Naafiul Hossain
ESE224
115107623
Tuesday 10-12:50am

Problem 1:

Main.cpp

```cpp
//Naafiul Hossain
//SBU ID: 115107623
#include <iostream>
#include <vector>
#include <algorithm>

using namespace std;

// Function to perform matrix multiplication
vector<vector<int>> multiplyMatrices(const vector<vector<int>>& A, const vector<vector<int>>& B) {
    int m = A.size();
    int k = A[0].size();
    int n = B[0].size();

    if (k != B.size()) {
        cerr << "Matrix dimensions are not suitable for multiplication. Exiting." << endl;
        exit(1);
    }

    vector<vector<int>> result(m, vector<int>(n, 0));

    for (int i = 0; i < m; i++) {
        for (int j = 0; j < n; j++) {
            for (int x = 0; x < k; x++) {
                result[i][j] += A[i][x] * B[x][j];
            }
        }
    }

    return result;
}
```

```cpp
// Function to perform matrix element-wise division
vector<vector<double>> elementWiseMatrixDivision(const vector<vector<int>>& A, const vector<vector<int
    int m = A.size();
    int n = A[0].size();
    int p = B.size();
    int q = B[0].size();

    if (m != p || n != q) {
        cerr << "Matrix dimensions are not suitable for element-wise division. Exiting." << endl;
        exit(1);
    }

    vector<vector<double>> result(m, vector<double>(n, 0.0));

    for (int i = 0; i < m; i++) {
        for (int j = 0; j < n; j++) {
            result[i][j] = static_cast<double>(A[i][j]) / static_cast<double>(B[i][j]);
        }
    }

    return result;
}
```


```cpp
// Function to perform matrix element-wise division
vector<vector<double>> elementWiseMatrixDivision(const vector<vector<int>>& A, const vector<vector<int
    int m = A.size();
    int n = A[0].size();
    int p = B.size();
    int q = B[0].size();

    if (m != p || n != q) {
        cerr << "Matrix dimensions are not suitable for element-wise division. Exiting." << endl;
        exit(1);
    }

    vector<vector<double>> result(m, vector<double>(n, 0.0));

    for (int i = 0; i < m; i++) {
        for (int j = 0; j < n; j++) {
            result[i][j] = static_cast<double>(A[i][j]) / static_cast<double>(B[i][j]);
        }
    }

    return result;
}
```

```
int main() {
    int m, k, n;
    cout << "Enter the dimensions of matrix A (m x k): ";
    cin >> m >> k;

    vector<vector<int>> matrixA(m, vector<int>(k));

    cout << "Enter the values of matrix A:" << endl;
    for (int i = 0; i < m; i++) {
        for (int j = 0; j < k; j++) {
            cin >> matrixA[i][j];
        }
    }

    cout << "Enter the dimensions of matrix B (k x n): ";
    cin >> k >> n;

    vector<vector<int>> matrixB(k, vector<int>(n));

    cout << "Enter the values of matrix B:" << endl;
    for (int i = 0; i < k; i++) {
        for (int j = 0; j < n; j++) {
            cin >> matrixB[i][j];
        }
    }

    vector<vector<int>> productMatrix = multiplyMatrices(matrixA, matrixB);
    vector<vector<double>> divisionMatrix = elementWiseMatrixDivision(matrixA, matrixB);
```

```
    // Sort the productMatrix in ascending order
    for (int i = 0; i < m; i++) {
        sort(productMatrix[i].begin(), productMatrix[i].end());
    }

    // Sort the divisionMatrix in descending order
    for (int i = 0; i < m; i++) {
        sort(divisionMatrix[i].rbegin(), divisionMatrix[i].rend());
    }

    // Output the results
    cout << "Matrix C (A * B) sorted in ascending order:" << endl;
    for (int i = 0; i < m; i++) {
        for (int j = 0; j < n; j++) {
            cout << productMatrix[i][j] << " ";
        }
        cout << endl;
    }

    cout << "Matrix D (A / B) sorted in descending order:" << endl;
    for (int i = 0; i < m; i++) {
        for (int j = 0; j < n; j++) {
            cout << divisionMatrix[i][j] << " ";
        }
        cout << endl;
    }

    return 0;
}
```

Screenshot of the running program:

```
Enter the dimensions of matrix A (m x k): 2 2
Enter the values of matrix A:
2 3
1 4
Enter the dimensions of matrix B (k x n): 2 2
Enter the values of matrix B:
5 6
7 8
Matrix C (A * B) sorted in ascending order:
31 36
33 38
Matrix D (A / B) sorted in descending order:
0.5 0.4
0.5 0.142857
```

Problem 2

main.h

```cpp
//Naafiul Hossain
//SBU id: 115107623
#include <iostream>
#include <vector>
#include <algorithm>
#include <string>

class Solution {
public:
    std::string longestCommonPrefix(std::vector<std::string>& strs) {
        std::sort(strs.begin(), strs.end()); //START at begin and end
        int a = strs.size(); //lengnth of our str
        std::string n = strs[0], m = strs[a - 1], ans = "";
        for (int i = 0; i < n.size() && i < m.size(); i++) { //transvser through our string
            if (n[i] == m[i]) { ans += n[i]; } //if same than add to answer
            else break;
        }
        return ans;
    }
};
```

```cpp
int main() {
    Solution solution;
    std::vector<std::string> strings = { "flower","flow","flight" };
    std::vector<std::string> strin = { "dog", "racecar", "car" };

    std::string result1 = solution.longestCommonPrefix(strings);

    Solution solution2;  // Create a new object for the second set of strings
    std::string result2 = solution2.longestCommonPrefix(strin);

    std::cout << "Longest common prefix for strings: " << result1 << std::endl;
    std::cout << "Longest common prefix for strin: " << result2 << std::endl;

    return 0;
}
```

Running of the Program:

```
Microsoft Visual Studio Debu

Longest common prefix for strings: fl
Longest common prefix for strin:

C:\Users\Naafiul Hossain\Documents\Lab2\Lab1Task1\NH
 0.
To automatically close the console when debugging st
le when debugging stops.
Press any key to close this window . . .
```

Problem 3

Main.cpp

```cpp
//Naafiul Hossain
//SBU ID: 115107623

#include <iostream>
#include <vector>
using namespace std;

// Returns a spiral matrix in clockwise order
vector<int> generateSpiralMatrix(vector<vector<int>>& matrix) {
    // Declare variables
    int top = 0, bottom = matrix.size() - 1;
    int left = 0, right = matrix[0].size() - 1;

    // Vector to store the spiral matrix
    vector<int> spiralResult;

    // Direction
    // '0' = Left to Right
    // '1' = Top to Bottom
    // '2' = Right to Left
    // '3' = Bottom to Top
    int direction = 0;

    while (top <= bottom && left <= right) {
        // Traverse left to right
        if (direction == 0) {
            for (int i = left; i <= right; i++)
                spiralResult.push_back(matrix[top][i]);

            top++;
            direction = 1;
        }
```

```
        }
        // Traverse top to bottom
        else if (direction == 1) {
            for (int i = top; i <= bottom; i++)
                spiralResult.push_back(matrix[i][right]);

            right--;
            direction = 2;
        }
        // Traverse right to left
        else if (direction == 2) {
            for (int i = right; i >= left; i--)
                spiralResult.push_back(matrix[bottom][i]);

            bottom--;
            direction = 3;
        }
        // Traverse bottom to top
        else if (direction == 3) {
            for (int i = bottom; i >= top; i--)
                spiralResult.push_back(matrix[i][left]);

            left++;
            direction = 0;
        }
    }

    return spiralResult;
}
```

```
int main() {
    // User input for 'm' and 'n'
    int rows, columns;
    //cin >> rows >> columns;
    cout << "Enter the number of rows: ";
    cin >> rows;

    cout << "Enter the number of columns: ";
    cin >> columns;
    // 2-D Vector
    vector<vector<int>> inputMatrix;

    // Populate the 2-D vector
    int count = 1;
    for (int i = 0; i < rows; i++) {
        vector<int> tempRow;
        for (int j = 0; j < columns; j++)
            tempRow.push_back(count++);

        inputMatrix.push_back(tempRow);
    }

    // Store the Spiral Matrix result in 'result'
    vector<int> result = generateSpiralMatrix(inputMatrix);

    // Print the spiral matrix
    for (int i = 0; i < result.size(); i++)
        cout << result[i] << " ";

    return 0;
```

Screenshot of the running program:

```
Microsoft Visual Studio Debu

Enter the number of rows: 3
Enter the number of columns: 3
1 2 3 6 9 8 7 4 5
C:\Users\Naafiul Hossain\Documents\Lab2\Lab1Task1\NHLab6Part3\x64\Debug\NHLab6Part3.exe (process 10612) exited with code
 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the conso
le when debugging stops.
Press any key to close this window . . .
```

Problem 4

Main.cpp

```cpp
//Naafiul Hossain
//SBU ID: 115107623

#include <iostream>
#include <vector>
//Sliding Window Algorthim

int maxLengthSubarraySum(std::vector<int>& nums, int k) {
    int left = 0;  //left pointer
    int sum = 0; //sum equal to 0
    int max_length = 0; //set max length to lowest value

    for (int right = 0; right < nums.size(); right++) { //start at right pointer and incrment right
        sum += nums[right]; //keep adding right pointer to the sum

        while (sum > k) { //as long as the sum is more than k
            sum -= nums[left]; //When sum exceeds k, we need to adjust the subarray by moving the left po
            left++;
        }

        max_length = std::max(max_length, right - left + 1);
        //We keep track of the maximum length of the subarray by calculating right - left + 1 and updatin
    }

    return max_length;
}
```

```cpp
int main() {
    // Test case 1
    std::vector<int> nums1 = { 3, 1, 3, 2, 2, 1 };
    int k1 = 4;
    int result1 = maxLengthSubarraySum(nums1, k1);
    std::cout << "Test Case 1: " << result1 << " (Expected Output: 2)" << std::endl;

    // You can add more test cases here to further validate the function.

    return 0;
}
```

Screenshot of the Program running:

```
Test Case 1: 2 (Expected Output: 2)

C:\Users\Naafiul Hossain\Documents\Lab2\Lab1Task1\NHLab6Par
 0.
To automatically close the console when debugging stops, en
le when debugging stops.
Press any key to close this window . . .
```

Problem 5

Main.cpp

```cpp
//Naafiul Hossain
//SBU ID: 115107623

#include <iostream>
#include <vector>

using namespace std;

vector<int> rowProduct(vector<vector<int>>& nums) {
    vector<int> result;
    for (int i = 0; i < nums.size(); i++) {
        int product = 1;
        for (int j = 0; j < nums[i].size(); j++) {
            product *= nums[i][j];
        }
        result.push_back(product);
    }
    return result;
}
```

```cpp
int maxProduct(vector<int>& nums) {
    if (nums.empty()) {
        return 0;
    }

    int maxProduct = nums[0];
    int currentMax = nums[0];
    int currentMin = nums[0];

    for (int i = 1; i < nums.size(); i++) {
        if (nums[i] < 0) {
            swap(currentMax, currentMin);
        }

        currentMax = max(nums[i], currentMax * nums[i]);
        currentMin = min(nums[i], currentMin * nums[i]);

        maxProduct = max(maxProduct, currentMax);
    }

    return maxProduct;
}
```

```cpp
int main() {
    vector<vector<int>> input = { {1, 2, 3}, {2, 2, 2}, {4, 0, 6}, {7, 5, 1}, {2, -2, 1} };

    vector<int> rowResult = rowProduct(input);
    int maxRowProduct = maxProduct(rowResult);

    cout << "Row products: ";
    for (int product : rowResult) {
        cout << product << " ";
    }
    cout << endl;

    cout << "Maximum product: " << maxRowProduct << endl;

    return 0;
}
```

Screenshot of the running program:

```
Row products: 6 8 0 35 -4
Maximum product: 48

C:\Users\Naafiul Hossain\Documents\Lab2\Lab1Task1\NHLab6P
 0.
To automatically close the console when debugging stops,
le when debugging stops.
Press any key to close this window . . .
```

Problem 6

Main.cpp

```cpp
//Naafiul Hossain

//backtracking


#include <iostream>
#include <vector>

using namespace std;

bool isValid(vector<vector<char>>& board, int row, int col, char digit) {
    for (int i = 0; i < 9; i++) {
        if (board[row][i] == digit || board[i][col] == digit || board[3 * (row / 3) + i / 3][3 * (col / 
            return false;
        }
    }
    return true;
}

bool solveSudoku(vector<vector<char>>& board) {
    for (int row = 0; row < 9; row++) {
        for (int col = 0; col < 9; col++) {
            if (board[row][col] == '.') {
                for (char digit = '1'; digit <= '9'; digit++) {
                    if (isValid(board, row, col, digit)) {
                        board[row][col] = digit;
                        if (solveSudoku(board)) {
                            return true;
                        }
                        board[row][col] = '.'; // Backtrack if the solution is not valid
                    }
                }
```

```cpp
void printSudoku(vector<vector<char>>& board) {
    for (int i = 0; i < 9; i++) {
        for (int j = 0; j < 9; j++) {
            cout << board[i][j] << " ";
        }
        cout << endl;
    }
}

int main() {
    vector<vector<char>> sudoku = {
        {'5', '3', '.', '.', '7', '.', '.', '.', '.'},
        {'6', '.', '.', '1', '9', '5', '.', '.', '.'},
        {'.', '9', '8', '.', '.', '.', '.', '6', '.'},
        {'8', '.', '.', '.', '6', '.', '.', '.', '3'},
        {'4', '.', '.', '8', '.', '3', '.', '.', '1'},
        {'7', '.', '.', '.', '2', '.', '.', '.', '6'},
        {'.', '6', '.', '.', '.', '.', '2', '8', '.'},
        {'.', '.', '.', '4', '1', '9', '.', '.', '5'},
        {'.', '.', '.', '.', '8', '.', '.', '7', '9'}
    };

    cout << "Original Sudoku Puzzle:" << endl;
    printSudoku(sudoku);

    if (solveSudoku(sudoku)) {
        cout << "\nSolved Sudoku Puzzle:" << endl;
        printSudoku(sudoku);
    }
    else {
```

```
    cout << "Original Sudoku Puzzle:" << endl;
    printSudoku(sudoku);

    if (solveSudoku(sudoku)) {
        cout << "\nSolved Sudoku Puzzle:" << endl;
        printSudoku(sudoku);
    }
    else {
        cout << "\nNo solution found." << endl;
    }

    return 0;
}
```

Running solution:

```
Original Sudoku Puzzle:
5 3 . . 7 . . . .
6 . . 1 9 5 . . .
. 9 8 . . . . 6 .
8 . . . 6 . . . 3
4 . . 8 . 3 . . 1
7 . . . 2 . . . 6
. 6 . . . . . 2 8 .
. . . 4 1 9 . . 5
. . . . 8 . . 7 9

Solved Sudoku Puzzle:
5 3 4 6 7 8 9 1 2
6 7 2 1 9 5 3 4 8
1 9 8 3 4 2 5 6 7
8 5 9 7 6 1 4 2 3
4 2 6 8 5 3 7 9 1
7 1 3 9 2 4 8 5 6
9 6 1 5 3 7 2 8 4
2 8 7 4 1 9 6 3 5
3 4 5 2 8 6 1 7 9

C:\Users\Naafiul Hossain\Documents\Lab2\Lab1
de 0.
```