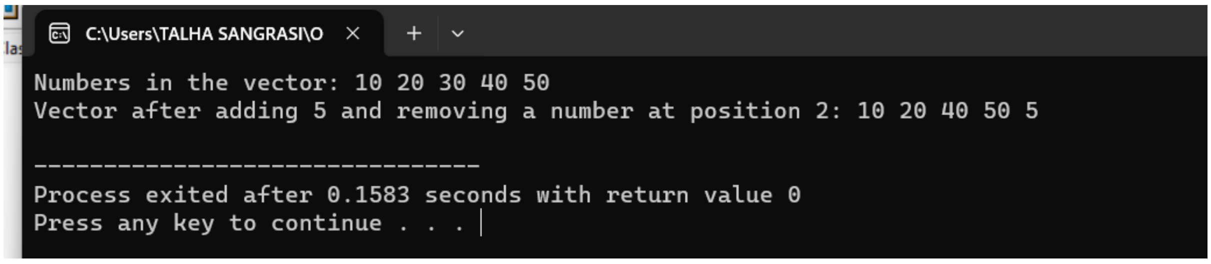
Lab Manual 10

Task 01

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

using namespace std;

int main() {
    vector<int> g2;

    g2.push_back(10);
    g2.push_back(20);
    g2.push_back(30);
    g2.push_back(40);
    g2.push_back(50);

    cout << "Numbers in the vector: ";
    for (vector<int>::iterator it = g2.begin(); it != g2.end(); ++it) {
        cout << *it << " ";
    }
    cout << endl;

    g2.push_back(5);

    int positionToRemove = 2;
    if (positionToRemove >= 0 && positionToRemove < static_cast<int>(g2.size())) {
        vector<int>::iterator itToRemove = g2.begin() + positionToRemove;
        g2.erase(itToRemove);
    }

    cout << "Vector after adding 5 and removing a number at position " << positionToRemove
<< ": ";
    for (vector<int>::iterator it = g2.begin(); it != g2.end(); ++it) {
        cout << *it << " ";
    }
    cout << endl;

    return 0;
}
```

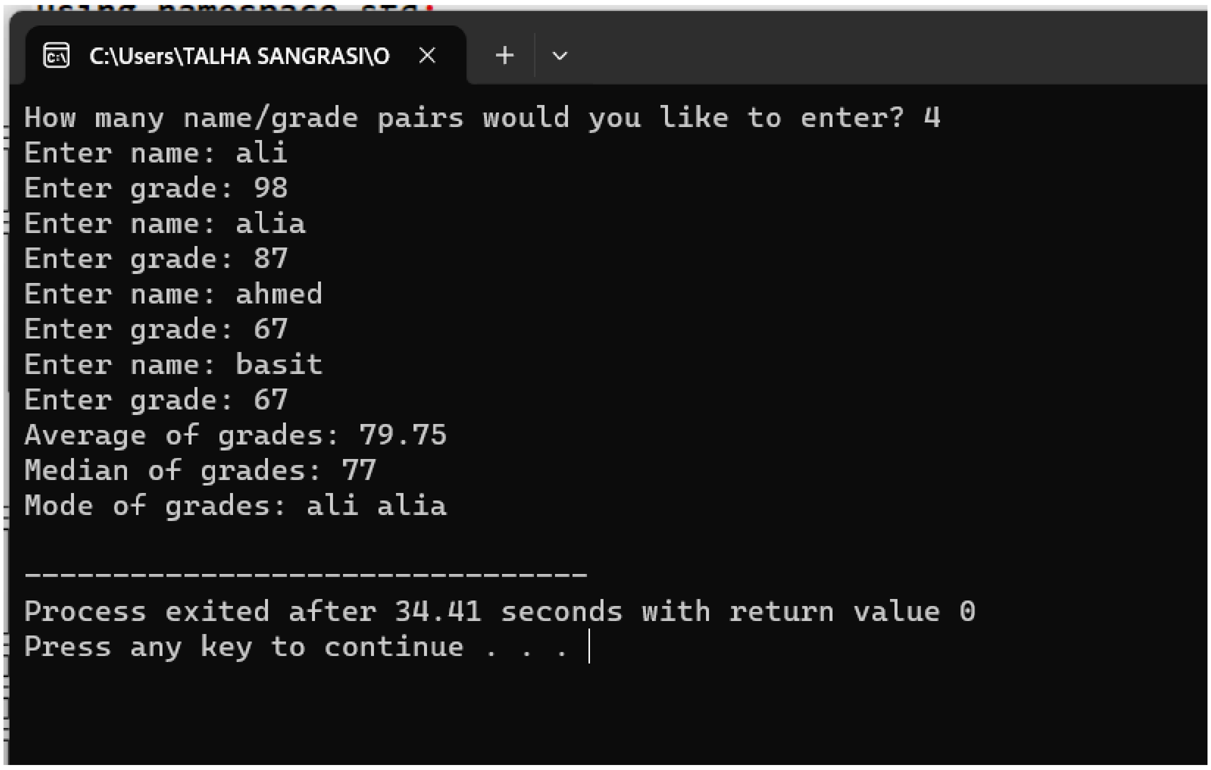


```
C:\Users\TALHA SANGRASI\O
Numbers in the vector: 10 20 30 40 50
Vector after adding 5 and removing a number at position 2: 10 20 40 50 5

--------------------------------
Process exited after 0.1583 seconds with return value 0
Press any key to continue . . .
```

Task 02

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

using namespace std;

// Function to calculate the mean of grades
double calculateMean(const vector<int>& studentGrades) {
    int sum = 0;
    for (size_t i = 0; i < studentGrades.size(); ++i) {
        sum += studentGrades[i];
    }
    return static_cast<double>(sum) / studentGrades.size();
}

// Function to calculate the median of grades without using <algorithm>
double calculateMedian(vector<int>& studentGrades) {
    size_t size = studentGrades.size();
    // Manual sorting using bubble sort
    for (size_t i = 0; i < size - 1; ++i) {
        for (size_t j = 0; j < size - i - 1; ++j) {
            if (studentGrades[j] > studentGrades[j + 1]) {
                // Swap studentGrades[j] and studentGrades[j+1]
                int temp = studentGrades[j];
                studentGrades[j] = studentGrades[j + 1];
                studentGrades[j + 1] = temp;
            }
        }
    }

    // Calculate median
    double median;
    if (size % 2 == 0) {
        median = static_cast<double>(studentGrades[size / 2 - 1] + studentGrades[size / 2]) / 2;
    } else {
        median = studentGrades[size / 2];
    }

    return median;
}
```

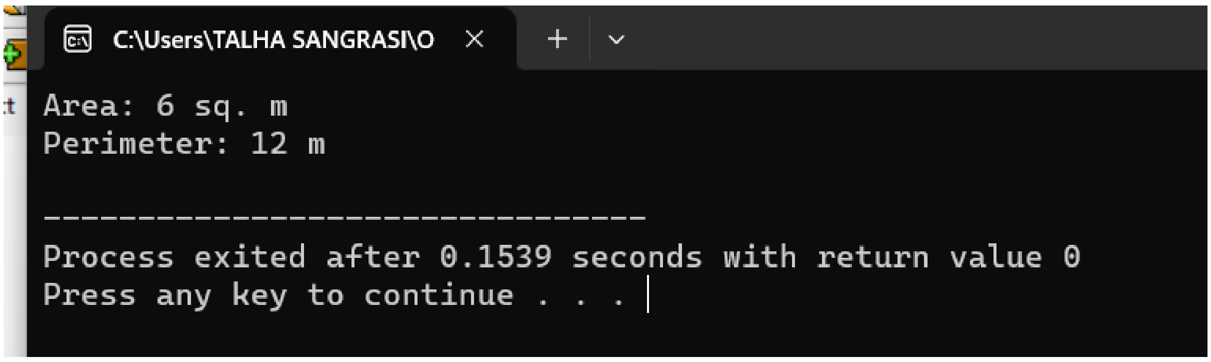
```cpp
// Function to calculate the mode without using <algorithm>
void calculateMode(const vector<int>& studentGrades, const vector<string>&
studentNames, vector<string>& modeNames) {
    int maxFrequency = 0;
    vector<int> frequency(101, 0); // Assuming grades are between 0 and 100

    for (size_t i = 0; i < studentGrades.size(); ++i) {
        frequency[studentGrades[i]]++;
        maxFrequency = max(maxFrequency, frequency[studentGrades[i]]);
    }

    for (size_t i = 0; i < studentGrades.size(); ++i) {
        if (frequency[studentGrades[i]] == maxFrequency) {
            modeNames.push_back(studentNames[i]);
        }
    }
}

int main() {
    int numPairs;
    cout << "How many name/grade pairs would you like to enter? ";
    cin >> numPairs;

    vector<string> studentNames(numPairs);
    vector<int> studentGrades(numPairs);

    // Input names and grades
    for (int i = 0; i < numPairs; i++) {
        cout << "Enter name: ";
        cin >> studentNames[i];
        cout << "Enter grade: ";
        cin >> studentGrades[i];
    }

    // Calculate and display average
    double average = calculateMean(studentGrades);
    cout << "Average of grades: " << average << endl;

    // Calculate and display median without using <algorithm>
    double median = calculateMedian(studentGrades);
    cout << "Median of grades: " << median << endl;

    // Calculate and display mode without using <algorithm>
    vector<string> modeNames;
    calculateMode(studentGrades, studentNames, modeNames);

    cout << "Mode of grades: ";
    for (size_t i = 0; i < modeNames.size(); ++i) {
        cout << modeNames[i];
        if (i < modeNames.size() - 1) {
            cout << " ";
        }
    }
    cout << endl;

    return 0;
}
```



```
How many name/grade pairs would you like to enter? 4
Enter name: ali
Enter grade: 98
Enter name: alia
Enter grade: 87
Enter name: ahmed
Enter grade: 67
Enter name: basit
Enter grade: 67
Average of grades: 79.75
Median of grades: 77
Mode of grades: ali alia

--------------------------------
Process exited after 34.41 seconds with return value 0
Press any key to continue . . .
```

Task 03

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

class Triangle {
private:
    double a, b, c;

public:
    Triangle(double a, double b, double c) {
        this->a = a;
        this->b = b;
        this->c = c;
    }
    void calculateAreaAndPerimeter() {
        double perimeter = a + b + c;
        double s = perimeter / 2;
        double area = sqrt(s * (s - a) * (s - b) * (s - c));

        std::cout << "Area: " << area << " sq. m" << std::endl;
        std::cout << "Perimeter: " << perimeter << " m" << std::endl;
    }
};

int main() {
    Triangle triangle(3, 4, 5);
    triangle.calculateAreaAndPerimeter();

    return 0;
}
```

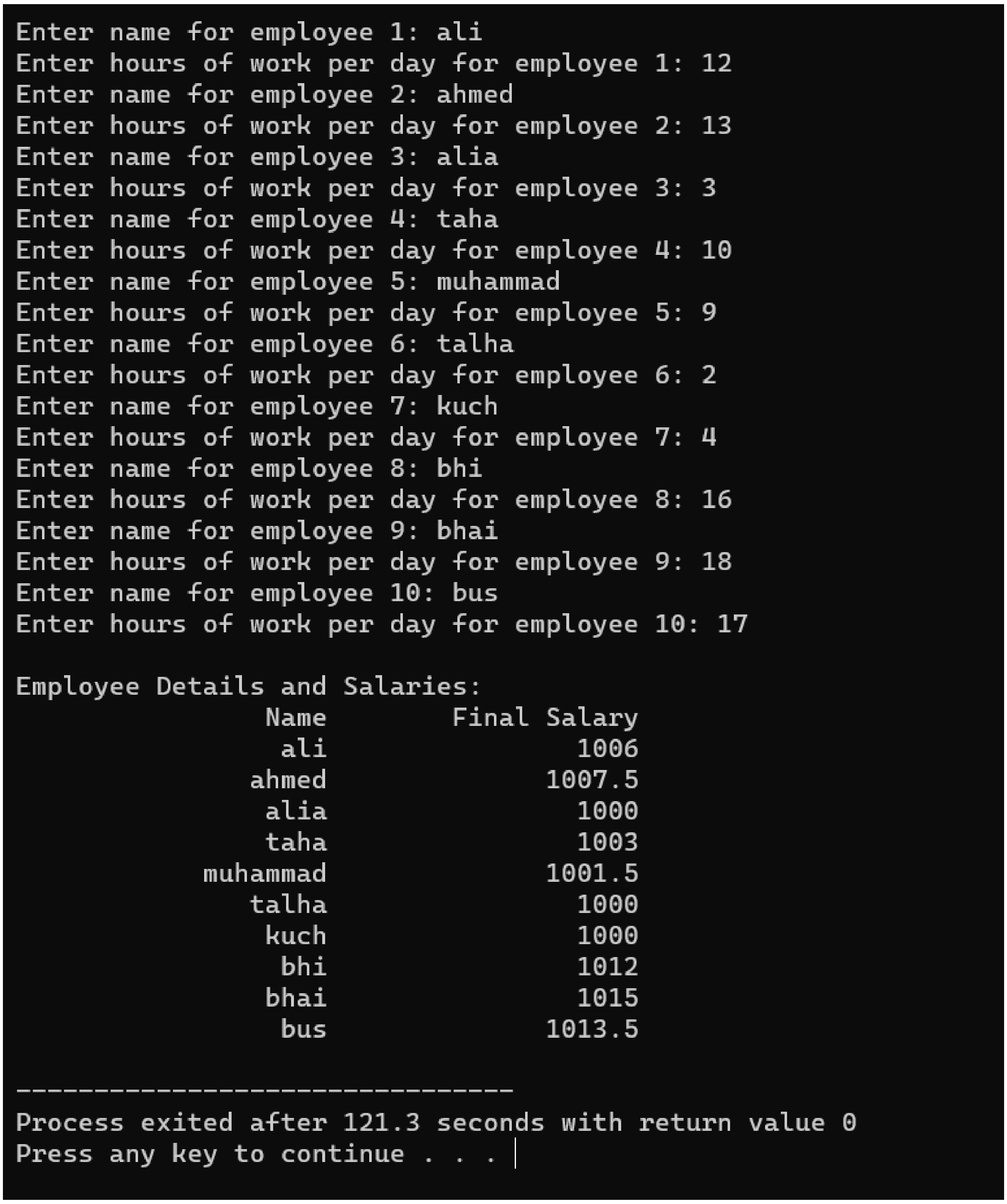


```
Area: 6 sq. m
Perimeter: 12 m

--------------------------------
Process exited after 0.1539 seconds with return value 0
Press any key to continue . . .
```

Task 04

```
#include <iostream>

#include <iomanip> // For setw function

#include <cstring> // For strcpy function


using namespace std;


const double OVERTIME_RATE = 1.5;

const double BASE_SALARY = 1000.0;


// Structure to store information about an employee

struct Employee {

   char name[50];

   double salary;

   int hoursWorked;

};


// Function to calculate the final salary based on hours worked

double calculateSalary(const Employee& emp) {

   double totalSalary = BASE_SALARY;

   if (emp.hoursWorked > 8) {

      int overtimeHours = emp.hoursWorked - 8;

      totalSalary += overtimeHours * OVERTIME_RATE;

   }

   return totalSalary;

}


int main() {

   const int numEmployees = 10;
```

```
    Employee employees[numEmployees];

    // Input information for each employee
    for (int i = 0; i < numEmployees; ++i) {
        cout << "Enter name for employee " << i + 1 << ": ";
        cin.getline(employees[i].name, sizeof(employees[i].name));
        cout << "Enter hours of work per day for employee " << i + 1 << ": ";
        cin >> employees[i].hoursWorked;
        cin.ignore(); // Ignore the newline character
    }

    // Increase salaries based on hours worked
    for (int i = 0; i < numEmployees; ++i) {
        employees[i].salary = calculateSalary(employees[i]);
    }

    // Print the names and final salaries of all employees
    cout << "\nEmployee Details and Salaries:\n";
    cout << setw(20) << "Name" << setw(20) << "Final Salary" << endl;
    for (int i = 0; i < numEmployees; ++i) {
        cout << setw(20) << employees[i].name << setw(20) << employees[i].salary << endl;
    }

    return 0;
}
```



```
Enter name for employee 1: ali
Enter hours of work per day for employee 1: 12
Enter name for employee 2: ahmed
Enter hours of work per day for employee 2: 13
Enter name for employee 3: alia
Enter hours of work per day for employee 3: 3
Enter name for employee 4: taha
Enter hours of work per day for employee 4: 10
Enter name for employee 5: muhammad
Enter hours of work per day for employee 5: 9
Enter name for employee 6: talha
Enter hours of work per day for employee 6: 2
Enter name for employee 7: kuch
Enter hours of work per day for employee 7: 4
Enter name for employee 8: bhi
Enter hours of work per day for employee 8: 16
Enter name for employee 9: bhai
Enter hours of work per day for employee 9: 18
Enter name for employee 10: bus
Enter hours of work per day for employee 10: 17

Employee Details and Salaries:
                Name        Final Salary
                 ali                1006
               ahmed              1007.5
                alia                1000
                taha                1003
            muhammad              1001.5
               talha                1000
                kuch                1000
                 bhi                1012
                bhai                1015
                 bus              1013.5

--------------------------------
Process exited after 121.3 seconds with return value 0
Press any key to continue . . .
```