CSE141 Object Oriented Programming (Fall'23)

*Maximum Time: 2 hours* May 10, 2024

# Lab Exam Questions

**Exercise 1** ........................................................................................

***Map of Highest Peak.*** You are given an integer matrix isWater of size $m \times n$ that represents a map of land and water cells.

If $\text{isWater}[i][j] = 0$, cell $(i, j)$ is a land cell. If $\text{isWater}[i][j] = 1$, cell $(i, j)$ is a water cell.

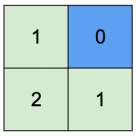
You must assign each cell a height in a way that follows these rules:

- The height of each cell must be non-negative.
- If the cell is a water cell, its height must be 0.
- Any two adjacent cells must have an absolute height difference of at most 1. A cell is adjacent to another cell if the former is directly north, east, south, or west of the latter (i.e., their sides are touching).

Find an assignment of heights such that the maximum height in the matrix is maximized.

Return an integer matrix height of size $m \times n$ where $\text{height}[i][j]$ is cell $(i, j)$'s height. If there are multiple solutions, return any of them.

**Example 1:**



| | |
|---|---|
| 1 | 0 |
| 2 | 1 |

**Input:** $\text{isWater} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}$

**Output:** $\begin{bmatrix} 1 & 0 \\ 2 & 1 \end{bmatrix}$

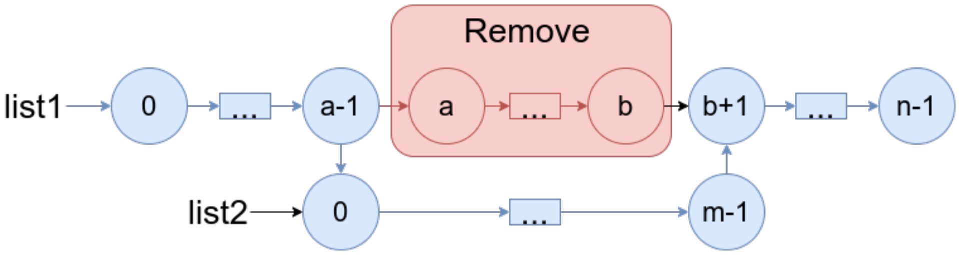
**Explanation:** The image shows the assigned heights of each cell. The blue cell is the water cell, and the green cells are the land cells.

**Exercise 2** ........................................................................................

***Merger in between LinkedList.*** You are given two linked lists: list1 and list2 of sizes n and m respectively.

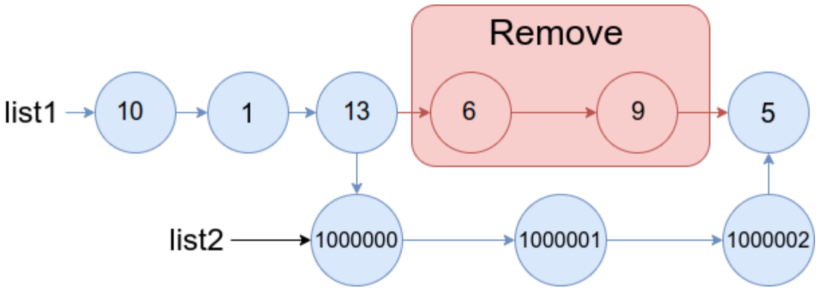
Remove list1's nodes from the ath node to the bth node, and put list2 in their place.

The blue edges and nodes in the following figure indicate the result:



Build the result list and return its head.

**Example 1:**



**Input:** list1 = $[10, 1, 13, 6, 9, 5]$, $a = 3$, $b = 4$, list2 = $[1000000, 1000001, 1000002]$
**Output:** $[10, 1, 13, 1000000, 1000001, 1000002, 5]$
**Explanation:** We remove the nodes 3 and 4 and put the entire list2 in their place. The blue edges and nodes in the above figure indicate the result.
A reference code for LinkedList creation is provided which is named as (**"LinkedList.cpp"**).

**Exercise 3** ........................................................................................

***CGPA Ranking.*** In this exercise, you are provided with the definition of a `Student` struct. Your task is to complete the `top_student` function, which finds the top male and female students based on their CGPA (Cumulative Grade Point Average).

Complete the implementation of the `top_student` function in the provided code (**"labexam4.cpp"**). It should find the top male and female students based on their CGPA. Use lambda functions and STL algorithms to achieve this. (Do not use any for/while loops.).

**Exercise 4** ........................................................................................

***Matrix Operations.*** You are provided with a Matrix class that supports basic operations like addition, subtraction, and multiplication of matrices. Your task is to create an exam question based on this code snippet.

Define a class `Matrix` with the following properties and methods:

**Properties:**

- `data`: A 2D vector representing the matrix elements.
- `rows`: An integer representing the number of rows in the matrix.
- `cols`: An integer representing the number of columns in the matrix.

**Methods:**

- `Matrix(size_t m, size_t n)`: A constructor that initializes the matrix with $m$ rows and $n$ columns.
- `Matrix operator+(const Matrix& other) const`: Overloads the addition operator $+$ to perform matrix addition.
- `Matrix operator-(const Matrix& other) const`: Overloads the subtraction operator $-$ to perform matrix subtraction.
- `Matrix operator*(const Matrix& other) const`: Overloads the multiplication operator $*$ to perform matrix multiplication.

Write the `main` function to demonstrate the usage of the `Matrix` class:

1. Create two matrices $A$ and $B$ with appropriate dimensions.
2. Perform matrix addition, subtraction, and multiplication using the overloaded operators $+$, $-$, and $*$, respectively.
3. Display the results of each operation.