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

using namespace std;

// Function to calculate the mean of a given set
of numbers
double calculateMean(const vector<int>&
numbers) {
    int sum = 0;
    for (int num : numbers) {
        sum += num;
    }
    return static_cast<double>(sum) /
numbers.size();
}

// Function to calculate the median of a given
set of numbers
double calculateMedian(const vector<int>&
numbers) {
    vector<int> sortedNumbers = numbers;
```

```
    sort(sortedNumbers.begin(),
sortedNumbers.end());

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

// Function to calculate the mode of a given set
of numbers
vector<int> calculateMode(const vector<int>&
numbers) {
    unordered_map<int, int> frequencyMap;
    vector<int> modes;

    int maxFrequency = 0;
    for (int num : numbers) {
        frequencyMap[num]++;
        maxFrequency = max(maxFrequency,
frequencyMap[num]);
```

```
    }

    for (const auto& pair : frequencyMap) {
        if (pair.second == maxFrequency) {
            modes.push_back(pair.first);
        }
    }

    return modes;
}

int main() {
    vector<int> numbers = {2, 4, 6, 2, 8, 4, 5, 8, 2,
4};  // Example set of numbers

    // Calculate and display the mean
    double mean = calculateMean(numbers);
    cout << "Mean: " << mean << endl;

    // Calculate and display the median
    double median = calculateMedian(numbers);
    cout << "Median: " << median << endl;

    // Calculate and display the mode
```

```
    vector<int> modes = calculateMode(numbers);
    cout << "Mode: ";
    for (int mode : modes) {
        cout << mode << " ";
    }
    cout << endl;

    return 0;
}
```