Problem 1  (20 points)
You are a Grad student at the Department of chemistry and you need to transport \( n \) chemical reagent. When two chemicals are in the same container, they may react. You know all such combination of \(<\text{chemical}_1, \text{chemical}_2>\) which react when coming in contact (for the sake of simplicity, let’s assume that all reactions involve exactly two reagents). Therefore, such reacting chemicals must be transported in separate containers.

A naïve way would be to choose one container for each of the \( n \) reagents. However, you want to figure out if two containers are enough. Write an algorithm that decides if that is possible.

Problem 2  (20 points)
A maze is a collection of walls where people enter and usually get lost. At some places there are openings in the walls, through which people can pass. There is usually only one entrance to the maze through which people can get in or get out (see the bird’s eye view of a sample maze below).

Suppose a blind and deaf person is about to enter a maze and he has no tools or marking materials using which he can “mark” or “identify” any wall or openings. He can only touch the walls and walk, and may choose to either change direction or keep on walking straight when an opening in the wall comes. Design a technique which, if followed, can allow even such a physically challenged person to enter the maze and come out eventually. The algorithm might not be efficient, but it must be robust – it must not fail under any circumstances and it must work for all the mazes (not just the one showed in the example).
Problem 3  (20 points)
You are the chief engineer of a huge water reservoir project. There are $N$ lakes, and the water level is same in all the lakes since every lake is connected with every other lake through one or more canals.

Your job would ultimately require changing the water levels in different lakes (or among different clusters of lakes). The way to achieve that is to separate the interconnected lake system into a set of disconnected cluster of lakes, so that lakes within the same cluster are connected to one another but are not connected to any other lake in a different cluster. Of course, in order to do that you would need to close some of the canals, which is expensive.

You want to make a list of all those canals which has the unique property that if that canal is closed, the two water bodies it was connecting earlier will now become disconnected components. Design an algorithm to achieve it.

Problem 4 (20 points)
It is easy to see that for any graph $G$, both DFS and BFS would take the same amount of time. However, the space requirement may be considerably different.

a) Give an example of an $n$-vertex graph for which the depth of recursion of DFS starting from a particular vertex $v$ is $n-1$ whereas the queue of BFS has at most one vertex at any given time if BFS is started from the same vertex $v$.

b) Give an example of an $n$-vertex graph for which the queue of BFS has $n-1$ vertices at one time whereas the depth of recursion of DFS is at most one if starting from the same vertex $v$.

Problem 5 (15 points)
Consider the 0/1 knapsack instance with $n=4$, $p=[4,3,2,1]$, $w=[1,2,3,4]$, and $c$ (capacity$)=6$.

(a) Draw the state space tree for a four-object knapsack instance.
   (In this state space tree, only the leaf nodes are solution states.)
(b) Trace through the working of a FIFO branch-and-bound search.
(c) Trace through the working of a max-profit branch-and-bound search.
   (If we are searching for a solution with least cost, then the list of live nodes can be setup as a min heap. The next E-node is the live node with least cost. If we want a solution with maximum profit, the live node list can be set up as a max heap. The next E-node is the live node with maximum profit.)

Problem 6 (5 points)
Briefly describe how branch and bound differs from backtracking?