

Experiment No. 15

Aim: Create a mini project using C++ programming for solving real time problem.

Code:

```
#include <iostream>
#include <fstream>
#include <string>
#include <vector>
#include <cctype>
#include <ctime>
#include <algorithm>
using namespace std;

static string trim(const string& s) {
    size_t a = s.find_first_not_of(" \t\r\n");
    if (a == string::npos) return "";
    size_t b = s.find_last_not_of(" \t\r\n");
    return s.substr(a, b - a + 1);
}

static string getName() {
    string name;
    while (true) {
        cout << "Enter your name: ";
        if (!getline(cin, name)) return "Player";
        name = trim(name);
        if (!name.empty()) return name;
        cout << "Name cannot be empty. Please try again.\n";
    }
}

static void saveSession(const string& name) {
    ofstream out("player_session.txt", ios::out);
    if (!out) return;
    time_t now = time(nullptr);
    out << "name=" << name << "\n";
    out << "timestamp=" << (long long)now << "\n";
}

struct Q { string t,a,b,c,d; char correct; };

static bool loadQs(const string& path, vector<Q>& qs) {
    ifstream in(path);
    if (!in) return false;
    string line, ans;
    while (true) {
        Q q;
        if (!getline(in, q.t)) break;
        q.t = trim(q.t);
        if (q.t.empty()) continue;
        if (!getline(in, q.a) || !getline(in, q.b) || !getline(in, q.c) || !getline(in, q.d)) break;
        if (!getline(in, ans)) break;
        ans = trim(ans);
```

```

        if (ans.empty()) break;
        q.correct = (char)toupper((unsigned char)ans[0]);
        getline(in, line);
        qs.push_back(q);
    }
    return !qs.empty();
}

```

```

static void playRound(const vector<Q>& qs, size_t startIndex, size_t roundSize, int& roundScore,
int& asked) {
    roundScore = 0;
    asked = 0;
    size_t endIndex = min(startIndex + roundSize, qs.size());
    for (size_t i = startIndex; i < endIndex; ++i) {
        const auto& q = qs[i];
        cout << "\n" << q.t << "\n";
        cout << "A) " << q.a << "\nB) " << q.b << "\nC) " << q.c << "\nD) " << q.d << "\n";
        string s; char ans = 0;
        while (true) {
            cout << "Your answer (A/B/C/D): ";
            if (!getline(cin, s)) return;
            s = trim(s);
            if (!s.empty()) {
                ans = (char)toupper((unsigned char)s[0]);
                if (ans=='A' || ans=='B' || ans=='C' || ans=='D') break;
            }
            cout << "Please enter A, B, C, or D.\n";
        }
        if (ans==q.correct) { ++roundScore; cout << "Correct!\n"; }
        else { cout << "Wrong. Correct answer: " << q.correct << "\n"; }
        ++asked;
    }
}

```

```

int main() {
    cout << "===== \n";
    cout << "    QUIZ APP    \n";
    cout << "===== \n\n";

```

```

    string name = getName();
    saveSession(name);

```

```

    vector<Q> qs;
    if (!loadQs("questions.txt", qs)) {
        cerr << "Error: Could not load questions from questions.txt or file is empty.\n";
        return 1;
    }

```

```

    const size_t ROUND_SIZE = 10;
    size_t cursor = 0; // Next question index to serve
    int totalScore = 0;
    size_t totalAsked = 0;

```

```

    // First run: start the first round immediately (no Home menu yet)
    if (cursor < qs.size()) {

```

```

cout << "\n";

int roundScore = 0; int asked = 0;
playRound(qs, cursor, ROUND_SIZE, roundScore, asked);
cursor += asked;
totalScore += roundScore;
totalAsked += asked;

cout << "\nCompleted!\n";
cout << "Total Score: " << roundScore << "/" << asked << "\n";

ofstream out("results.txt", ios::app);
if (out) out << name << ", " << (long long)time(nullptr) << ", " << roundScore << ", " << asked <<
"\n";

cout << "Press Enter to go to Home...";
string tmp; getline(cin, tmp);
}

while (true) {
    cout << "\n===== HOME =====\n";
    cout << "Hello, " << name << "\n";
    cout << "A) Start Quiz \n";
    cout << "B) Restart from beginning\n";
    cout << "Q) Quit\n";
    cout << "Choose (A/B/Q): ";
    string choice;
    if (!getline(cin, choice)) return 0;
    choice = trim(choice);
    char ch = choice.empty()? '\0' : (char)toupper((unsigned char)choice[0]);

    if (ch == 'Q') {
        break;
    } else if (ch == 'B') {
        cursor = 0;
        totalScore = 0;
        totalAsked = 0;
        cout << "Progress reset.\n";
        continue;
    } else if (ch == 'A') {
        if (cursor >= qs.size()) {
            cout << "All questions completed. Choose 'B' to restart or 'Q' to quit.\n";
            continue;
        }

        cout << "\n";

        int roundScore = 0; int asked = 0;
        playRound(qs, cursor, ROUND_SIZE, roundScore, asked);
        cursor += asked;
        totalScore += roundScore;

        cout << "\nCompleted!\n";

```

```

cout << "Total Score: " << roundScore << "/" << asked << "\n";

ofstream out("results.txt", ios::app);
if (out) out << name << ", " << (long long)time(nullptr) << ", " << roundScore << ", " << asked
<< "\n";

    cout << "Press Enter to return to Home...";
    string tmp; getline(cin, tmp);
    continue;
} else {
    cout << "Please choose A, B, or Q.\n";
}
}

cout<<"\nCONGRATS!!!";
cout<<"\nCompleted ";
return 0;
}

```