```
/**
* File Name: project4_Senn_abs0113.cpp
* Author: Blake Senn (abs0113)
*
* Compiled with MinGW GCC C++ compiler.
*
* I used the hints in Canvas and previous
* coding knowledge as sources.
*/

#include <iostream>
#include <string>
#include <stdlib.h>
#include <assert.h>
using namespace std;

struct Trivia {
    // Node Structure
    struct TriviaNode {
        string question;
        string answer;
        int value;
        TriviaNode *next;

        TriviaNode(string question_in, string answer_in, int value_in) {
            question = question_in;
            answer = answer_in;
            value = value_in;
            next = nullptr;
        }
    };

    // Iterator
    struct TriviaIterator {
        TriviaNode *current;

        TriviaIterator(TriviaNode *node_in) {
            current = new TriviaNode("", "", 0);
            current->next = node_in;
        }

        bool has_next() {
            return current->next != nullptr;
        }

        TriviaNode next() {
            current = current->next;
            return (*current);
        }
    };

    // List Variables
    TriviaNode *first;
    TriviaNode *last;
    int size;
    int score;

    Trivia() {
        first = nullptr;
        last = nullptr;
        size = 0;
        score = 0;
    }

    bool add(string question_in, string answer_in, int value_in) {
        TriviaNode* new_node = new TriviaNode(question_in, answer_in, value_in);
        if (size == 0) {
            first = new_node;
        }
        else {
            last->next = new_node;
        }
        last = new_node;
        size++;
        return true;
    }

    TriviaIterator iterator() {
        return TriviaIterator(first);
    }

    // Prompt user to add a question
    void add_question() {
        string new_question;
        string new_answer;
        string value_response;
        int new_value;
        cout << "\nEnter a question: ";
        getline(cin, new_question);
        cout << "Enter an answer: ";
        getline(cin, new_answer);
        cout << "Enter award points: ";
        getline(cin, value_response);
        new_value = stoi(value_response);
        add(new_question, new_answer, new_value);
    }

    // Ask the specified number of questions
    bool ask_question(int num_ask) {
        if (num_ask < 1) {
            cout << "Warning - the number of trivia to be asked must equal to or be
larger than 1." << endl;
            return false;
        }
        else if (size < num_ask) {
            cout << "Warning - there (is/are) only " << size << " trivia
question(s) in the list." << endl;
            return false;
        }
        else {
            score = 0;
            TriviaIterator itr = iterator();
            int count = 0;
            while (itr.has_next() && count < num_ask) {
                TriviaNode current = itr.next();
                string user_answer;
                cout << "\nQuestion: " << current.question;
                cout << "\nAnswer: ";
                getline(cin, user_answer);
                if (user_answer.compare(current.answer) == 0) {
                    cout << "Your answer is correct. You receive " << current.value
<< " points." << endl;
                    score += current.value;
                }
                else {
                    cout << "Your answer is wrong. The correct answer is: " <<
current.answer << endl;
                }
                cout << "Your total points: " << score << endl;
                count++;
            }
            return true;
        }
    }

    // Initialize a list with some example questions. Currently accepted versions
are 0 and 1
    void initial_questions(int version) {
        size = 0;
        score = 0;
        if (version > -1) {  // Version 0 selects this list only
            add("How long was the shortest war on record? (Hint: how many
minutes)", "38", 100);
        }
        if (version > 0) {  // Version 1 adds this block and the previous
            add("What was Bank of America's original name? (Hint: Bank of Italy or
Bank of Germany)?", "Bank of Germany", 50);
            add("What is the best-selling video game of all time? (Hint: Call of
Duty or Wii Sports)?", "Wii Sports", 20);
        }
    }
};

// Conditional Compilation
#define trivia_quiz
//#define UNIT_TESTING

int main() {
    Trivia game = Trivia();

    #ifdef trivia_quiz
        cout << "*** Welcome to Blake's trivia quiz game ***" << endl;

        bool add_more = false;
        do {
            game.add_question();
            string user_response = "";
            while (user_response == "") {
                cout << "Continue? (y/n): ";
                getline(cin, user_response);
                if (user_response.at(0) == 'y' || user_response.at(0) == 'Y') {
                    add_more = true;
                }
                else if (user_response.at(0) == 'n' || user_response.at(0) == 'N')
{
                    add_more = false;
                }
                else {
                    cout << "Invalid Response" << endl;
                    user_response = "";
                }
            }
        } while (add_more);

        cout << "\n\nWelcome to the Trivia Game!" << endl;
        game.ask_question(game.size);

        cout << "\n*** Thank you for playing the trivia quiz game. Goodbye! ***";
    #endif

    #ifdef UNIT_TESTING
        cout << "*** This is a debugging version ***" << endl;
        cout << "Unit Test Case 1: Ask no question. The program should give a
warning message." << endl;
        bool warning = game.ask_question(0);
        assert(!warning);
        cout << "\nCase 1 Passed" << endl << endl;

        game.initial_questions(1);
        cout << "Unit Test Case 2.1: Ask 1 question in the linked list. The tester
enters an incorrect answer." << endl;
        game.ask_question(1);
        assert(game.score == 0);
        cout << "\nCase 2.1 passed" << endl << endl;

        cout << "Unit Test Case 2.2: Ask 1 question in the linked list. The tester
enters a correct answer." << endl;
        game.ask_question(1);
        assert(game.score == 100);
        cout << "\nCase 2.2 passed" << endl << endl;

        cout << "Unit Test Case 3: Ask all the questions in the linked list." <<
endl;
        bool played = game.ask_question(3);
        assert(played);
        cout << "\nCase 3 passed" << endl << endl;

        cout << "Unit Test Case 4: Ask 5 questions in the linked list." << endl;
        warning = game.ask_question(5);
        assert(!warning);
        cout << "\nCase 4 passed" << endl << endl;

        cout << "*** End of debugging version ***";
    #endif

    return 0;
}
```