Lab 2: Backtracking algorithms

Q1. The n-queens problem.

Problem statement: Apply the Backtracking approach to place n chess queens on an $n \times n$ chessboard such that no two queens attack each other. Note that two queens can attack each other if they are on the same column, row, or diagonal.

Input: The user inputs an integer n from the keyboard, representing both the number of queens and the size of the chessboard.

Output: Print to the console a simple chessboard with n queens, none of which attacks the others.

Example of input and output:

Input from user keyboard	Output to console
4	- Q - - - - - Q Q - - - - - Q -

Q2. The knight tour problem.

Problem statement: A chess knight is placed on the first cell $\langle r_0, c_0 \rangle$ of an empty chessboard of the size $n \times n$. Apply the Backtracking approach to move the knight according to the rules of chess such that it visits each cell exactly once.

Input: Read the input data from a text file whose format is as follows

- 1st line: a positive integer n to indicate the size of the chessboard
- 2nd line: two non-negative integers representing the row index and column index, respectively. Indexes start from zero. The two numbers are separated by single space " ".

Output: If there exists a solution, print to the console a simple chessboard, each cell on the board shows the step i^{th} ($i \in [1,64]$) of the tour. Otherwise, output the string "No solution".

Example of input and output:

Input text file	Output to console
5 0 0	1 14 9 20 3 24 19 2 15 10 13 8 25 4 21 18 23 6 11 16 7 12 17 22 5

Q3. The maze problem.

Problem statement: A robot is placed at a certain position (starting position) in the maze of size $w \times h$, and it must navigate to reach another position (goal position). Positions in the maze will either be open or blocked with an obstacle. Of course, the robot can only move to positions without obstacles and must stay within the maze. At any given moment, the robot can only move 1 step in one of 4 directions: North, East, South, and West.

Apply the Backtracking approach to help the robot search for a path from the starting position to the goal position.

Input: Read the input data from a text file whose format is as follows

- 1st line: two positive integers, w and h, separated by single space " ", for the size of the maze
- The x-coordinate ranges in $[0, w-1]$, from left to right. The y-coordinate ranges in $[0, h-1]$, from top to bottom.
- 2nd line: two non-negative integers representing the x-coordinate and y-coordinate of the starting position, respectively. The two numbers are separated by single space " ".
- 3rd line: same format as in the above line, yet the data is for the goal position
- The following lines represents the maze, in which each square can be one of the following:
  - S: the starting position
  - G: the goal position
  - #: a square block with an obstacle
  - Empty: an open square

Output: If there exists a solution, print to the console the maze of format described above and mark the found path by putting an X to every square on the path. Otherwise, output the string "No solution".

Example of input and output:

Input text file	Output to console
6 7 1 0 5 4 # S # # # # # # # # # # # # G # # # # # #	# S # # # # X X # # X # # X X # # X # # X G X X X X # # # # # #

Q4. The sum of subsets problem.

Problem statement: Given a set of n distinct positive integers, $A = \{a_1, a_2, .., a_n\}$. Apply the Backtracking approach to find all subsets of A that sum to a positive integer k.

Input: Read the input data from a text file whose format is as follows

- 1st line: a positive integer n to indicate the size of the input array
- 2nd line: n integers, two consecutive numbers are separated by a single space " "
- 3rd line: a positive integer k

Output: If there exists a solution, print all the subsets to the console, one per line, and two consecutive numbers in a subset are separated by a single space " ". Otherwise, print the string "No solution".

Example of input and output:

Input text file	Output to console
4 3 5 6 7 15	3 5 7

Other requirements: You must implement both the non-upgraded and upgraded versions.

Q5. The Hamiltonian circuit problem.

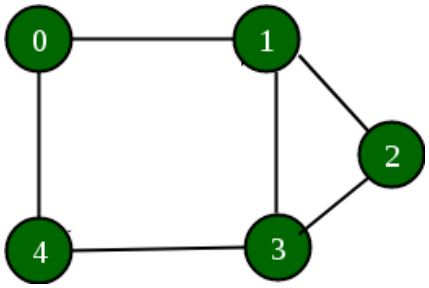
Problem statement: Consider the graph G of n vertices (for simplicity, undirected graph only). Apply the Backtracking approach to find a Hamiltonian cycle from G, starting from a vertex V.

Input: Read the input data from a text file whose format is as follows

- 1st line: a positive integer n to indicate the number of vertices.
- n next lines present the adjacency matrix, each of which has n integers. The value at row i and column j, $A[i, j]$, equals to 1 if there is an edge from i to j, and $A[i, j] = 0$ otherwise.
- The last line shows a non-negative integer k, representing the starting vertex.

Output: If there exists a solution, print the sequence of vertices forming the cycle (in their exact order) to the console. Otherwise, print the string "No solution".

Example of input and output:



Input text file	Output to console
5 0 1 0 0 1 1 0 1 1 0 0 1 0 1 0 0 1 1 0 1 1 0 0 1 0 2	2 1 0 4 3

Regulations for completing the lab work

- Each question must be implemented as an independent program in a single C++ file (of format .cpp).
- The program must receive input and return output as specified Submissions with wrong regulation will result in a "0" (zero).
- Plagiarism and Cheating will result in a "0" (zero) for the entire course.
- Contact: Here.