Lab Manual #10

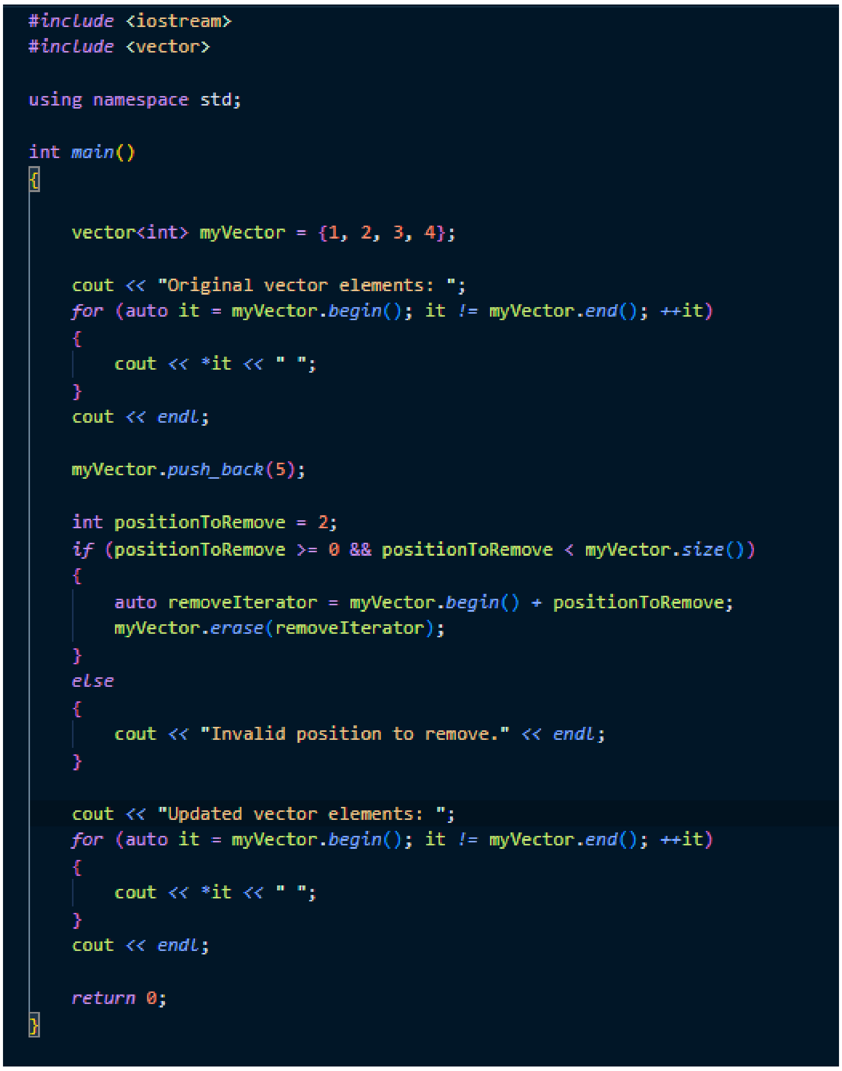
(Lab Tasks)

Course: **Fundamentals of Programming (CS 114)**

Lab Instructor: **Muhammad Affan**

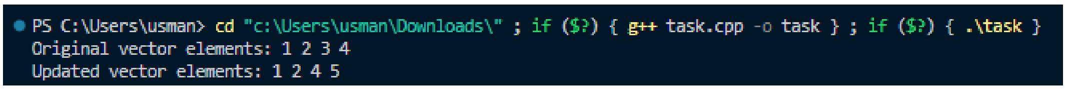
Name	**Muhammad Abdullah**
ID	**460901**
Section	**C**

Task-01



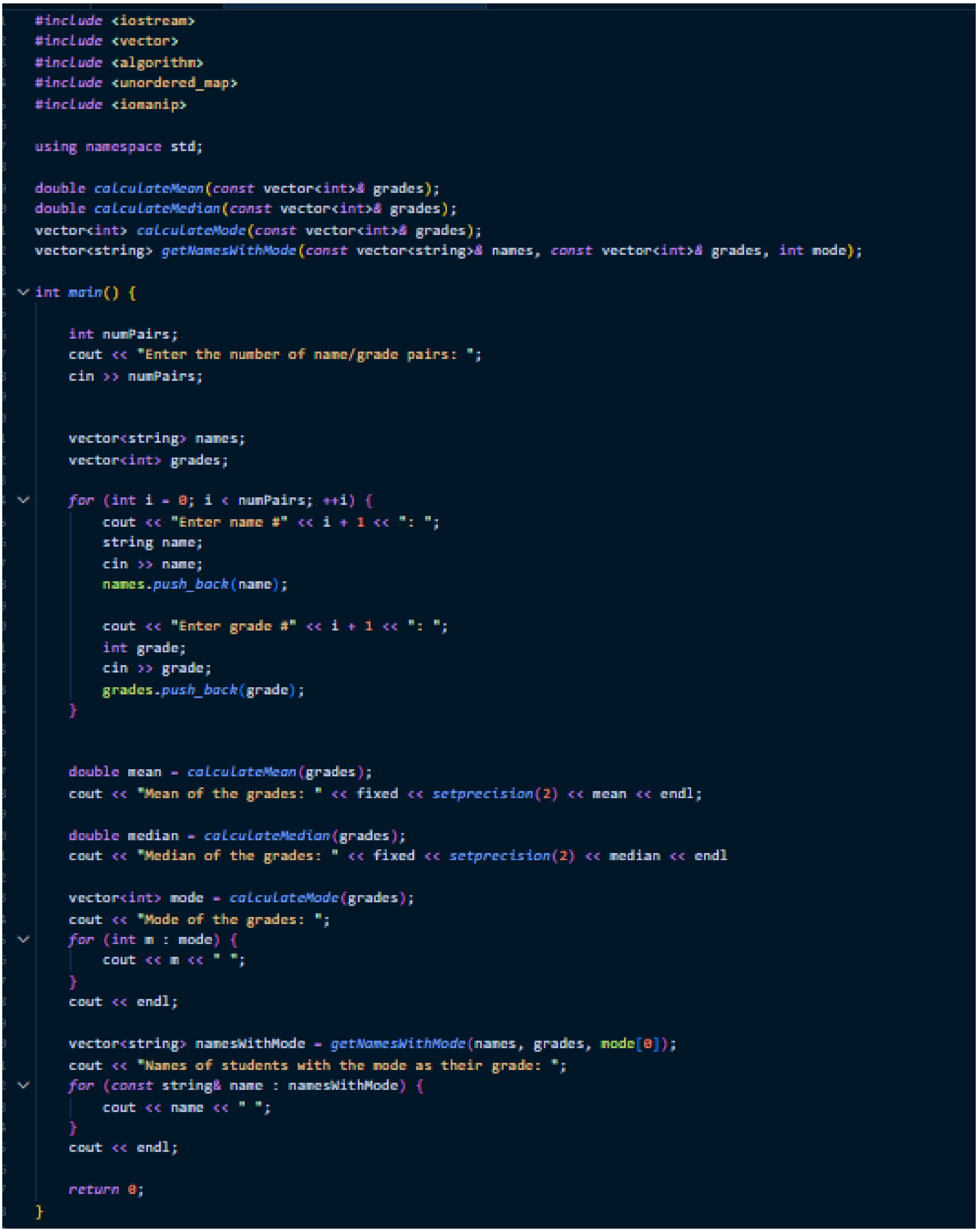
```cpp
#include <iostream>
#include <vector>

using namespace std;

int main()
{

    vector<int> myVector = {1, 2, 3, 4};

    cout << "Original vector elements: ";
    for (auto it = myVector.begin(); it != myVector.end(); ++it)
    {
        cout << *it << " ";
    }
    cout << endl;

    myVector.push_back(5);

    int positionToRemove = 2;
    if (positionToRemove >= 0 && positionToRemove < myVector.size())
    {
        auto removeIterator = myVector.begin() + positionToRemove;
        myVector.erase(removeIterator);
    }
    else
    {
        cout << "Invalid position to remove." << endl;
    }

    cout << "Updated vector elements: ";
    for (auto it = myVector.begin(); it != myVector.end(); ++it)
    {
        cout << *it << " ";
    }
    cout << endl;

    return 0;
}
```

Output

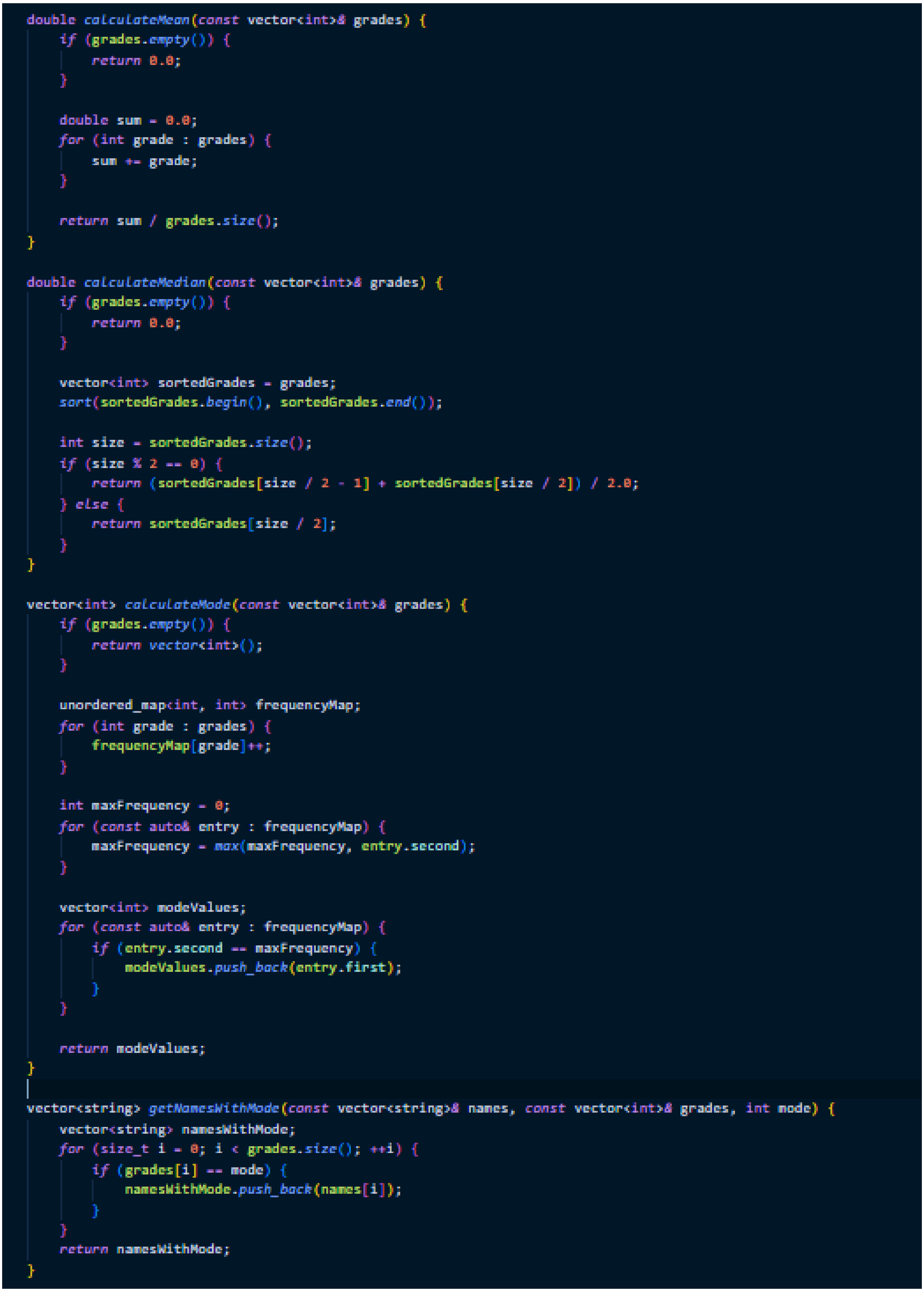


```
PS C:\Users\usman> cd "c:\Users\usman\Downloads\" ; if ($?) { g++ task.cpp -o task } ; if ($?) { .\task }
Original vector elements: 1 2 3 4
Updated vector elements: 1 2 4 5
```

Task-02

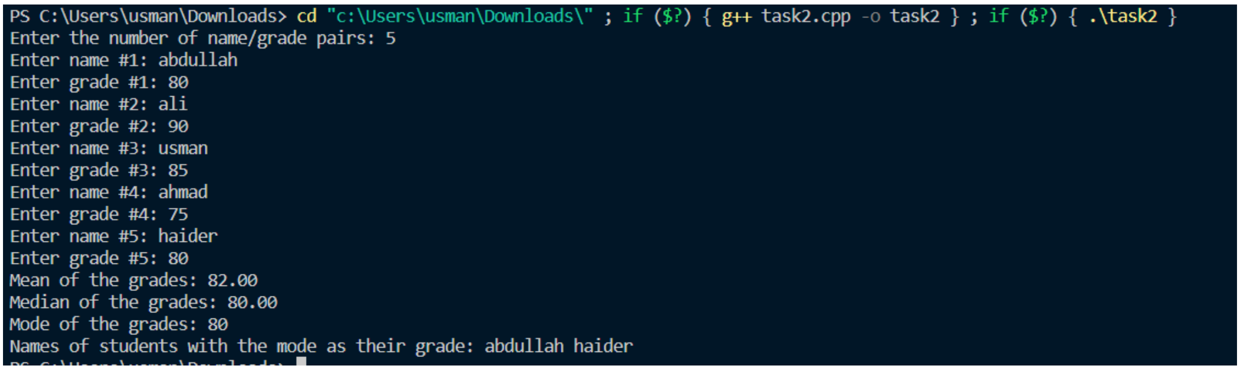


```
#include <iostream>
#include <vector>
#include <algorithm>
#include <unordered_map>
#include <iomanip>

using namespace std;

double calculateMean(const vector<int>& grades);
double calculateMedian(const vector<int>& grades);
vector<int> calculateMode(const vector<int>& grades);
vector<string> getNamesWithMode(const vector<string>& names, const vector<int>& grades, int mode);

int main() {

    int numPairs;
    cout << "Enter the number of name/grade pairs: ";
    cin >> numPairs;


    vector<string> names;
    vector<int> grades;

    for (int i = 0; i < numPairs; ++i) {
        cout << "Enter name #" << i + 1 << ": ";
        string name;
        cin >> name;
        names.push_back(name);

        cout << "Enter grade #" << i + 1 << ": ";
        int grade;
        cin >> grade;
        grades.push_back(grade);
    }


    double mean = calculateMean(grades);
    cout << "Mean of the grades: " << fixed << setprecision(2) << mean << endl;

    double median = calculateMedian(grades);
    cout << "Median of the grades: " << fixed << setprecision(2) << median << endl

    vector<int> mode = calculateMode(grades);
    cout << "Mode of the grades: ";
    for (int m : mode) {
        cout << m << " ";
    }
    cout << endl;

    vector<string> namesWithMode = getNamesWithMode(names, grades, mode[0]);
    cout << "Names of students with the mode as their grade: ";
    for (const string& name : namesWithMode) {
        cout << name << " ";
    }
    cout << endl;

    return 0;
}
```



```cpp
double calculateMean(const vector<int>& grades) {
    if (grades.empty()) {
        return 0.0;
    }

    double sum = 0.0;
    for (int grade : grades) {
        sum += grade;
    }

    return sum / grades.size();
}

double calculateMedian(const vector<int>& grades) {
    if (grades.empty()) {
        return 0.0;
    }

    vector<int> sortedGrades = grades;
    sort(sortedGrades.begin(), sortedGrades.end());

    int size = sortedGrades.size();
    if (size % 2 == 0) {
        return (sortedGrades[size / 2 - 1] + sortedGrades[size / 2]) / 2.0;
    } else {
        return sortedGrades[size / 2];
    }
}

vector<int> calculateMode(const vector<int>& grades) {
    if (grades.empty()) {
        return vector<int>();
    }

    unordered_map<int, int> frequencyMap;
    for (int grade : grades) {
        frequencyMap[grade]++;
    }

    int maxFrequency = 0;
    for (const auto& entry : frequencyMap) {
        maxFrequency = max(maxFrequency, entry.second);
    }

    vector<int> modeValues;
    for (const auto& entry : frequencyMap) {
        if (entry.second == maxFrequency) {
            modeValues.push_back(entry.first);
        }
    }

    return modeValues;
}

vector<string> getNamesWithMode(const vector<string>& names, const vector<int>& grades, int mode) {
    vector<string> namesWithMode;
    for (size_t i = 0; i < grades.size(); ++i) {
        if (grades[i] == mode) {
            namesWithMode.push_back(names[i]);
        }
    }
    return namesWithMode;
}
```

Output



```
PS C:\Users\usman\Downloads> cd "c:\Users\usman\Downloads\" ; if ($?) { g++ task2.cpp -o task2 } ; if ($?) { .\task2 }
Enter the number of name/grade pairs: 5
Enter name #1: abdullah
Enter grade #1: 80
Enter name #2: ali
Enter grade #2: 90
Enter name #3: usman
Enter grade #3: 85
Enter name #4: ahmad
Enter grade #4: 75
Enter name #5: haider
Enter grade #5: 80
Mean of the grades: 82.00
Median of the grades: 80.00
Mode of the grades: 80
Names of students with the mode as their grade: abdullah haider
PS C:\Users\usman\Downloads>
```