Task 1:

```
#include <iostream>
#include <vector>
using namespace std;
int main() {
   vector<int> myVector;
   myVector.push_back(10);
   myVector.push_back(20);
   myVector.push_back(30);
   myVector.push_back(40);
   cout << "Elements in the vector: ";
   for (auto it = myVector.begin(); it != myVector.end(); ++it) {
      cout << *it << " ";  }
   cout << endl;
   myVector.push_back(5);
   if (!myVector.empty() && myVector.size() > 2) {
      vector<int>::iterator itToRemove = myVector.begin() + 2;
      myVector.erase(itToRemove); }
   cout << "Elements after pushing 5 and removing element at position 2: ";
   for (const auto &element : myVector) {
      cout << element << " "; }
   cout << endl;
   return 0; }
```

Task 2:

```
#include <iostream>
#include <vector>
using namespace std;
double calculateMean(const vector<int>& grades) {
   if (grades.empty()) {
      return 0.0; }
   int sum = 0;
   for (int grade : grades) {
      sum += grade;}
   return static_cast<double>(sum) / grades.size(); }
double calculateMedian(const vector<int>& grades) {
   if (grades.empty()) {
      return 0.0; }
   vector<int> sortedGrades = grades;
   for (size_t i = 0; i < sortedGrades.size() - 1; ++i) {
      for (size_t j = 0; j < sortedGrades.size() - i - 1; ++j) {
         if (sortedGrades[j] > sortedGrades[j + 1]) {
            swap(sortedGrades[j], sortedGrades[j + 1]); }}}
   size_t size = sortedGrades.size();
   if (size % 2 == 0) {
      return (sortedGrades[size / 2 - 1] + sortedGrades[size / 2]) / 2.0;
   } else {
      return sortedGrades[size / 2];}}
```

```
int calculateMode(const vector<int>& grades, vector<int>& modeGrades) {
  if (grades.empty()) {
    return 0;}
  vector<int> frequency(101, 0);
  int maxFrequency = 0;
  for (int grade : grades) {
    ++frequency[grade];
    maxFrequency = max(maxFrequency, frequency[grade]); }
  for (size_t i = 0; i < frequency.size(); ++i) {
    if (frequency[i] == maxFrequency) {
      modeGrades.push_back(static_cast<int>(i)); } }
  return maxFrequency; }
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) {
    string name;
    int grade;
    cout << "Enter name #" << i + 1 << ": ";
    cin >> name;
    cout << "Enter grade #" << i + 1 << ": ";
    cin >> grade;
    names.push_back(name);
    grades.push_back(grade);   }
  double mean = calculateMean(grades);
  double median = calculateMedian(grades);
  vector<int> modeGrades;
  int modeFrequency = calculateMode(grades, modeGrades);
  cout << fixed;
  cout.precision(2);
  cout << "Mean of the grades: " << mean << endl;
  cout << "Median of the grades: " << median << endl;
  if (modeFrequency > 1) {
    cout << "Mode of the grades: ";
    for (int modeGrade : modeGrades) {
      cout << modeGrade << " ";   }
    cout << endl;
    cout << "Students with the mode as their grade: ";
    for (size_t i = 0; i < grades.size(); ++i) {
      if (find(modeGrades.begin(), modeGrades.end(), grades[i]) != modeGrades.end()) {
        cout << names[i] << " ";} }
    cout << endl;
  } else {
    cout << "No mode found." << endl;  }
  return 0; }
```