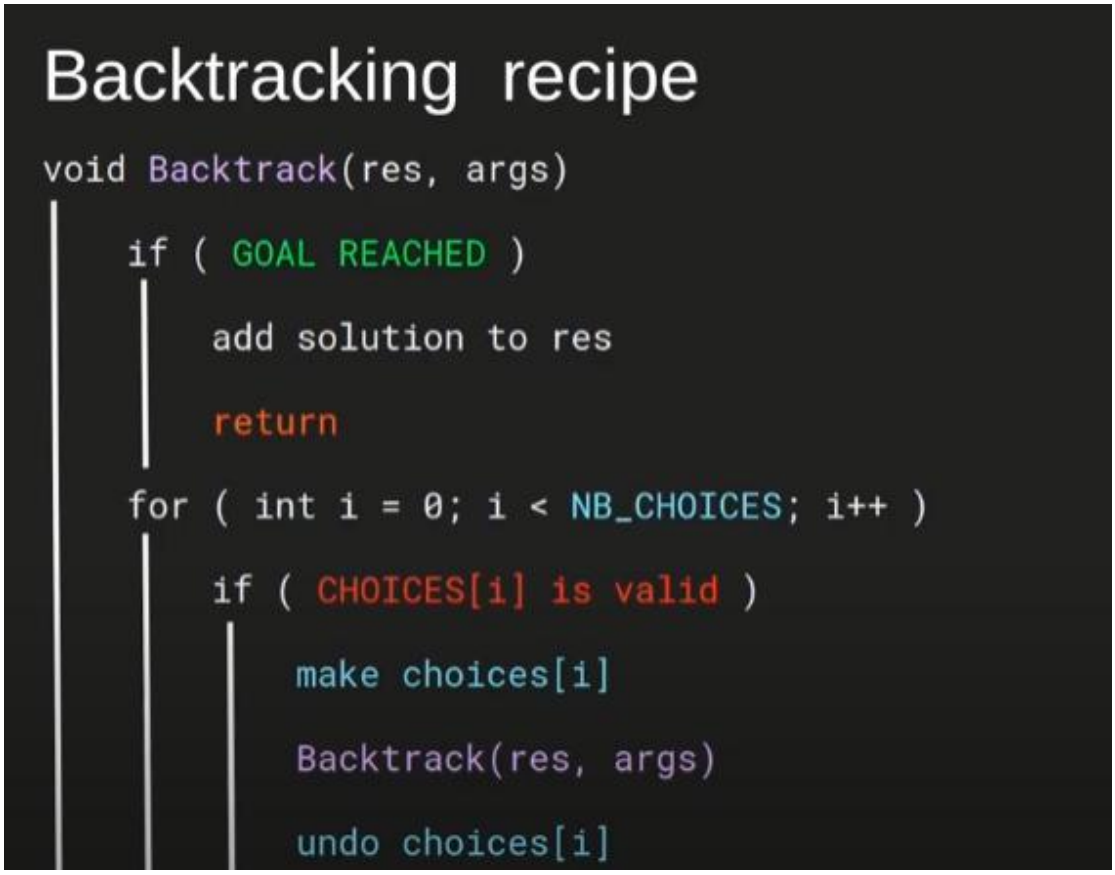
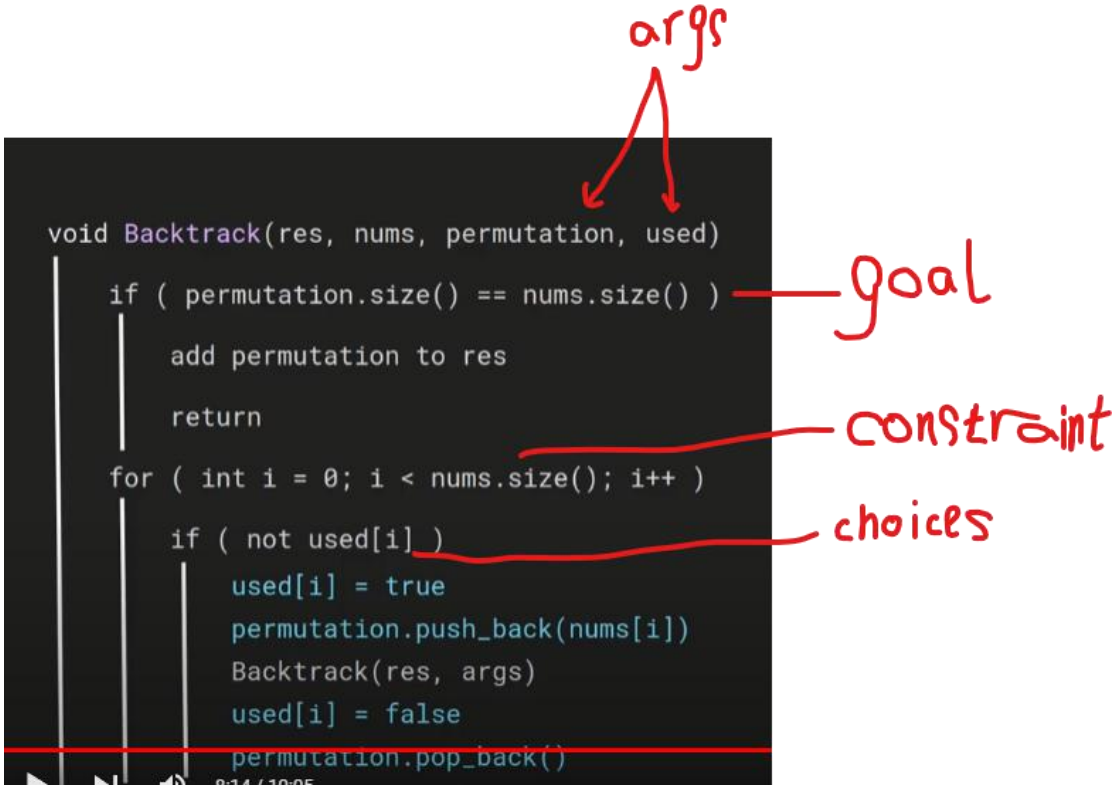
Backtracking Receipt

1. Choices
2. Constraints
3. Goal



```
public class Solution {
    public IList < IList < int >> Permute(int[] nums) {
        List < IList < int >> res = new List < IList < int >> ();
            if (nums == null || nums.Length == 0)
                return res;
            bool[] visited = new bool[nums.Length];
            for (int i = 0; i < nums.Length; i++)
            {
                visited[i] = false;
            }
            dfs(nums, res, new Stack < int > (), visited);
            return res;
    }

    private static void dfs(int[] nums, List < IList < int >> res, Stack < int > subset, bool[] visited)
        {
            if(subset.Count == nums.Length)
            {
                res.Add(subset.ToArray());
                return;
            }

            for (int i = 0; i < nums.Length; i++)
            {
                if (visited[i]) continue;
                subset.Push(nums[i]);
                visited[i] = true;
                dfs(nums, res, subset, visited);
                subset.Pop();
                visited[i] = false;
            }
        }
}
```