BACKTRACKING EXERCISES

For the following questions, you may write any additional functions if needed. However, the main function should be written using recursion.

1. **All Subsets:** Write a function to print all subsets of a given set of strings.
   ```
   void PrintSubsets(char* str_arr[], int n);
   ```
   E.g. str_arr = {"Tom", "Jerry", "Fred"}

   → Print to screen: {}, { Tom }, { Jerry }, { Fred }, { Tom, Jerry }, { Tom, Fred }, { Jerry, Fred }, { Tom, Jerry, Fred }

2. **k-Permutation:** Write a function to print all k-permutations (without repetition of each number) of given set of integers.
   ```
   void k_Permute(int arr[], int n, int k);
   ```
   Notes: k-Permutations (without repetition of each number) of array A are different ordered arrangements of all k-element subsets of A.

   For example: $A = \{ 4, 5, 1 \}, n = 3, k = 2$

   ➔ Print to screen: {4 5} {4 1} {5 4} {5 1} {1 4} {1 5}

   ➔ {4 4}, {5 5}, ... are not accepted since they duplicate number 4 and 5.

3. **All permutation:** Write a function to print all permutations of a given string.
   ```
   void PrintAllPermutation(char* str);
   ```
   For example: $S = abc$

   ➔ Print to screen: $abc, acb, bac, bca, cab, cba$

4. **Subset sum problem.**

   Given an array of non-negative integers A and a value s, write a function to print all subsets A whose sum is equal to s.
   ```
   void PrintAllSubsetsum(int arr[], int n, int s);
   ```
   E.g. $A = \{1,3,2,5,1,3\}, s = 5$

   → Print to screen: $\{1,3,1\}, \{5\}, \{3,2\}, \{2,3\}, \{1,1,3\}$

5. **Partition Sum:** Write a function to make a partition of a set of n elements into k subsets with equal sum and print the result to the screen. The function returns 0 if there is no solution, 1 otherwise.
   ```
   int Partition(int arr[], int n, int k);
   ```
   E.g. $A = \{2,1,4,5,6\}, n = 5, k = 3$

   → Print to screen: $\{2,4\}, \{1,5\}, \{6\}$

   $A = \{2,1,5,5,6\}, n = 5, k = 3$

   → Print to screen: $No\ solution$

6. **Combinational Sum.**
   Given an array of positive integers A and a value s, write a function to print all unique combinations in A where the sum is equal to s. (One element in A can be repeated unlimited times in a combination)
   E.g. $A = \{2,4,6,8\}, s = 8$
   → Print to screen: $\{2,2,2,2\}, \{2,2,4\}, \{2,6\}, \{4,4\}, \{8\}$
   $A = \{2,4,6,8\}, s = 1$
   → Print to screen: $No\ solution$

7. **Letter Case Permutation.**
   Given a string containing alphabetic characters, write a function to return all possible letter case permutations of the string (i.e., convert it to uppercase or lowercase).
   Example:
   Input: "a1b2"
   Output: "a1b2", "a1B2", "A1b2", "A1B2"

8. **Sudoku Solver:** Create a program to solve Sudoku puzzles using backtracking. Given a partially filled 9x9 grid, the goal is to fill the grid according to Sudoku rules.

9. **Remove invalid parentheses.**
   Given a string S that contains parentheses and letters, write a function to remove the minimum number of invalid parentheses and print all valid strings of S to the screen (if there are more than 1 solution).
   ```
   void PrintAllSubsetsum(int arr[], int n, int s);
   ```

   E.g. S =`()())()`
   → Print to screen: `()()(), (())()`