Shri Shankaracharya Institute of
Professional Management and Technology
Department of Computer Science and Engineering

Project for C/C++ CLASS

B-Tech (CSE) – 4th Semester

Shri Shankaracharya Institute of Professional Management and Technology
P.O. Sejbahar, Mujgahan, Raipur (C.G.)
Phone No. 0771-2120666, 2120777 Fax. 0771-2120555
E-Mail: info@ssipmt.com

SSIPMT RAIPUR	**Shri Shankaracharya** **Institute of Professional Management and Technology**	
	Department of Computer Science and Engineering	
Class: B. Tech 4th Semester	**Subject: C/C++**	

Title
Problem Title: C++ Code Quiz App for Learners

Submitted by:

1. Shashwat Khandelwal (197)
2. Shashvat singh (196)
3. Yogendra Singh (244)

Professor In-Charge

Saurabh Vashisth

INTRODUCTION:-

The C++ Code Quiz App is an educational tool designed to test and improve a learner's understanding of C++ programming concepts.
It exists in two complementary versions:

1. **Console-Based CLI Version (C++):**
   A lightweight, terminal-driven quiz system developed entirely in C++, allowing learners to answer randomized multiple-choice questions with hints, explanations, and performance feedback.

2. **Web-Based Interactive Version (HTML/CSS/JavaScript):**
   A visually appealing browser-based interface with progress bars, colored feedback, timed questions, hints, explanations, and topic-wise performance charts for a richer learning experience.

Both versions share the same goal – to promote self-assessment, interactive learning, and consistent practice in a gamified format.

DESIGN

Core Components

Data Members (CLI Version – `struct Question`):

- **`question`: String storing the quiz question.**
- **`options[4]`: Array holding four possible answers.**
- **`correctOption`: Character ('1'-'4') indicating the correct answer.**
- **`timeLimit`: Integer specifying time allowed per question.**
- **`hint`: String containing a short clue for the answer.**

- **`explanation`: String providing detailed reasoning.**
- **`topic`: String indicating the category of the question.**

Data Structure (Web Version – `questions` Array):

- **Each question is stored as an object with:**
  - **`question`, `options[]`, `correctOption`, `timeLimit`, `hint`, `explanation`, `topic`.**

Methods & Functional Modules

1. **getQuestions():**
   Returns a hardcoded vector/array of C++ quiz questions.
2. **displayQuestion():**
   CLI – Prints the question and options in the console.
   Web – Dynamically generates HTML elements to show questions and options.
3. **Answer Validation:**
   Checks user's choice, marks correct/incorrect answers, updates scores, and displays explanations.
4. **Timing Mechanism:**
   - **CLI – Uses `chrono` to track elapsed seconds for each question.**
   - **Web – Uses JavaScript's `setInterval()` to display a countdown**

and auto-lock when time expires.

5. **Performance Tracking:**
    - Topic-wise analysis stored in a `map` (CLI) or object (Web).
    - Calculates percentage accuracy for each topic.

IMPLEMENTATION MECHANISM

CLI Version (main.cpp)

- **Language:** C++ (Standard Library + `<chrono>` for timing).
- **Randomization:** `shuffle()` from `<algorithm>` with a random seed.
- **Encapsulation:** `struct Question` holds all data related to a single quiz item.
- **Interactive I/O:** `cin`/`cout` for question display and user input.
- **Hint System:** Allows user to enter 'h' for hints during a question.

Web Version (index.html + JavaScript)

- **Interface:** HTML for structure, CSS for styling, JavaScript for quiz logic.
- **Dynamic Rendering:** DOM manipulation to load questions and answers interactively.
- **Enhanced UX:**

- **Progress bar.**
- **Option highlighting (selected, correct, incorrect).**
- **Real-time countdown per question.**
- **Topic-wise colored progress bars.**
- **Feedback emojis and personalized results.**

- **No Dependencies: Runs in any modern browser without installations.**

FEATURES

Common to Both Versions:

1. **Hardcoded Question Bank of 20 curated C++ questions.**
2. **Randomized Order of questions in each session.**
3. **Score Calculation & Feedback based on performance.**
4. **Topic-Wise Analysis to identify strong and weak areas.**

Exclusive to CLI Version:

- **Fully terminal-based (lightweight, no GUI dependency).**
- **Keyboard-based input system with real-time hints.**

Exclusive to Web Version:

- **Graphical user interface with colors, layouts, and animations.**
- **Visual progress indicators.**
- **Interactive option selection and auto-locking after time expiry.**

FUTURE ENHANCEMENTS

1. **Database Integration: Store and retrieve questions dynamically.**
2. **User Profiles: Track performance over multiple sessions.**
3. **Difficulty Levels: Categorize questions into Easy, Medium, Hard.**
4. **PDF Report Generation: Export results for review or submission.**
5. **GUI for CLI Version: Integrate Qt/SFML for a native app feel.**
6. **Leaderboard System: For competitive learning in classrooms.**

CONCLUSION

The C++ Code Quiz App demonstrates how a single educational concept can be implemented in two complementary environments –
a console-based C++ version for lightweight use and an interactive web version for engaging visuals.

Both implementations share the same quiz logic and educational objectives but

differ in user experience.
The project effectively:

- **Encourages consistent programming practice.**
- **Reinforces C++ fundamentals.**
- **Provides immediate feedback and detailed explanations.**
- **Supports topic-wise performance tracking for targeted improvement.**

With its modular design and extendable architecture, this app serves as a foundation for more advanced learning platforms in programming education.

SOURCE CODE

```
#include <iostream>
#include <vector>
#include <cstdlib>
#include <ctime>
#include <algorithm>
#include <limits>
```

```cpp
#include <chrono>
#include <random>
#include <map>

using namespace std;

struct Question {
    string question;
    string options[4];
    char correctOption;
    int timeLimit;
    string hint;
    string explanation;
    string topic;
};

vector<Question> getQuestions() {
    return {
        // Basic Questions
        {"Which of the following is a correct identifier in C++?",
         {"1variable", "variable_1", "variable-1", "variable 1"}, '2',
         20, "Consider C++ variable naming rules",
         "In C++, identifiers can't start with numbers or contain spaces/hyphens",
         "Basic Syntax"},

        {"Which keyword is used to define a class in C++?",
         {"def", "function", "class", "object"}, '3',
```

```
 25, "It's the same as the concept it defines",
 "The 'class' keyword is used to define classes in C++",
 "OOP Concepts"},

{"Which operator is used to allocate memory dynamically in C++?",
 {"malloc", "alloc", "new", "make"}, '3',
 30, "Think about C++ specific operators",
 "'new' is the C++ operator for dynamic memory allocation",
 "Memory Management"},

{"Which of the following is not a C++ data type?",
 {"float", "bool", "real", "char"}, '3',
 15, "Which one doesn't appear in primitive type lists?",
 "'real' is not a C++ primitive data type (use float/double instead)",
 "Data Types"},

{"What is the size of int in C++ (typically)?",
 {"2 bytes", "4 bytes", "8 bytes", "16 bytes"}, '2',
 20, "Think about common architectures",
 "On most modern systems, int is typically 4 bytes",
 "Data Types"},

{"Which of the following is used to take input in C++?",
 {"cin", "cout", "input", "scanf"}, '1',
 15, "Which one is the input stream?",
 "'cin' is the standard input stream object",
 "Basic I/O"},
```

```
{"Which is the correct syntax to define a function in C++?",
 {"function myFunc()", "void myFunc()", "myFunc void()", "def myFunc()"}, '2',
 25, "Consider return type placement",
 "C++ functions specify return type before the function name",
 "Functions"},

{"Which header file is needed for input/output in C++?",
 {"iostream", "stdio.h", "conio.h", "fileio.h"}, '1',
 15, "Which is the standard C++ header?",
 "<iostream> is the standard C++ header for I/O operations",
 "Basic I/O"},

{"What is the scope resolution operator?",
 {"::", "..", "--", "->"}, '1',
 20, "Think about namespace and class scope",
 "'::' is used to access global scope or class members",
 "Operators"},

{"What is the output of: cout << 5 / 2;",
 {"2.5", "2", "2.0", "Error"}, '2',
 25, "Consider integer division",
 "Integer division truncates the decimal part",
 "Operators"},

// Advanced Questions
{"What is the purpose of a virtual destructor in C++?",
```

```
         {"To prevent memory leaks when deleting polymorphic objects",
          "To make a class abstract",
          "To enable multiple inheritance",
          "To improve performance"}, '1',
         25, "Think about polymorphism and object destruction",
         "Virtual destructors ensure proper cleanup when deleting objects through base class
pointers",
         "OOP Concepts"},


        {"What does RAII stand for in C++?",
         {"Resource Allocation Is Initialization",
          "Runtime Allocation of Internal Instances",
          "Recursive Algorithm Implementation",
          "Random Access Iterator Interface"}, '1',
         20, "It's a fundamental C++ idiom",
         "RAII ties resource management to object lifetime",
         "Memory Management"},


        {"Which STL container provides O(1) average complexity for insertions and deletions?",
         {"std::vector",
          "std::list",
          "std::unordered_map",
          "std::set"}, '2',
         25, "Think about linked data structures",
         "std::list (doubly-linked list) provides O(1) insertions/deletions",
         "STL Containers"},
```

```
{"What is the output of: std::cout << sizeof('a'); in C++?",
 {"1", "2", "4", "Depends on the compiler"}, '1',
 15, "Character literals in C++ vs C",
 "In C++, character literals are of type char (1 byte), unlike in C",
 "Data Types"},

{"Which keyword is used to prevent method overriding in a derived class?",
 {"static", "const", "final", "sealed"}, '3',
 20, "Introduced in C++11",
 "The 'final' keyword prevents further overriding of virtual methods",
 "OOP Concepts"},

{"What is the purpose of std::move?",
 {"To physically move an object in memory",
  "To transfer ownership of resources",
  "To copy an object efficiently",
  "To change an object's type"}, '2',
 25, "Part of move semantics",
 "std::move casts to rvalue reference, enabling resource transfer",
 "Move Semantics"},

{"Which of these is not a C++ smart pointer?",
 {"std::unique_ptr",
  "std::shared_ptr",
  "std::auto_ptr",
  "std::raw_ptr"}, '4',
 20, "Deprecated pointers vs non-existent ones",
```

```
         "std::raw_ptr doesn't exist (though std::auto_ptr is deprecated)",
         "Memory Management"},

        {"Which C++17 feature allows accessing multiple return values more easily?",
         {"auto keyword",
          "structured bindings",
          "template deduction",
          "lambda captures"}, '2',
         25, "Works with tuples and structs",
         "Structured bindings allow auto [x,y] = func_returning_pair();",
         "Modern C++"}
    };
}

void displayQuestion(const Question& q, int number) {
    cout << "\nQ" << number << ". " << q.question << "\n";
    for (int i = 0; i < 4; i++) {
        cout << "  " << i+1 << ") " << q.options[i] << "\n";
    }
    cout << "Time limit: " << q.timeLimit << " seconds\n";
    cout << "Enter your answer (1-4) or 'h' for hint: ";
}

int main() {
    auto start = chrono::high_resolution_clock::now();
    vector<Question> questions = getQuestions();
    int score = 0, total = questions.size();
```

```
    map<string, pair<int, int>> topicPerformance;

    unsigned seed = static_cast<unsigned>(time(0));
    shuffle(questions.begin(), questions.end(), default_random_engine(seed));

    cout << "======== C++ CODE QUIZ APP ========\n";
    cout << "  Answer each question (1-4)\n";
    cout << "  Type 'h' for hint during question\n";
    cout << "===================================\n";

    for (int i = 0; i < total; ++i) {
        displayQuestion(questions[i], i + 1);

        auto questionStart = chrono::high_resolution_clock::now();
        char answer = ' ';
        bool timeExpired = false;

        // Non-threaded timer implementation
        while (true) {
            auto currentTime = chrono::high_resolution_clock::now();
            auto elapsed = chrono::duration_cast<chrono::seconds>(currentTime -
questionStart).count();

            if (elapsed >= questions[i].timeLimit) {
                cout << "\n⏰ Time's up! The correct answer was: " <<
questions[i].correctOption << "\n";
                timeExpired = true;
```

```
            break;
        }

        if (cin.peek() != EOF) {
            cin >> answer;

            if (answer == 'h' || answer == 'H') {
                cout << "💡 Hint: " << questions[i].hint << "\n";
                cout << "Enter your answer (1-4): ";
                cin >> answer;
            }

            if (answer >= '1' && answer <= '4') {
                break;
            } else {
                cin.clear();
                cin.ignore(numeric_limits<streamsize>::max(), '\n');
                cout << "❗ Invalid input. Please enter 1-4: ";
            }
        }
    }

    if (!timeExpired) {
        if (answer == questions[i].correctOption) {
            cout << "✅ Correct! " << questions[i].explanation << "\n";
            ++score;
        } else {
```

```
            cout << "❌ Wrong! Correct Answer: Option " << questions[i].correctOption
                 << "\nExplanation: " << questions[i].explanation << "\n";
        }
    }

    string topic = questions[i].topic;
    topicPerformance[topic].first += (answer == questions[i].correctOption);
    topicPerformance[topic].second += 1;
}

auto end = chrono::high_resolution_clock::now();
auto duration = chrono::duration_cast<chrono::seconds>(end - start);

cout << "\n========== RESULTS ==========\n";
cout << " Your Score: " << score << " / " << total << "\n";
float percent = (score * 100.0f) / total;
cout << " Percentage: " << percent << "%\n";
cout << " Time Taken: " << duration.count() << " seconds\n";

if (percent == 100)
    cout << "🌟 Excellent! Perfect score!\n";
else if (percent >= 75)
    cout << "🎉 Great job!\n";
else if (percent >= 50)
    cout << "👍 Good effort. Keep learning.\n";
else
    cout << "📚 Needs improvement. Try again!\n";
```

```
    cout << "\n======== PERFORMANCE ANALYSIS ========\n";
    for (const auto& entry : topicPerformance) {
        double topicPercent = (entry.second.first * 100.0) / entry.second.second;
        cout << entry.first << ": " << entry.second.first << "/" << entry.second.second
             << " (" << topicPercent << "%)" << endl;
        if (topicPercent < 50) {
            cout << "   ➡ Focus more on this topic!\n";
        }
    }

    return 0;
}
```

RESULT / OUTPUT

The program will produce the following outputs based on user input during the quiz:

- For each question, the program displays the question text along with four options.
- The user enters a number (**1–4**) as their selected answer.
- The app outputs “✅ **Correct!**” if the selected answer matches the correct one.
- Otherwise, it displays “❌ **Wrong! Correct Answer: Option X**”.
- After all questions are answered, a **result summary** is shown.

🔍 **Final Result Summary Includes:**

- **Total Score** out of 20
- **Percentage (%)**
- **Performance Feedback:**
  - 🌟 **Perfect score** → “Excellent!”
  - 🎉 **75%+** → “Great job!”
  - 👍 **50%+** → “Good effort”
  - 📚 **Below 50%** → “Needs improvement”