Project 5 Test Routine

We took your Project 5 source file, and made the following tranformations:

- Rename `main` to `xxxmain`.
- Rename `loadWords` to `xxxloadWords`.
- Rename `manageOneRound` to `xxxmanageOneRound`.

We then appended the following code to your file and ran the 21 test cases. Each test case was worth 4 points, and you got 1 point for turning something in.

```
#include <iostream>
#include <string>
#include <sstream>
#include <algorithm>
#include <cstring>
#include <cstdlib>

using namespace std;

int xxxtestno;

void addWord(char words[][7], int maxWords, int& num, const char w[])
{
        if (num < maxWords)
                strcpy(words[num++], w);
}

int loadWords(char words[][7], int maxWords)
{
        int num = 0;
        switch (xxxtestno)
        {
            default:
               break;
            case 1:
            case 2:
            case 3:
               addWord(words, maxWords, num, "hello");
               break;
            case 4:
               addWord(words, maxWords, num, "hello");
               addWord(words, maxWords, num, "hello");
               num--;
               break;
            case 5:
            case 6:
            case 7:
            case 8:
            case 9:
            case 10:
            case 11:
               addWord(words, maxWords, num, "wowie");
```

```
                addWord(words, maxWords, num, "hello");
                break;
            case 12:
                addWord(words, maxWords, num, "daisy");
                addWord(words, maxWords, num, "hello");
                break;
            case 13:
                addWord(words, maxWords, num, "lurid");
                addWord(words, maxWords, num, "hello");
                break;
            case 14:
                addWord(words, maxWords, num, "lulls");
                addWord(words, maxWords, num, "hello");
                break;
            case 15:
                addWord(words, maxWords, num, "stare");
                addWord(words, maxWords, num, "aster");
                break;
            case 16:
                addWord(words, maxWords, num, "lone");
                addWord(words, maxWords, num, "hello");
                break;
            case 17:
                addWord(words, maxWords, num, "lonely");
                addWord(words, maxWords, num, "hello");
                break;
            case 18:
                addWord(words, maxWords, num, "lone");
                addWord(words, maxWords, num, "hello");
                break;
            case 19:
                addWord(words, maxWords, num, "lonely");
                addWord(words, maxWords, num, "hello");
                break;
            case 20:
                addWord(words, maxWords, num, "level");
                addWord(words, maxWords, num, "hello");
                break;
            case 21:
                for (int k = 0; k < 10; k++)
                        addWord(words, maxWords, num, "hello");
                break;
        }
        return num;
}

int manageOneRound(char words[][7], int num, int wordnum)
{
        switch (xxxtestno)
        {
            default:
                return xxxmanageOneRound(words, num, wordnum);
            case 2:
                return 42;
            case 3:
                {
                static const size_t NUM_SCORES = 4;
                static int scores[NUM_SCORES] = { 4, 6, 3, 8 };
                static int r = NUM_SCORES - 1;
                if (++r == NUM_SCORES)
                        r = 0;
                return scores[r];
                }
            case 21:
                {
                static int counts[10] = { 0 };
                static int errors = 0;
                if (wordnum == 999999)
                {
                        if (errors > 0)
                                cout << "ERRORS: " << errors << endl;
                        bool ok = true;
                        for (int k = 0; k < 10; k++)
                                if (counts[k]-100 < -25  ||  counts[k]-100 >
25)
                                {
                                        ok = false;
                                        break;
                                }
                        if (ok)
                                cout << "CORRECT";
                        else
                        {
                                for (int k = 0; k < 10; k++)
                                        cout << counts[k] << " ";
                        }
                }
                else if (wordnum < 0  ||  wordnum >= 10)
                        errors++;
                else
                        counts[wordnum]++;
                return 1;
                }
        }
}

char xxxinput[][100] = {
        "",
        /*  1 */ "0\n1\nhello\n",
        /*  2 */ "1\n",
        /*  3 */ "4\n",
        /*  4 */ "hello\n",
        /*  5 */ "hello\n",
        /*  6 */ "wowie\nhello\n",
        /*  7 */ "he@lo\nweird\nhello\n",
        /*  8 */ "he@lo\nhello\n",
        /*  9 */ "abc\nhello\n",
        /* 10 */
"abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwx
yz\nhello\n",
        /* 11 */ "gleet\nhello\n",
        /* 12 */ "daisy\nhello\n",
        /* 13 */ "lurid\nhello\n",
        /* 14 */ "lulls\nhello\n",
        /* 15 */ "stare\naster\n",
        /* 16 */ "lone\nhello\n",
        /* 17 */ "lonely\nhello\n",
        /* 18 */ "hello\nlone\n",
        /* 19 */ "hello\nlonely\n",
        /* 20 */ "level\nlevel\nhello\nhello\nhello\nlevel\n",
        /* 21 */ "1000\n",
};

int main()
{
        cout << "Enter test number (1-21): ";
        cin >> xxxtestno;

        char wordList[9000][7];

         istringstream iss(xxxinput[xxxtestno]);
         streambuf* isb = cin.rdbuf(iss.rdbuf());

        switch (xxxtestno)
        {
                default: {
            return 1;
                } break; case 1: {     // 0 rounds
            xxxmain();
                } break; case 2: {     // stats for one game
            xxxmain();
                } break; case 3: {     // stats for several games
            xxxmain();
                } break; case 4: {     // bad arg to manageOneRound
            int nWords = loadWords(wordList, 10);
            int n = xxxmanageOneRound(wordList, nWords, nWords);
            cout << (n == -1 ? "CORRECT" : "WRONG");
                } break; case 5: {     // guessed in one round
            int nWords = loadWords(wordList, 10);
            int n = xxxmanageOneRound(wordList, nWords, 1);
            cout << (n == 1 ? "CORRECT" : "WRONG");
                } break; case 6: {     // guessed in two rounds
            int nWords = loadWords(wordList, 10);
            int n = xxxmanageOneRound(wordList, nWords, 1);
            cout << (n == 2 ? "CORRECT" : "WRONG");
                } break; case 7: {     // errors count as guesses
            int nWords = loadWords(wordList, 10);
            int n = xxxmanageOneRound(wordList, nWords, 1);
            cout << (n == 3 ? "CORRECT" : "WRONG");
                } break; case 8: {     // invalid characters in trial word
            int nWords = loadWords(wordList, 10);
            xxxmanageOneRound(wordList, nWords, 1);
                } break; case 9: {     // too many characters in trial word
            int nWords = loadWords(wordList, 10);
            xxxmanageOneRound(wordList, nWords, 1);
                } break; case 10: {    // too many characters in trial word
            int nWords = loadWords(wordList, 10);
            xxxmanageOneRound(wordList, nWords, 1);
                } break; case 11: {    // trial word not in list
            int nWords = loadWords(wordList, 10);
            xxxmanageOneRound(wordList, nWords, 1);
```

```
                } break; case 12: {    // no letters in common
            int nWords = loadWords(wordList, 10);
            xxxmanageOneRound(wordList, nWords, 1);
                } break; case 13: {    // more of a letter in mystery word
than trial word
            int nWords = loadWords(wordList, 10);
            xxxmanageOneRound(wordList, nWords, 1);
                } break; case 14: {    // fewer of a letter in mystery word
than trial word
            int nWords = loadWords(wordList, 10);
            xxxmanageOneRound(wordList, nWords, 1);
                } break; case 15: {    // all letters, but not mystery word
            int nWords = loadWords(wordList, 10);
            xxxmanageOneRound(wordList, nWords, 1);
                } break; case 16: {    // trial word 4, mystery word 5
            int nWords = loadWords(wordList, 10);
            xxxmanageOneRound(wordList, nWords, 1);
                } break; case 17: {    // trial word 6, mystery word 5
            int nWords = loadWords(wordList, 10);
            xxxmanageOneRound(wordList, nWords, 1);
                } break; case 18: {    // trial word 5, mystery word 4
            int nWords = loadWords(wordList, 10);
            xxxmanageOneRound(wordList, nWords, 0);
                } break; case 19: {    // trial word 5, mystery word 6
            int nWords = loadWords(wordList, 10);
            xxxmanageOneRound(wordList, nWords, 0);
                } break; case 20: {    // checking doesn't corrupt word in
list
            ostringstream oss;
            streambuf* oldoutbuf = cout.rdbuf(oss.rdbuf());
            int nWords = loadWords(wordList, 10);
            int n1 = xxxmanageOneRound(wordList, nWords, 1);
            int n2 = xxxmanageOneRound(wordList, nWords, 0);
            cout.rdbuf(oldoutbuf);
            string s = oss.str();
            size_t pos = s.find('\n');
            bool sameCounts = false;
            if (pos != string::npos  &&  pos > 0)
            {
                char c = s[pos-1];
                if (isascii(c)  &&  isdigit(c)  &&  count(s.begin(), s.end(),
c) == 4)
                        sameCounts = true;
            }
            cout << (n1 == 3 && n2 == 3 && sameCounts ? "CORRECT" : "WRONG");
                } break; case 21: {    // random numbers used
            ostringstream oss;
            streambuf* oldoutbuf = cout.rdbuf(oss.rdbuf());
            int nWords = loadWords(wordList, 10);
            xxxmain();
            cout.rdbuf(oldoutbuf);
            manageOneRound(wordList, nWords, 999999);
                } break;
        }
        cout << endl;
        cin.rdbuf(isb);
}
```

In case you're interested, the code

```
        istringstream iss(someText);
        cin.rdbuf(iss.rdbuf());
```

arranges things so that reading from `cin` no longer takes input from your keyboard; instead the characters are taken from *someText*.