CLASS 76

04 May 2023 21:10

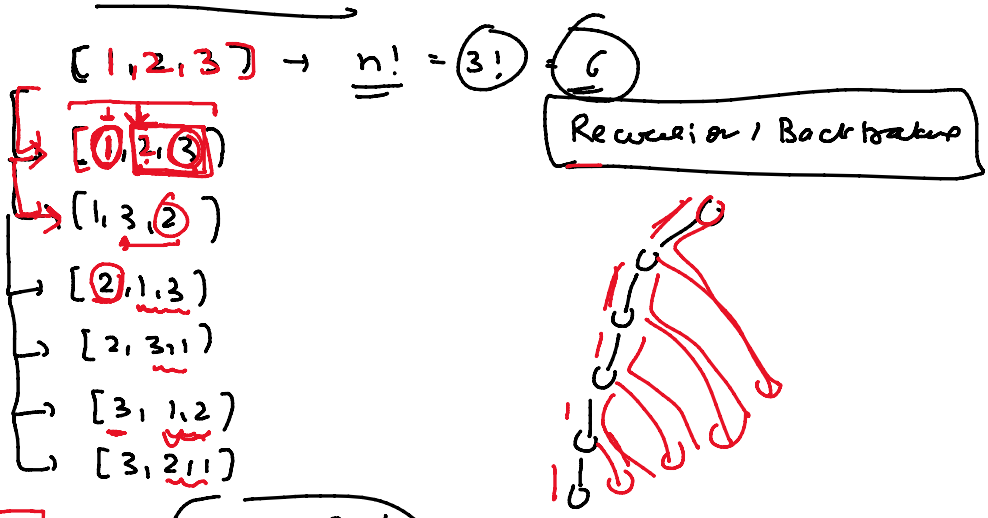
Class 76

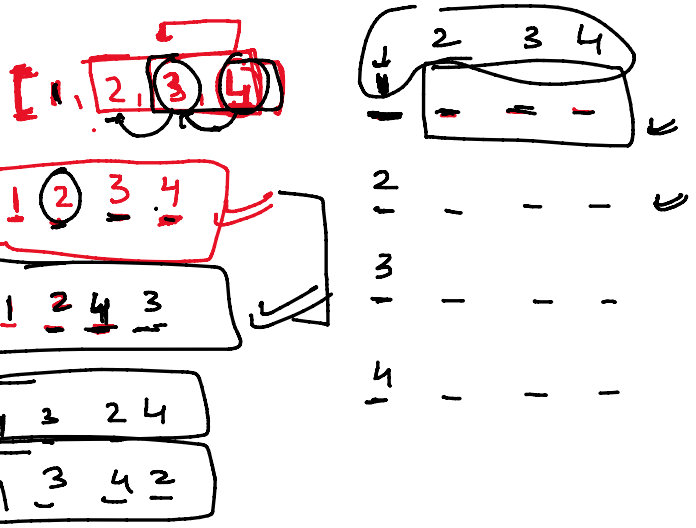
Backtracking

Permutation I, II

Permutation I

[1,2,3] → n! = 3! = 6





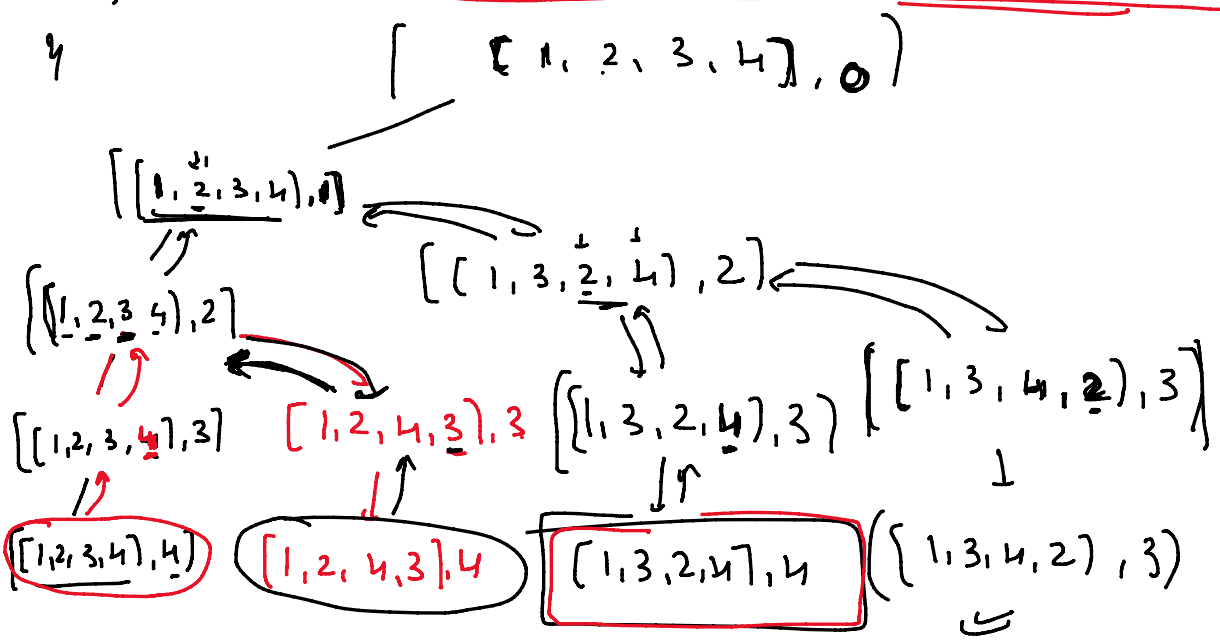

```
vector<vector<int>> Permute(vector<int>& nums
{
    vector<vector<int>> ans;
    vector<int> temp;
    int i,j,k,n;
    n = nums.size();
    Per(ans, nums, 0);
}
```

```
void Per (vector<vector<int>>&ans, vector<int>nums, int idx)
{   int n=nums.size();
    if(idx==n)
    {  ans.push_back(nums);
       return;
    }
    For (i=idx; i<n; i++)
    { swap(nums[idx], nums[i]);
      Per(ans, nums, idx+1);
      swap(nums[idx], nums[i]);
    }
}
```

[1 2 3]

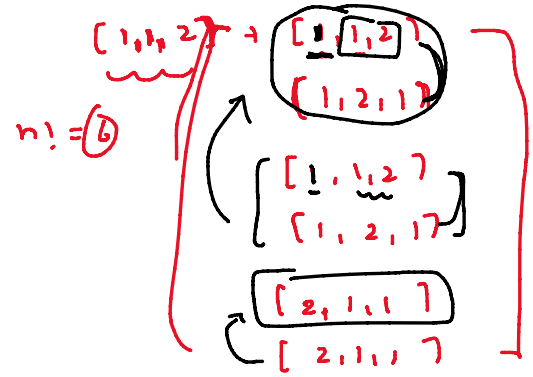
temp.push_back()

temp.pop_back()



Set

Permutation II



n! = 6

{1,1,2,4}

3 permutations

1 {1,2} [1,2] [2,1]

2 {1,2,4} [1,2] [2,1]

```
void Permu (vector<int> &nums, vector<vector<int>> &ans, int idx)
{
    int n = nums.size();

    if (idx == n)
    {
        ans.pb(nums);
        return;
    }
    map<int,int> ma;
    for (i = idx; i < n; i++)
    {
        if (ma.count(nums[i]) == 0)
        {
            swap(nums[i], nums[idx]);
            Permu(nums, idx+1);
            swap(nums[i], nums[idx]);
        }
        ma[nums[i]] = 1;
    }
}
```

local to a function

[1, 1, 2] {1, 1, 2}

[1, 2, 1]

[1, 1, 2]

ma → [1]

(0 – 2) (3 – 5) (6 – 8) (1, 1)

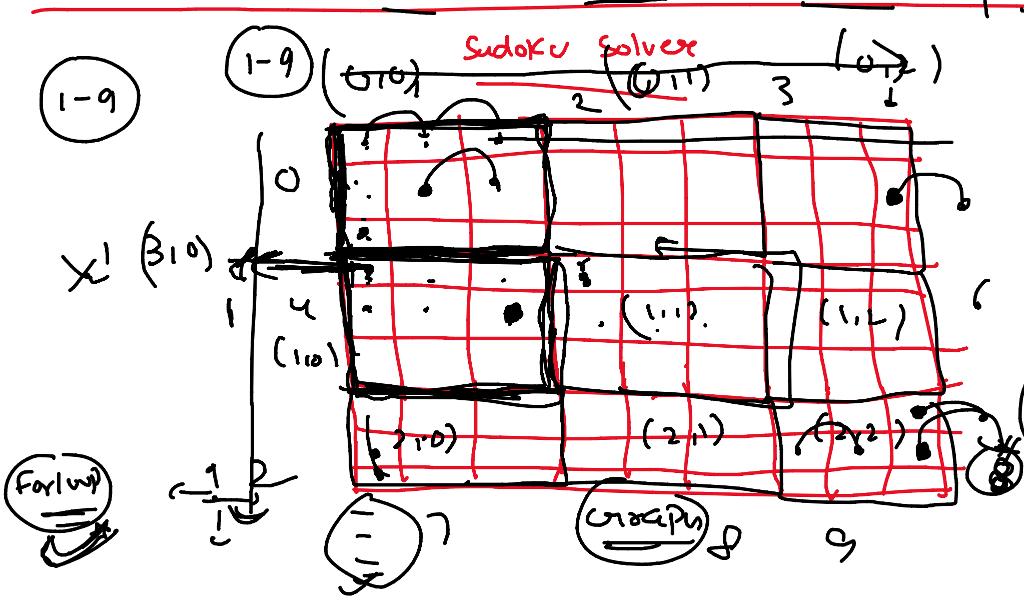
(0 – 2) (3 – 5) (6 – 8) (3, 3)

x, y

x/=3

y/=3

Sudoku solver



(1, 0)

4/ = 3 = 1

(4, 2)

2/3 = 0

2/3 = 0

4/3 = 1

1*3 = 3

0*3 = 0

(i, j+1) (i+1, 0)

```
bool solve (vector<vector<char>> &board, int i, int j)
{
    int nx = i;
    int ny = j+1;
```

(i, j+1)

(i, j)

(i, j+1)

```
int ny = j+1;
if (i==9 and j==0)
    return true;
if (j==8)
{ nx = nx+1;
  ny = 0;
}
if (board[i][j] != '.')
{ return solve(board, nx, ny); }
for (k=1; k<=9; k++)
{ board[i][j] = (k + '0');
  if (is_valid(board, i, j))
  { if (solve(board, nx, ny))
      return true;
  }
  board[i][j] = '.';
}
return false;
}
bool is_valid(vector<vector<char>>& board, int i, int j)
{
  //row
  for (k=0; k<9; k++)
  { if (k != j and board[i][k] == board[i][j])
      return false;
  }
```

(i+1, 0)

(i, j+1)

```
// column
for(k=0; k<9; k++)
{ if (k!=i and board[k][j] == board[i][j])
        return false;
}

// 3*3 Matrix
int x= i/3;
int y= j/3;

x*= 3;
y*= 3;

for(k=x; k<x+3; k++)
{ for(z=y; z<y+3; z++)
  { if(k!=i and z!=j) and board[k][z] ==
                              board[i][j])
    return false;
  }
}
return true;
}
```