Name	M.Salar Azam
Class	ME-15(C)
CMS	479001

Q1:

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

using namespace std;

int main() {
   vector<int> numbers = {1, 4, 2, 8, 5};

   // Iterate through the vector using iterators and print elements
   cout << "Original vector elements: ";
   for (vector<int>::iterator it = numbers.begin(); it != numbers.end(); ++it) {
```

```
    cout << *it << " ";
  }

  // Push integer 5
  numbers.push_back(5);

  // Remove the element at the position of the newly added 5
  int index_to_remove = numbers.size() - 1;  // Get the index of the last element
  numbers.erase(numbers.begin() + index_to_remove);

  // Print the modified vector elements
  cout << "\nModified vector elements: ";
  for (int num : numbers) {
    cout << num << " ";
  }
```

```
    cout << endl;

    return 0;
}
```

Q2:

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

using namespace std;

// Function to calculate the mean of a vector of integers
double calculateMean(const vector<int>& grades) {
    int sum = 0;
    for (int grade : grades) {
```

```
        sum += grade;
    }
    return static_cast<double>(sum) /
grades.size();
}

// Function to calculate the median of a vector of
integers
double calculateMedian(vector<int>& grades) {
    sort(grades.begin(), grades.end());
    int size = grades.size();
    if (size % 2 == 0) {
        return (grades[size / 2 - 1] + grades[size / 2])
/ 2.0;
    } else {
        return grades[size / 2];
    }
}
```

```
// Function to calculate the mode of a vector of integers
vector<int> calculateMode(const vector<int>& grades) {
    unordered_map<int, int> frequency;
    for (int grade : grades) {
        frequency[grade]++;
    }

    vector<int> modes;
    int maxFrequency = 0;
    for (const auto& pair : frequency) {
        if (pair.second > maxFrequency) {
            maxFrequency = pair.second;
            modes.clear();
            modes.push_back(pair.first);
        } else if (pair.second == maxFrequency) {
            modes.push_back(pair.first);
        }
```

```
    }
    return modes;
}

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

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

    // Input name/grade pairs
    for (int i = 0; i < numPairs; ++i) {
        string name;
        int grade;
        cout << "Enter name " << i + 1 << ": ";
        cin >> name;
```

```
        names.push_back(name);
        cout << "Enter grade for " << name << ": ";
        cin >> grade;
        grades.push_back(grade);
    }

    // Calculate mean
    double mean = calculateMean(grades);
    cout << "Mean of the grades: " << mean << endl;

    // Calculate median
    double median = calculateMedian(grades);
    cout << "Median of the grades: " << median << endl;

    // Calculate mode
    vector<int> modes = calculateMode(grades);
    cout << "Mode of the grades: ";
```

```
    for (int mode : modes) {
        cout << mode << " ";
    }
    cout << endl;

    // Display names with the mode as their grade
    cout << "Names with the mode as their grade:
";
    for (size_t i = 0; i < grades.size(); ++i) {
        if (find(modes.begin(), modes.end(),
grades[i]) != modes.end()) {
            cout << names[i] << " ";
        }
    }
    cout << endl;

    return 0;
}
```

Q4:

```
#include <iostream>
#include <string>

using namespace std;

struct Employee {
    string name;
    double salary;
    int hoursWorked;
};

void increaseSalary(Employee &employee) {
    if (employee.hoursWorked >= 12) {
        employee.salary += 150;
    } else if (employee.hoursWorked >= 10) {
        employee.salary += 100;
    } else if (employee.hoursWorked >= 8) {
        employee.salary += 50;
```

```
    }
}

int main() {
    const int numEmployees = 10;
    Employee employees[numEmployees];

    // Input employee information
    for (int i = 0; i < numEmployees; ++i) {
        cout << "Enter name of employee " << i + 1
<< ": ";
        cin >> employees[i].name;
        cout << "Enter salary of employee " <<
employees[i].name << ": ";
        cin >> employees[i].salary;
        cout << "Enter hours of work per day of
employee " << employees[i].name << ": ";
        cin >> employees[i].hoursWorked;
    }
```

```
    // Increase salary based on hours worked per
day
    for (int i = 0; i < numEmployees; ++i) {
        increaseSalary(employees[i]);
    }

    // Display employees' names and final salaries
    cout << "\nEmployees and their final salaries
after increase:\n";
    for (int i = 0; i < numEmployees; ++i) {
        cout << employees[i].name << ": $" <<
employees[i].salary << endl;
    }

    return 0;
}
```

Q3:

```
#include <iostream>
```

```
#include <cmath>

using namespace std;

class Triangle {
private:
    double side1, side2, side3;

public:
    Triangle(double s1, double s2, double s3) :
side1(s1), side2(s2), side3(s3) {}

    double calculatePerimeter() {
        return side1 + side2 + side3;
    }

    double calculateArea() {
        double s = calculatePerimeter() / 2.0;
```

```
        return sqrt(s * (s - side1) * (s - side2) * (s - side3));
    }

    void printDetails() {
        cout << "Triangle with sides: " << side1 << " m, " << side2 << " m, " << side3 << " m\n";
        cout << "Perimeter: " << calculatePerimeter() << " m\n";
        cout << "Area: " << calculateArea() << " square meters\n";
    }
};

int main() {
    double side1 = 3.0, side2 = 4.0, side3 = 5.0;

    Triangle triangle(side1, side2, side3);
    triangle.printDetails();
```

```
    return 0;
}
```