Lab Manual 10:

Name: Romaisa Yaqoob

ID: 469297

Class: ME15 A

Subject: Fundamentals of Programming Lab

Date: 25 Dec 2023

Submitted to: Sir Affan

Task 1: Iterate Through Vector Using Iterators and print all pushed elements. Next you need to push integer 5 and remove element at that position.

Code:

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

using namespace std;

int main() {
    // Declare a vector of integers with at least 10 elements
    vector<int> myVector;
    myVector.push_back(1);
    myVector.push_back(2);
    myVector.push_back(3);
    myVector.push_back(4);
    myVector.push_back(5);
    myVector.push_back(6);
    myVector.push_back(7);
    myVector.push_back(8);
    myVector.push_back(9);
    myVector.push_back(10);

    // Print elements using iterators
    cout << "Elements in the vector: ";
    for (vector<int>::iterator it = myVector.begin(); it != myVector.end(); ++it) {
        cout << *it << " ";
    }
    cout << endl;

    // Push integer 5 into the vector
    myVector.push_back(5);

    // Remove the element at position 5 (index 4)
    if (!myVector.empty() && myVector.size() > 4) {
        vector<int>::iterator itToRemove = myVector.begin() + 4;
        myVector.erase(itToRemove);
    }

    // Print elements after pushing 5 and removing an element
    cout << "Elements after pushing 5 and removing an element: ";
    for (vector<int>::iterator it = myVector.begin(); it != myVector.end(); ++it) {
        cout << *it << " ";
    }
    cout << endl;

    return 0;
}
```

Output:

```
Elements in the vector: 1 2 3 4 5 6 7 8 9 10
Elements after pushing 5 and removing an element: 1 2 3 4 6 7 8 9 10 5

--------------------------------
Process exited after 0.4609 seconds with return value 0
Press any key to continue . . .
```

Task 2: 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.
e. Display the names of the students with the mode as their grade.

Code:

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

using namespace std;

// Function to calculate the mean of grades
double calculateMean(const vector<int>& grades) {
    if (grades.empty()) {
        return 0.0;
    }

    int sum = 0;
    for (size_t i = 0; i < grades.size(); ++i) {
        sum += grades[i];
    }

    return static_cast<double>(sum) / grades.size();
}

// Function to calculate the median of grades
double calculateMedian(vector<int>& grades) {
    if (grades.empty()) {
        return 0.0;
    }

    // Sort the grades (using bubble sort for simplicity)
    for (size_t i = 0; i < grades.size(); ++i) {
        for (size_t j = i + 1; j < grades.size(); ++j) {
            if (grades[j] < grades[i]) {
                // Swap elements
                int temp = grades[i];
                grades[i] = grades[j];
                grades[j] = temp;
            }
        }
    }

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

```cpp
// Function to calculate the mode of grades
vector<int> calculateMode(const vector<int>& grades) {
    vector<int> gradeFrequency(101, 0);  // Assuming grades are between 0 and 100

    // Count the frequency of each grade
    for (size_t i = 0; i < grades.size(); ++i) {
        gradeFrequency[grades[i]]++;
    }

    // Find the mode(s)
    vector<int> mode;
    int maxFrequency = 0;
    for (size_t i = 0; i < gradeFrequency.size(); ++i) {
        if (gradeFrequency[i] > maxFrequency) {
            maxFrequency = gradeFrequency[i];
            mode.clear();
            mode.push_back(static_cast<int>(i));
        } else if (gradeFrequency[i] == maxFrequency) {
            mode.push_back(static_cast<int>(i));
        }
    }

    return mode;
}

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

    // Vectors to store names and grades
    vector<string> names(numPairs);
    vector<int> grades(numPairs);

    // Input name/grade pairs from the user
    for (size_t i = 0; i < static_cast<size_t>(numPairs); ++i) {
        cout << "Enter name for student " << i + 1 << ": ";
        cin >> names[i];

        cout << "Enter student's marks: " << i + 1 << ": ";
        cin >> grades[i];
    }

    // Display the mean of the grades
    cout << "Mean of grades: " << calculateMean(grades) << endl;

    // Display the median of the grades
    cout << "Median of grades: " << calculateMedian(grades) << endl;

    // Display the mode of the grades
    vector<int> mode = calculateMode(grades);
    cout << "Mode of grades: ";
    for (size_t i = 0; i < mode.size(); ++i) {
        cout << mode[i] << " ";
    }
    cout << endl;

    // Display the names of students with the mode as their grade
    cout << "Names of students with the mode as their grade: ";
    for (size_t i = 0; i < grades.size(); ++i) {
        bool isMode = false;
        for (size_t j = 0; j < mode.size(); ++j) {
            if (grades[i] == mode[j]) {
                isMode = true;
                break;
            }
        }
        if (isMode) {
            cout << names[i] << "  ";
        }
    }
    cout << endl;

    return 0;
}
```

Output:

```
Enter the number of students: 4
Enter name for student 1: qwerty
Enter student's marks: 1: 90
Enter name for student 2: poverty
Enter student's marks: 2: 10
Enter name for student 3: prosperity
Enter student's marks: 3: 95
Enter name for student 4: equality
Enter student's marks: 4: 90
Mean of grades: 71.25
Median of grades: 90
Mode of grades: 90
Names of students with the mode as their grade: poverty prosperity

--------------------------------
Process exited after 77.29 seconds with return value 0
Press any key to continue . . .
```

Task 3: Write a program to print the area and perimeter of a triangle having sides of 3 m, 4 m and 5 m by creating a class named 'Triangle' with a function to print the area and perimeter.

Code:

```cpp
#include <iostream>
#include <cmath>

using namespace std;

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

public:
    // Constructor to initialize the sides of the triangle
    Triangle(double s1, double s2, double s3) : side1(s1), side2(s2), side3(s3) {}

    // Function to calculate the perimeter of the triangle
    double calculatePerimeter() {
        return side1 + side2 + side3;
    }

    // Function to calculate the area of the triangle using Heron's formula
    double calculateArea() {
        double s = calculatePerimeter() / 2; // Semi-perimeter
        return sqrt(s * (s - side1) * (s - side2) * (s - side3));
    }

    // Function to print the area and perimeter of the triangle
    void printDetails() {
        cout << "Triangle with sides: " << side1 << " m, " << side2 << " m, " << side3 << " m\n";
        cout << "Perimeter: " << calculatePerimeter() << " m\n";
        cout << "Area: " << calculateArea() << " m^2\n";
    }
};

int main() {
    // Create a Triangle object with sides 3 m, 4 m, and 5 m
    Triangle myTriangle(3, 4, 5);

    // Print the details of the triangle (area and perimeter)
    myTriangle.printDetails();

    return 0;
}
```

Output:

```
Triangle with sides: 3 m, 4 m, 5 m
Perimeter: 12 m
Area: 6 m^2

--------------------------------
Process exited after 0.4141 seconds with return value 0
Press any key to continue . . .
```

Task 4: Write a structure to store the names, salary, and hours of work per day of 10 employees in a company. Write a program to increase the salary depending on the number of hours of work per day as follows and then print the name of all the employees along with their final salaries.

Hours of work per day	8	10	>=12
Increase in Salary	$50	$100	$150

Code:

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

using namespace std;

// Define the structure to store employee information
struct Employee {
    string name;
    double salary;
    int hoursWorked;
};

// Function to calculate and update salary based on hours worked
void updateSalary(Employee& emp) {
    if (emp.hoursWorked >= 8 && emp.hoursWorked < 10) {
        emp.salary += 50.0;
    } else if (emp.hoursWorked >= 10 && emp.hoursWorked < 12) {
        emp.salary += 100.0;
    } else if (emp.hoursWorked >= 12) {
        emp.salary += 150.0;
    }
}
```

```cpp
int main() {
    const int numEmployees = 10;

    // Declare an array of Employee structures
    Employee employees[numEmployees];

    // Input employee information
    for (int i = 0; i < numEmployees; ++i) {
        cout << "Enter name for employee " << i + 1 << ": ";
        cin >> employees[i].name;

        cout << "Enter salary for employee " << i + 1 << ": $";
        cin >> employees[i].salary;

        cout << "Enter hours worked for employee " << i + 1 << ": ";
        cin >> employees[i].hoursWorked;
    }

    // Update salaries based on hours worked
    for (int i = 0; i < numEmployees; ++i) {
        updateSalary(employees[i]);
    }

    // Print employee names and final salaries
    cout << "\nEmployee Names and Final Salaries:\n";
    for (int i = 0; i < numEmployees; ++i) {
        cout << employees[i].name << ": $" << employees[i].salary << endl;
    }

    return 0;
}
```

Output:

```
Enter name for employee 1: groot
Enter salary for employee 1: $100
Enter hours worked for employee 1: 5
Enter name for employee 2: smoot
Enter salary for employee 2: $200
Enter hours worked for employee 2: 6
Enter name for employee 3: froot
Enter salary for employee 3: $300
Enter hours worked for employee 3: 7
Enter name for employee 4: soot
Enter salary for employee 4: $400
Enter hours worked for employee 4: 8
Enter name for employee 5: moot
Enter salary for employee 5: $500
Enter hours worked for employee 5: 9
Enter name for employee 6: scoot
Enter salary for employee 6: $600
Enter hours worked for employee 6: 10
Enter name for employee 7: foot
Enter salary for employee 7: $700
Enter hours worked for employee 7: 11
Enter name for employee 8: hoot
Enter salary for employee 8: $800
Enter hours worked for employee 8: 12
Enter name for employee 9: shoot
Enter salary for employee 9: $900
Enter hours worked for employee 9: 13
Enter name for employee 10: scoobytoo
Enter salary for employee 10: $1000
Enter hours worked for employee 10: 14

Employee Names and Final Salaries:
groot: $100
smoot: $200
froot: $300
soot: $450
moot: $550
scoot: $700
foot: $800
hoot: $950
shoot: $1050
scoobytoo: $1150

--------------------------------
Process exited after 298.6 seconds with return value 0
Press any key to continue . . .
```