CS2001- Data Structures
Fall 2022

Question # 01:

Given an array arr[0 ... n-1] of distinct integers, the task is to find a local minima in it. We say that an element arr[x] is a local minimum if it is less than or equal to both its neighbors.

- For corner elements, we need to consider only one neighbor for comparison.
- There can be more than one local minima in an array, we need to find any one of them.

Input: arr[] = {1, 2, 3};
Output: Index of local minima is 0

Input: arr[] = {3, 2, 1};
Output: Index of local minima is 2

Input: arr[] = {23, 8, 15, 2, 3};
Output: Index of local minima is 1

Question # 02: *Recursion*

Azad is in wonderland and he has got a problem. He needs to estimate the number of possible ways in which he can escape a linear (1D) maze in front of him. He can move one step, two steps or four steps at a time(cells).
The size of maze is Max; given this information you need to write a recursive routine to check in how many different ways he can escape this maze.

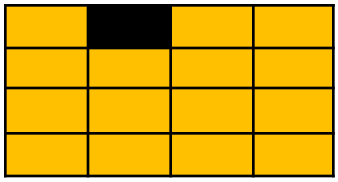
You need to identify the base case(s), recurring case(s).

i/j	0	1	2	3	4
0		*		*	*
1	*		*		*
2		*			
3	*				*
4	*	*		*	

Question # 03: *DMA*

Write a program that creates a 2D array of 5x5 values of type boolean. Suppose indices represent people and that the value at row i, column j of a 2D array is true just in case i and j are friends and false otherwise. Use initializer list to instantiate and initialize your array to represent the following configuration: (* means “friends”)

Question # 04: *Backtracking*



You are given a 4x4 grid and the objective is to have 4 obstacles in the grid in such a way that no two obstacles are placed in the same row, in the same column or in the same diagonal.
Now, assume that the initial state of the input grid is such that one obstacle is already placed in cell **(0,1).**

Write a program that places the remaining three obstacles in the grid using recursive function and backtracking.

Question # 05: *Linked List*

1. Describe in detail how to swap two nodes x and y (and not just their contents) in a singly linked list L given references only to x and y. Repeat this exercise for the case when L is a doubly linked list. Which algorithm takes more time?
2. Give an efficient algorithm for concatenating two doubly linked lists L and M, with head and tail preserved nodes, into a single list L's that contains all the nodes of L followed by all the nodes of M.
3. Take a number from user, search it in Doubly Linked List and delete that if exists(test it as front, tail, mid or no found).
4. Remove duplicate elements from Circular Singly Linked List.
5. Write a function to modify the linked list such that all even numbers appear before all the odd nodes in the modified linked list. Also keep the order of even and odd numbers same.
6. Given a 1d dynamic safe array, code the function to circularly shift the array contents backward to a k value which will be given by the user as input together with the array contents.
   Input: K=3 and Array={1,2,3,4,5,6}
   Output: {4,5,6,1,2,3}
   We will see implementation of concept through a source file named as “j4.cpp”.