*National University of Science and Technology*

**School of Mechanical and Manufacturing Engineering**

Lab Manual #10

**CS-114 Fundamentals of Programming**

**Course Instructor:** Khawaja Fahad Iqbal

**Lab Instructor:** Muhammad Affan

**Introduction:**

**Name:** Muhammad Furqan Ul Arsh

**CMS ID:** 476347

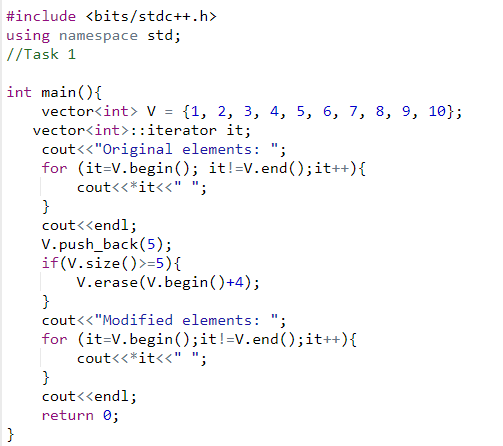
**Section:** ME-15B

**Date:** 27-12-2023

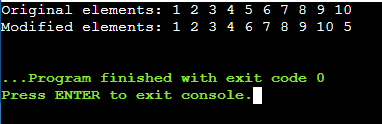
**Task 1:**

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

**Solution:**



**Result:**



**Task 2:**

1. Write a complete C++ program that uses 2 vectors, 1 for names (string) and 1 for grades (int)
   1. Ask the user for the number of name/grade pairs that will be entered.
   2. Display the mean of the grades.
   3. Display the median of the grades.
   4. Display the mode of the grades.
   5. Display the names of the students with the mode as their grade.

**Solution:**

#include <bits/stdc++.h>

using namespace std;

//Task 2

// 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(const vector<int>& grades) {

vector<int> sortedGrades = grades;

sort(sortedGrades.begin(), sortedGrades.end());

size\_t size = sortedGrades.size();

if (size % 2 == 0) {

// If the size is even, return the average of the middle two elements

return static\_cast<double>(sortedGrades[size / 2 - 1] + sortedGrades[size / 2]) / 2.0;

} else {

// If the size is odd, return the middle element

return static\_cast<double>(sortedGrades[size / 2]);

}

}

// Function to calculate the mode of a vector of integers

vector<int> calculateMode(const vector<int>& grades) {

map<int, int> gradeFrequency;

// Count the frequency of each grade

for (int grade : grades) {

gradeFrequency[grade]++;

}

// Find the mode(s)

int maxFrequency = 0;

vector<int> modeGrades;

for (const auto& pair : gradeFrequency) {

if (pair.second > maxFrequency) {

maxFrequency = pair.second;

modeGrades.clear();

modeGrades.push\_back(pair.first);

} else if (pair.second == maxFrequency) {

modeGrades.push\_back(pair.first);

}

}

return modeGrades;

}

int main() {

// Ask the user for the number of name/grade pairs

int numPairs;

cout << "Enter the number of name/grade pairs: ";

cin >> numPairs;

// Validate input

if (numPairs <= 0) {

cout << "Invalid input. Exiting program." << endl;

return 1;

}

// Create vectors for names and grades

vector<string> names;

vector<int> grades;

// Input names and grades

for (int i = 0; i < numPairs; ++i) {

string name;

int grade;

cout << "Enter name #" << i + 1 << ": ";

cin >> name;

cout << "Enter grade for " << name << ": ";

cin >> grade;

names.push\_back(name);

grades.push\_back(grade);

}

// Display the mean of the grades

double mean = calculateMean(grades);

cout << "Mean of the grades: " << mean << endl;

// Display the median of the grades

double median = calculateMedian(grades);

cout << "Median of the grades: " << median << endl;

// Display the mode of the grades

vector<int> modeGrades = calculateMode(grades);

cout << "Mode of the grades: ";

for (int mode : modeGrades) {

cout << mode << " ";

}

cout << endl;

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

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;

return 0;

}

**Result:**

