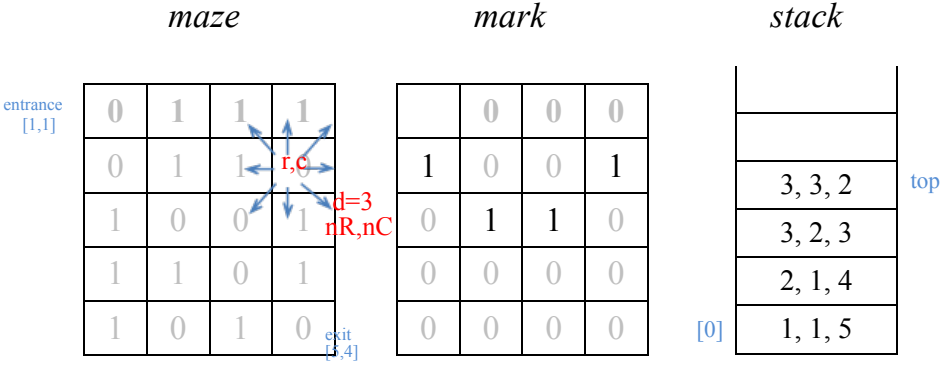


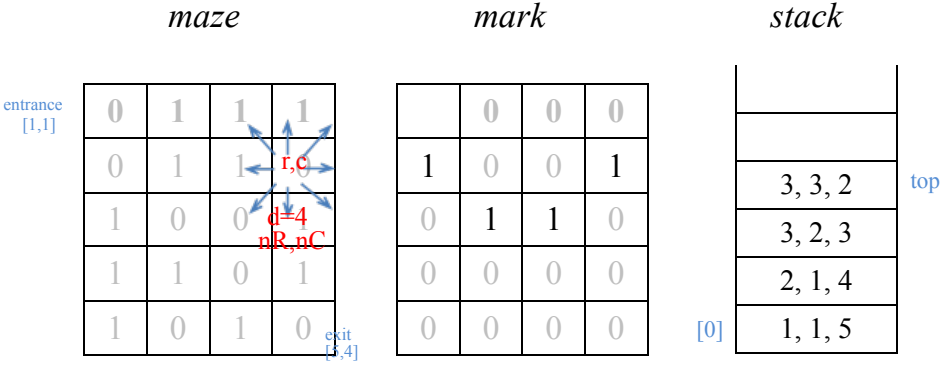

maze
mark
stack
entrance
[1,1]
d=0
nR,nC
r,c
exit
[3,4]
0 1 1 1
0 1 1 0
1 0 0 1
1 1 0 1
1 0 1 0
0 0 0
1 0 0 1
0 1 1 0
0 0 0 0
0 0 0 0
top
3, 3, 2
3, 2, 3
2, 1, 4
[0]
1, 1, 5

maze *mark* *stack*

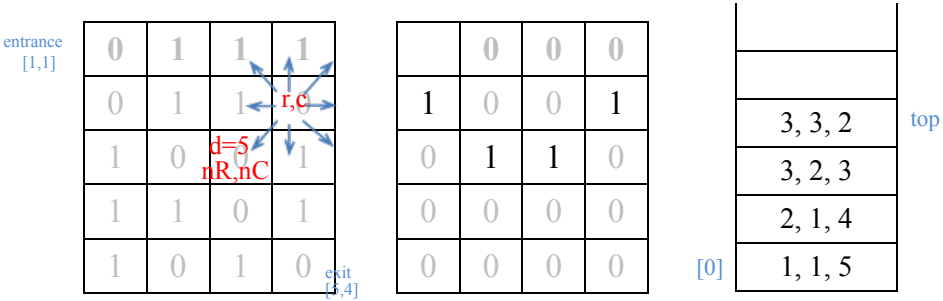



maze
mark
stack
entrance
[1,1]
0 1 1 1
0 1 1 0
1 0 0 1
1 1 0 1
1 0 1 0
exit
[3,4]
r,c
d=2
nR,nC
0 0 0
1 0 0 1
0 1 1 0
0 0 0 0
0 0 0 0
3, 3, 2
3, 2, 3
2, 1, 4
1, 1, 5
top
[0]

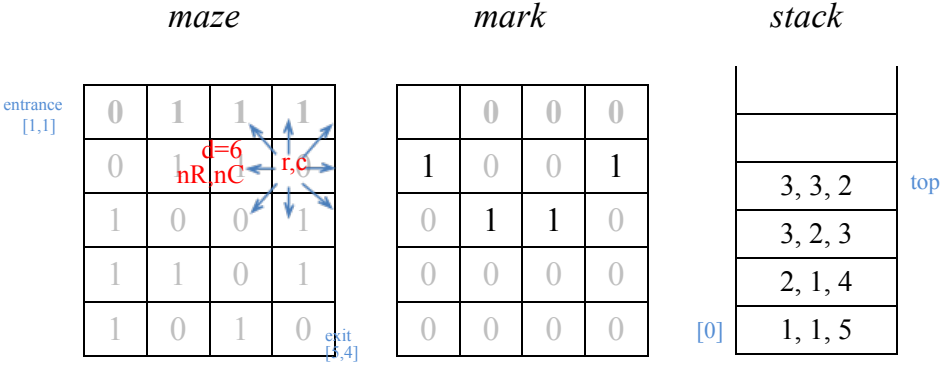

maze
mark
stack
entrance
[1,1]
0 1 1 1
0 1 1 0
1 0 0 1
1 1 0 1
1 0 1 0
exit
[3,4]
r,c
d=3
rR,nC
0 0 0
1 0 0 1
0 1 1 0
0 0 0 0
0 0 0 0
3, 3, 2
3, 2, 3
2, 1, 4
1, 1, 5
top
[0]


maze
mark
stack
entrance
[1,1]
0 1 1 1
0 1 1 0
1 0 0 1
1 1 0 1
1 0 1 0
r,c
d=4
nR,nC
exit
[3,4]
0 0 0
1 0 0 1
0 1 1 0
0 0 0 0
0 0 0 0
3, 3, 2
3, 2, 3
2, 1, 4
1, 1, 5
top
[0]

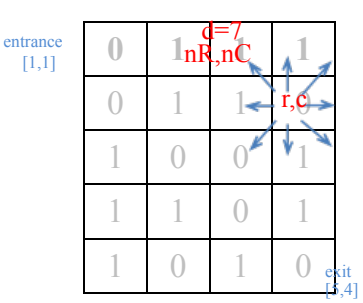
maze *mark* *stack*

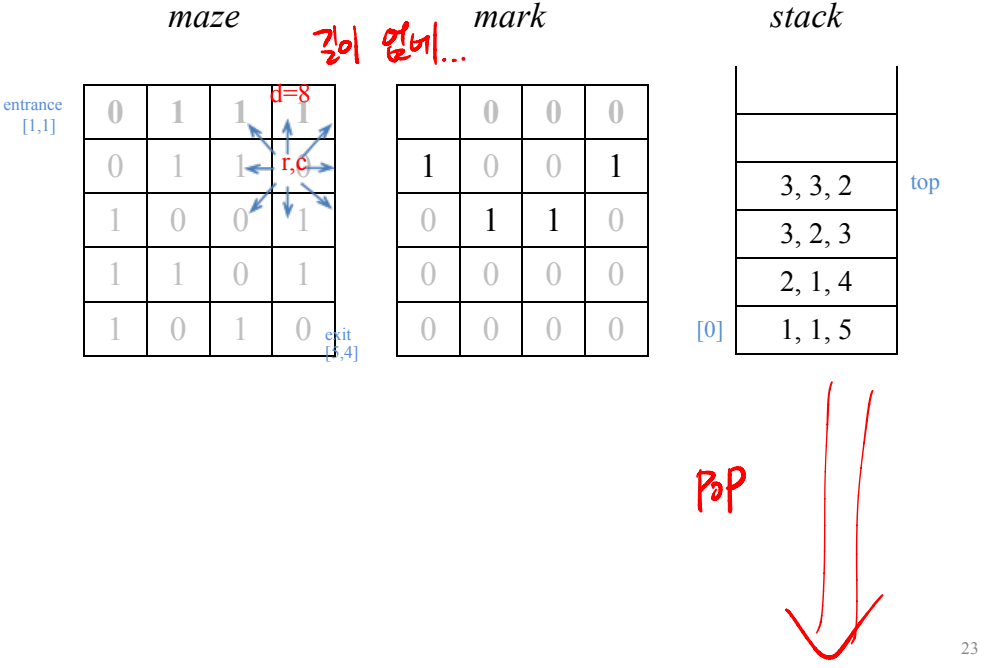


Maze Search : Example



maze *mark* *stack*

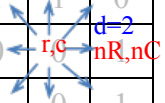



maze
mark
stack
길이 없네...
entrance
[1,1]
d=8
r,c
exit
[3,4]
top
[0]
3, 3, 2
3, 2, 3
2, 1, 4
1, 1, 5
pop

maze

entrance
[1,1]

0	1	1	1
0	1	1	0
1	0	0	0
1	1	0	1
1	0	1	0



exit
[3,4]

mark

	0	0	0
1	0	0	1
0	1	1	0
0	0	0	0
0	0	0	0

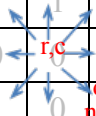
stack

3, 2, 3	top
2, 1, 4	
1, 1, 5	[0]

maze

entrance
[1,1]

0	1	1	1
0	1	1	0
1	0	0	1
1	1	0	0
1	0	1	0



exit
[3,4]

mark

	0	0	0
1	0	0	1
0	1	1	0
0	0	0	0
0	0	0	0

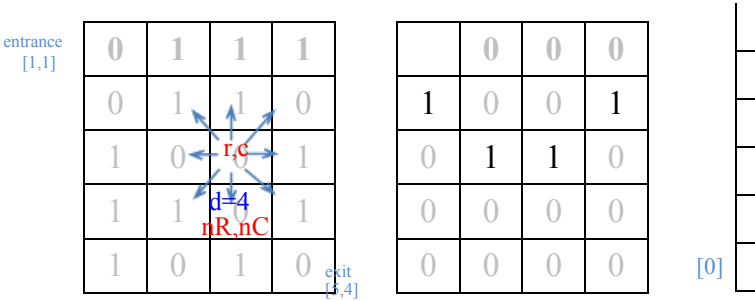
stack

3, 2, 3
2, 1, 4
1, 1, 5

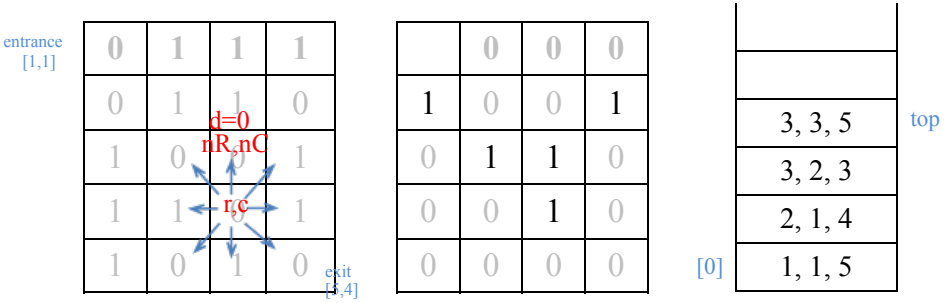
top

[0]

maze *mark* *stack*



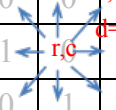
maze *mark* *stack*



maze

entrance
[1,1]

0	1	1	1
0	1	1	0
1	0	0	1
1	1	0	1
1	0	1	0



exit
[3,4]

mark

	0	0	0
1	0	0	1
0	1	1	0
0	0	1	0
0	0	0	0

stack

3, 3, 5
3, 2, 3
2, 1, 4
1, 1, 5

top

[0]

maze

entrance
[1,1]

0	1	1	1
0	1	1	0
1	0	0	1
1	1	r,c	d=2 nR,nC
1	0	1	0

exit
[3,4]

mark

	0	0	0
1	0	0	1
0	1	1	0
0	0	1	0
0	0	0	0

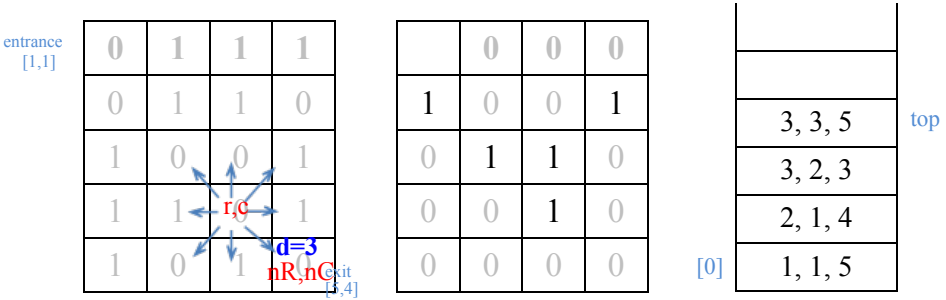
stack

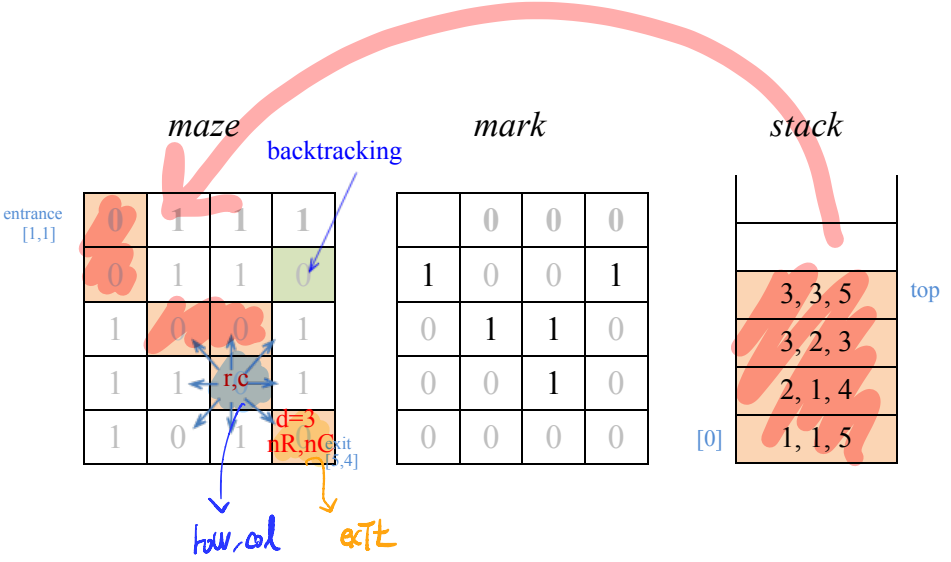
top

3, 3, 5
3, 2, 3
2, 1, 4
1, 1, 5

[0]

maze *mark* *stack*




maze
mark
stack
backtracking
entrance
[1,1]
0 1 1 1
0 1 1 0
1 0 0 1
1 1 r,c 1
1 0 1
d=3
nR,nC
exit
[5,4]
how, col
exit
1 0 0 1
0 1 1 0
0 0 1 0
0 0 0 0
3, 3, 5
3, 2, 3
2, 1, 4
1, 1, 5
top
[0]