BITS Pilani Hyderabad Campus

2nd Semester 2024-25 (CS F211: DSA) Lab-sheet 1

Practice the following C++ programs on a Linux machine:

Q.1:

a) Write a program in C++ to input two numbers and print their sum, difference, product, and quotient.

```
#include <iostream>
using namespace std;
int main() {
    double num1, num2;
    cout << "Enter the first number: ";
    cin >> num1;
    cout << "Enter the second number: ";
    cin >> num2;
    // Perform operations. Few places are left blank (?)for you to fill in.
    cout << "Sum: " << ? << endl;
    cout << "Difference: " << ? << endl;
    cout << "Product: " << ? << endl;

    // Handle division by zero
    if (num2 != 0) {
        cout << "Quotient: " << ? << endl;
    } else {
        cout << "Division by zero is not allowed." << endl;
    }

    return 0;
}
```

b) Write a C++ program to input a string from the user and output its length and the number of vowels in it.

```
#include <iostream>
#include <string>
using namespace std;
int main() {
    string str;
    int vowelCount = 0;
    cout << "Enter a string: ";
    getline(cin, str);
    // Count vowels. Few places are left blank (?)for you to fill in.
    for (char ch : str) {
        ch = tolower(ch);
        if (ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u')
{
            vowelCount?;
        }
    }

    // Output results
    cout << "Length of the string: " << str.length() << endl;
    cout << "Number of vowels: " << ? << endl;

    return 0;
}
```

Task: Using the standard C++ library "algorithm" (#include <algorithm>) that has an STL function reverse (reverse (Str.begin(), Str.end());), reverse an input string.

Q.2 Write a program in C++ to find out smallest and largest in an array of integers.

```
#include <iostream>
using namespace std;
int main() {
    int n;
    cout << "Enter the number of elements in the array: ";
    cin >> n;
```

```
    int arr[n];
    cout << "Enter the elements of the array: ";
    for (int i = 0; i < n; i++) {
        cin >> arr[i];
    }
    // Initialize smallest and largest element. Fill in the blanks (?).
    int smallest = ?;
    int largest = ?;

    for (int i = 1; i < n; i++) {
        if (arr[i] < smallest) {
                ? = arr[i];
        }
        if (arr[i] > largest) {
                ? = arr[i];
        }
    }
    cout << "Smallest element: " << smallest << endl;
    cout << "Largest element: " << largest << endl;
    return 0;
}
```

Task: Extend the above program to handle two edge cases, *Empty Array:* The program should not crash and should provide an appropriate message, *Array with Negative Numbers:* The program should handle negative values correctly and find the smallest and largest, even if they are negative.

Q.3 Write a C++ program that implements binary search to find whether a given number exists in a sorted array.

```
#include <iostream>
using namespace std;
// Binary Search Function
bool binarySearch(int arr[], int n, int target) {
    int low = 0, high = n - 1;
    while (low <= high) {
        int mid = low + (high - low) / 2;
        // Check if the target is at the middle. Fill in the blanks (?).
        if (arr[mid] == ?) {
            return true;
        }
        // If target is greater, ignore the left half
        if (arr[mid] < ?) {
            ? = mid + 1;
        }
        // If target is smaller, ignore the right half
        else {
            high = mid - 1;
        }
    }
    return false; // Target not found
}

int main() {
    int n, target;
    cout << "Enter the number of elements in the array: ";
    cin >> n;
    int arr[n];
    cout << "Enter the elements of the array (sorted): ";
    for (int i = 0; i < n; i++) {
        cin >> arr[i];
    }
    cout << "Enter the number to search: ";
    cin >> target;

    bool result = binarySearch(arr, n, target);
```

```
    // Output the result
    if (result) {
        cout << "Number " << target << " exists in the array." << endl;
    } else {
       cout << "Number"<< target<< " does not exist in the array." << endl;
    }
    return 0;
}
```

Task: Modify the above binary search function to return the *index* of the target when found. Return -1, if not found.

Q.4 Write a C++ code for the Student class that includes, a constructor to initialize the student's name, roll number, and marks; a destructor that displays a message like "Object Destroyed" when an object is destroyed; and input data for three students and display their details using a member function.

```
#include <iostream>
#include <string>
using namespace std;
// Student class
class Student {
private:
    string name;
    int rollNumber;
    float marks;
public:
    // Constructor to initialize student details
    Student(string n, int roll, float m) {
        name = n;
        rollNumber = roll;
        marks = m;
        cout << "Student object created for: " << name << endl;
    }

    // Destructor to display message when object is destroyed
    ~Student() {
        cout << "Object Destroyed for: " << name << endl;
    }
    // Member function to display student details
    void displayDetails() {
        cout << "Name: " << name << endl;
        cout << "Roll Number: " << rollNumber << endl;
        cout << "Marks: " << marks << endl;
    }
};
int main() {
    string name; int roll; float marks;
    Student student1("John", 101, 85.5);
    Student student2("Alice", 102, 90.0);
    Student student3("Bob", 103, 78.0);
    cout << "\nDetails of Student 1: " << endl;
    student1.displayDetails();
    cout << "\nDetails of Student 2: " << endl;
    student2.displayDetails();
    cout << "\nDetails of Student 3: " << endl;
    student3.displayDetails();
    return 0;
}
```

Task: Modify the above Student class program to include an additional member function updateMarks() that allows updating the marks of a student. This function should take the new marks as input and update the marks of the student accordingly.

Q.5 Write a C++ code that extends Q.4 (previous code) to include protected access modifier. The protected access modifier allows members to be accessible within the class, its derived classes, but not outside these classes.

```
#include <iostream>
#include <string>
using namespace std;
// Base class: Student
class Student {
protected: // Protected members
    string name;
    int rollNumber;
public:
    // Constructor to initialize the student
    Student(string n, int roll) {
        name = n;
        rollNumber = roll;
    }
    // Public function to display basic student details
    void displayBasicDetails() {
        cout << "Name: " << name << endl;
        cout << "Roll Number: " << rollNumber << endl;
    }
};
// Derived class: Result
class Result : public Student {
private:
    float marks;

public:
    // Constructor for Result class
    Result(string n, int roll, float m) : Student(n, roll) {
        marks = m;
    }
    // Function to display complete details
    void displayCompleteDetails() {
        // Accessing protected members from the base class
        cout << "Name: " << name << endl;
        cout << "Roll Number: " << rollNumber << endl;
        cout << "Marks: " << marks << endl;
    }
};

int main() {
    // Creating an object of the derived class
    Result student1("Alice", 101, 92.5);

    // Display basic details using the base class function
    student1.displayBasicDetails();

    cout << "\nComplete Details:\n";
    // Display complete details using the derived class function
    student1.displayCompleteDetails();

    return 0;
}
```

Task: Modify the above program to include another derived class called Sports that also inherits from the Student class. Add a private member sportsGrade (e.g., 'A', 'B', 'C') to the Sports class and a public method to set and display the sports grade. Display the combined details, e.g. For a student who participates in both academics (Result class) and sports (Sports class), display their name, roll number, marks, and sports grade.

Refs: cplusplus.com/doc/tutorial/
