NAME: MUNEEB UR REHMAN

CLASS: ME (15) C

CMS   : 471811

Iterate Through Vector Using Iterators and print all pushed elements. Next you need to push integer 5 and remove element at that position

```
#include <iostream>
#include <vector>
using namespace std;
int main() {

    vector<int> aVector;



    aVector.push_back(1);
    aVector.push_back(2);
    aVector.push_back(3);
    aVector.push_back(4);

    // Iterate through the vector using iterators and print elements
    cout << "Original Vector: ";
    for (auto i = aVector.begin(); i != aVector.end(); ++i) {
        cout << *i << " ";
    }
    cout << endl;

    aVector.push_back(5);
    // Remove an element at a specific position
    if (!aVector.empty()) {

        int positionToRemove = 2;
        if (positionToRemove >= 0 && positionToRemove < aVector.size()) {
            aVector.erase(aVector.begin() + positionToRemove);
            cout << "Element at position " << positionToRemove << " removed." << endl;
        } else {
            cout << "Invalid position to remove element." << endl;
        }
    } else {
        cout << "Vector is empty. Cannot remove element." << endl;
    }


    cout << "Modified Vector: ";
    for (const auto &element : aVector) {
        cout << element << " ";
    }
    cout << endl;

    return 0;
}
```

1. **Write a complete C++ program that uses 2 vectors, 1 for names (string) and 1 for grades (int)**
    - a. **Ask the user for the number of name/grade pairs that will be entered.**
    - b. **Display the mean of the grades.**
    - c. **Display the median of the grades.**

d. Display the mode of the grades.

Display the names of the students with the mode as their grade

```
#include <iostream>
#include <vector>
#include <string>
#include <algorithm>
#include <unordered_map>
using namespace std;
int main() {
  //   for the number of name/grade pairs
  int numPairs;
  cout << "Enter the number of name/grade pairs: ";
  cin >> numPairs;

  // Create vectors for names and grades
  vector<std::string> names;
  vector<int> grades;

  // Step 3: Input name/grade pairs from the user
  for (int i = 0; i < numPairs; ++i) {
    string name;
    int grade;

    cout << "Enter name for student " << i + 1 << ": ";
    cin >> name;

    cout << "Enter grade for student " << i + 1 << ": ";
    cin >> grade;


    names.push_back(name);
    grades.push_back(grade);
}


// the mean of the grades
double mean = 0.0;
for (int grade : grades) {
    mean += grade;
}
mean /= numPairs;
cout << "Mean of grades: " << mean << endl;


// the median of the grades
sort(grades.begin(), grades.end());
double median;
if (numPairs % 2 == 0) {
    median = (grades[numPairs / 2 - 1] + grades[numPairs / 2]) / 2.0;
} else {
    median = grades[numPairs / 2];
}
cout << "Median of grades: " << median << endl;


// the mode of the grades
unordered_map<int, int> gradeFrequency;
for (int grade : grades) {
    gradeFrequency[grade]++;
}
```

```
    int mode = grades[0];
    int maxFrequency = 1;

    for (const auto& pair : gradeFrequency) {
        if (pair.second > maxFrequency) {
            mode = pair.first;
            maxFrequency = pair.second;
        }
    }

    cout << "Mode of grades: " << mode << endl;

    //  the names of the students with the mode as their grade
    cout << "Students with mode grade (" << mode << "): ";
    for (size_t i = 0; i < grades.size(); ++i) {
        if (grades[i] == mode) {
            std::cout << names[i] << " ";
        }
    }
    cout << endl;

    return 0;
}
    int mode = grades[0];
    int maxFrequency = 1;

    for (const auto& pair : gradeFrequency) {
        if (pair.second > maxFrequency) {
            mode = pair.first;
            maxFrequency = pair.second;
        }
    }

    std::cout << "Mode of grades: " << mode << std::endl;

    // Step 7: Display the names of the students with the mode as their grade
    std::cout << "Students with mode grade (" << mode << "): ";
    for (size_t i = 0; i < grades.size(); ++i) {
        if (grades[i] == mode) {
            std::cout << names[i] << " ";
        }
    }
    std::cout << std::endl;

    return 0;
}
```