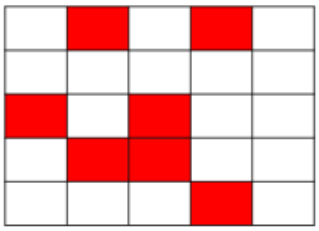
Lab 05 Exercises

1. Assume you are developing a program to simulate a game where players take turns to guess a number between 1 and 100. The program generates a random number between 1 and 100 at the start of the game, and the player who guesses the correct number wins the game. If a player guesses incorrectly, the program tells them whether their guess was too high or too low, and the turn passes to the next player. Solve it using Direct Recursion?
2. Write a C++ function to find the length of a singly linked list using tail recursion.
3. Create a C++ program to search for a value in a singly linked list, using non-tail recursion?
4. Imagine you are working on a scientific application that involves processing a multi-dimensional array. You need to implement a function that calculates the sum of all elements in this array. The array can contain nested arrays, forming a multi-dimensional structure represented as a jagged array. Write a C++ function int recursiveArraySum(int* arr[], int sizes[], int dim) that computes the sum of all the elements in such a multi-dimensional array. The sizes array contains the sizes of each dimension, and dim indicates the current dimension being processed.
5. Imagine you are given a grid representing a maze, where “0”s represent obstacles(red highlighted) or walls, and “1”s represent open paths or empty cells. You have two arrays at your disposal: an empty Solution array and a maze represented by the following grid:



   Your task is to guide a lion through a maze from the starting point at (0, 0) to reach a piece of meat located at the destination point (4, 4). The lion is not allowed to move diagonally and can only move through open paths (represented by '1's) while avoiding obstacles (represented by '0's). Please provide a C++ code that uses a backtracking algorithm to navigate the maze and move the lion to the meat. Additionally, display the contents of the Solution array after the lion has reached the meat.
6. Imagine a 4x4 grid where a person is on a mission to collect as many flags as possible. Each cell of the grid represents a potential location for placing a flag. However, there are constraints:
   i. Only one flag can be placed in each row or column.
   ii. Additionally, no two flags can ever be placed on the same diagonal (i.e., no two flags can threaten each other diagonally).
   iii. Your goal is to design a backtracking algorithm in C++ to help this person maximize the number of flags collected while adhering to these constraints. After implementing the algorithm, provide the code and report the maximum number of flags that can be placed on the 4x4 grid.