Q1)

```cpp
#include <iostream> // Added to include std::cout
#include <vector>
#include <algorithm>

// Assuming Vector is similar to std::vector
typedef std::vector<int> Vector;

void rvrs(Vector& vct){
    int n = vct.size();
    for(int i = 0; i < n / 2; i++) {
        std::swap(vct[i], vct[n - i - 1]);
    }
}

int main() {
    Vector vct = {1, 2, 3, 4, 5};
    rvrs(vct);

    // Print the reversed vector
    for(int i : vct) {
        std::cout << i << " ";
    }
    std::cout << std::endl; // Added to print a newline after the vector

    return 0;
}
```

```
5 4 3 2 1
```

Q2)

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

void printDiagonal(const vector<vector<int>>& vals, int row, int col) {
    int numRows = vals.size();
    if (numRows == 0) return; // Check for empty input

    while (row >= 0 && col < vals[row].size()) {
        cout << vals[row][col] << " ";
        // Move to the next element in the diagonal
        row--;
        col++;
    }
    cout << endl;
}

int main() {
    vector<vector<int>> vals = {
        {1, 2, 3, 4},
        {5, 6, 7, 8},
        {9, 10, 11, 12},
        {13, 14, 15, 16}
    };

    // Example: Start from the lower-left corner
```

```cpp
int main() {
    vector<vector<int>> vals = {
        {1, 2, 3, 4},
        {5, 6, 7, 8},
        {9, 10, 11, 12},
        {13, 14, 15, 16}
    };

    // Example: Start from the lower-left corner
    printDiagonal(vals, 3, 0); // Starting from row 3, col 0


    return 0;
}
```

```
13 10 7 4
```

Q3)

```cpp
#include <iostream>
#include <vector>
#include <algorithm> // Needed for std::sort function

// Define the Tensor class
class Tensor {
public:
    // Method to sort a given vector and print its contents
    void sort(std::vector<int>& vec) {
        // Sorting the vector in ascending order
        std::sort(vec.begin(), vec.end());

        // Iterating over the sorted vector to print each element
        for (int i : vec) {
            std::cout << i << " ";
        }
        std::cout << std::endl; // End the line after printing the vector
    }
};
```

```cpp
int main() {
    // Creating an instance of the Tensor class
    Tensor tensor;

    // Defining a test vector with unsorted integers
    std::vector<int> vec = {3, 1, 4, 1, 5, 9, 2, 6};

    // Printing the original, unsorted vector to the console
    std::cout << "Original vector: ";
    for (int i : vec) {
        std::cout << i << " ";
    }
    std::cout << "\nSorted vector:   "; // Notice for starting the print of sorted vector

    // Calling the sort method of the tensor instance, which sorts and prints the vector
    tensor.sort(vec);

    return 0; // End of main function
}
```

```
Original vector: 3 1 4 1 5 9 2 6
Sorted vector:   1 1 2 3 4 5 6 9
```

Q4)

1. getIncrementedData is const but modifies data:
To fix this, you could either remove the const qualifier if modifying data is intended, or change the method's behavior so it does not modify dataRemove const or don't modify data.

2. getCount tries to access non-static data:
Remove access to data or make data static.

3. count is declared but not defined:
Static member variables must be defined outside the class definition in one (and only one) source file.Define count outside the class with int Example::count = 0;.

4. Missing #include <iostream> and namespace for cout:

The getCount method uses cout, which is defined in the <iostream> header,Add #include <iostream> and use std::cout or using namespace std;.

Correction:

```cpp
#include <iostream>
using namespace std;

class Example{
public:
    Example( int y = 10 ): data( y ){
        ++count;
    } // end Example constructor

    int getIncrementedData() {
        return ++data;
    } // end function getIncrementedData

    static int getCount(){
        cout << "Count is " << count << endl;
        return count;
    } // end function getCount

private:
    int data;
    static int count;
}; // end class Example

int Example::count = 0;
```

```cpp
int main() {
    Example ex;
    cout << ex.getIncrementedData() << endl;
    Example::getCount();
}
```

```
11
Count is 1
```