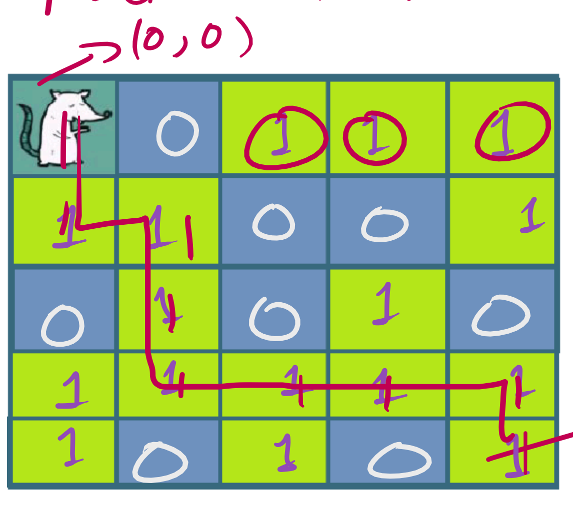
Applications of Recursion :

Backtracking — Dynamic Programming

* NQueens → VTU → easy
* Rat In A Maze ✗✗✗✗ } Hard
* Sudoku Solves ✗✗✗✗ }
* Subsets of an array (Power Set) $n \to 2^n$
* Subsequences of a string
* Phone keypad problem

Rat In A Maze (Recursion)

→ (0,0)



Dark → Unsafe walls

Green → Safe path

→ cheese (dest^n)

→ $(n-1, n-1)$

2D Matrix : $n \times n$

* → Forward Traversal $(x+1, y)$
* → Downward Traversal $(x, y+1)$

stand move

→ cell → (x, y)

$(x, y) = 1$ 0

* Introduction to Greedy Algos :→

* Minimum number of coins
* Activity Selection Problem
* Minimum cost of ropes
* Chocolate Distribution Problem
* Police men & Thieves
* Minimum number of Stairs Problem
* Job Scheduling Problem
* Fractional Knapsack
* 0 - 1 Knapsack
* Huffman Encoding
* Nikunj & Donuts

int[] coins = (min / max)
{ 1, 2, 5, 10, 20, 50, 100, 500, 2000 }

int V = 91; (4)

V - coins[i] >= 0

50	91 − 50 = 41	→	1
20	41 − 20 = 21	→	2
20	21 − 20 = 1	→	3
1	1 − 1 = 0	→	4

91

31

Activity Selection Problem :→

Given a set of activities with their start & finish times, select the maximum number of activities that can be performed by a single person, assuming that a person can only perform a single activity at a given time.

(1, 4), (5, 7), (8, 9)

Activity	Start	Finish	Sort(f)	O/P
A1	5	7	A3(1,4)	A3(1,4)
A2	8	9	A6(3,5)	A1(5,7)
A3	1	4	A5(0,6)	A2(8,9)
A4	5	9	A1(5,7)	
A5	0	6	A2(8,9)	
A6	3	5	A4(5,9)	

* Sliding Window

Chocolate Distribution

→ 7, 3, 2, 4, 9, 12, 56

$i = 0,\ j = m - 1 = 2$

$m = 3$

Step 1 →

Sort → 2, 3, 4, 7, 9, 12, 56

0 1 2 3 4 5 6

mini = 2

$m_x - m_n$

4 − 2 = 2

7 − 3 = 4

9 − 4 = 5

12 − 7 = 5

56 − 9 → too large