**Run the code by checking the output below to see the output are correct.**

**Second, Write C++ in your own version(make the code different, even variable), add comments,s and run it again to make sure the output is the same.**

**Eventually, please write 500 words to describe why you chose to write it in that way and compare the previous code and your own code. describe if you think your version is better than the previous code.**

main.cpp

#include <iostream> // For Input and Output

#include <fstream> // For file input and output

#include <cassert> // For the assert statement

#include <cctype> // Allows using the tolower() function

#include <vector> // For vectors

#include <ctime> // Timer functions

using namespace std;

//--------------------------------------------------------------------------------

// Display name and program information

void displayIdentifyingInformation()

{

cout << " \n"

<< "Program: #3, Typing Tutor \n"

<< "CS 141, Fall 2021 \n"

<< endl;

}//end displayIdentifyingInformation()

//--------------------------------------------------------------------------------

// Use binary search to look up a search word in the dictionary vector, returning

// the index if found, -1 otherwise.

long binarySearch( string searchWord, // Word we are searching for

vector< string> dictionary) // Dictionary of words as a vector

{

long low, mid, high; // array indices for binary search

int searchResult = -1; // Stores index of word if search succeeded, else -1

int guessNumber = 1; // Used to display the number of guesses

// Binary search for word

low = 0;

high = dictionary.size() - 1;

while( low <= high) {

mid = (low + high) / 2;

// searchResult negative value means word is to the left, positive value means

// word is to the right, value of 0 means word was found

cout << " " << guessNumber++ << ". Comparing to: " << dictionary[ mid] << endl;

searchResult = searchWord.compare( dictionary[ mid]);

if ( searchResult == 0) {

// Word IS in dictionary, so return the index where the word was found

return mid;

}

else if (searchResult < 0) {

high = mid - 1; // word should be located prior to mid location

}

else {

low = mid + 1; // word should be located after mid location

}

} //end while( low...

// Word was not found

return -1;

}//end binarySearch()

//---------------------------------------------------------------------------

void readWordsIntoDictionary(

vector <string> & dictionary, // Dictionary where words will be stored

char fileName[]) // File name where words will be read from

{

ifstream inStream; // Declare an input stream for reading

inStream.open( fileName); // Open dictionary file

assert( inStream.fail() == false ); // Ensure file open worked

// Clear vector in case it already had some words in it

dictionary.clear();

// Keep repeating while input from the file yields a word

string newWord; // Store a single input word

while( inStream >> newWord) { // While there is another word to be read

// Add this new word to the end of the vector, growing it in the process

dictionary.push\_back( newWord);

}

// Close the dictionary file

inStream.close();

} //end readWordsIntoDictionary()

//---------------------------------------------------------------------------

// Read in set of ~100 commonly misspelled words. Each line has a common

// misspelling followed by the correct spelling. These are read into two

// separate vectors: dictionary and correctSpellings.

void readCommonMisspelledWords(

vector <string> & dictionary, // Dictionary for incorrectly spelled words

vector <string> & correctSpellings, // Dictionary for corresponding correctly spelled words

char fileName[]) // File name where words will be read from

{

ifstream inStream; // Declare an input stream for reading

inStream.open( fileName); // Open dictionary file

assert( inStream.fail() == false ); // Ensure file open worked

// Clear vectors in case they already had some words in them

dictionary.clear();

correctSpellings.clear();

// Keep repeating while input from the file yields a word

string incorrectWord, correctWord; // Store the incorrect and correct words from each input line

while( inStream >> incorrectWord >> correctWord ) { // While there is another word pair to be read

// Add the incorrect and correct new words to their respective vectors, growing them in the process

dictionary.push\_back( incorrectWord);

correctSpellings.push\_back( correctWord);

}

// Close the dictionary file

inStream.close();

} //end readCommonMisspelledWords()

//---------------------------------------------------------------------------

// Calculate and display the score for this move based on the number of words

// spelled correctly/incorrectly and the length of time taken.

int calculateAndDisplayScore(

int elapsedSeconds, // How long the move took

int timeLimit, // Expected time limit

int wordLength, // Length of words (when using large dictionary)

int howManyWords, // How many words are being displayed to be typed

int numberOfMisspelledWords) // How many words were typed incorrectly

{

int score = 0; // Value will be computed and returned at the end

// Handle bonus/penalty for time

int timePointsAdjustment = 0;

if( elapsedSeconds < timeLimit) {

// Bonus for finishing under the designated number of seconds

timePointsAdjustment = (timeLimit - elapsedSeconds) \* 2;

cout << timePointsAdjustment << " point bonus for finishing " << (timeLimit - elapsedSeconds)

<< " seconds early.";

}

if( elapsedSeconds > timeLimit) {

timePointsAdjustment = -1 \* (elapsedSeconds - timeLimit) \* 3;

cout << timePointsAdjustment << " points for finishing " << (elapsedSeconds - timeLimit)

<< " seconds late.";

}

cout << endl;

// Calculate the number of points per word.

int pointsPerWord = 3; // The default, used for commonly misspelled words

int incorrectWordDeduction = 0; // Stores deduction points for incorrectly spelled words

if( wordLength != 0) {

// We're using the full dictionary, not commonly misspelled words, so points per word

// is relative to the word length.

pointsPerWord = wordLength - 4;

}

// Handle score calculation based on number spelled correctly/incorrectly

int numberOfCorrectWords = howManyWords - numberOfMisspelledWords;

cout << " " << numberOfCorrectWords \* pointsPerWord << " points: " << numberOfCorrectWords

<< " spelled correctly x " << pointsPerWord << " points each" << endl;

// Only display the misspellings penalty if some words were missed

if( numberOfMisspelledWords > 0) {

incorrectWordDeduction = pointsPerWord \* 2;

cout << " " << numberOfMisspelledWords \* incorrectWordDeduction << " point penalty: "

<< numberOfMisspelledWords << " spelled incorrectly x "

<< incorrectWordDeduction << " points each" << endl;

}

// Calculate and display score based on number of misspellings and time, as shown above

score = (numberOfCorrectWords \* pointsPerWord) -

(numberOfMisspelledWords \* incorrectWordDeduction) + timePointsAdjustment;

cout << " " << "Score: " << score

<< endl;

return score;

} //end calculateAndDisplayScore()

//---------------------------------------------------------------------------

// Do the word test, giving a limited length of time for the user to type in

// a set of randomly selected words.

// This same function is used both for typing randomly selected words from the

// dictionary, as well as in typing randomly selected words from the list of

// commonly misspelled words.

// The second parameter gives the vector of correctly spelled words to compare

// against. In the case of using normal dictionary words, both parameters point to

// the same dictionary. In the case of using commonly misspelled words, the two are

// different.

// The last parameter is a default parameter. It is used only when using the full

// dictionary, not when using the commonly misspelled words. It determines what

// length words will be used. Longer words give higher scores.

void wordTest(

vector <string> wordsToUse, // Randomly selected words to be typed

vector <string> dictionaryForLookups, // Dictionary to use to find correctly spelled words

int &score, // Score on this run

int wordLength = 0) // Default parameter, not used w/commonly misspelled words

{

int howManyWords = 5; // How many words to be typed

int timeLimit = 15; // How many seconds the user has to type in the words

vector <string> wordsToType;

vector <string> correctlySpelledWords;

vector <string> userTypedWords;

// Select some number of randomly selected words to be typed, displaying them and storing

// them into a vector. Store the correctly spelled versions of each one in a parallel array.

// We use a while loop here rather than a for loop to facilitate the use of the continue

// statement, used when working with the full dictionary, to only choose certain length words.

int i = 0;

while( i<howManyWords) {

int wordIndex = rand() % wordsToUse.size(); // Generate an index for some random word

// Validate word length, but only when using the full dictionary and wordLength is non-zero

if( wordLength != 0 && wordsToUse.at( wordIndex).length() != wordLength) {

continue; // Try again, until we find the correct length word

}

wordsToType.push\_back( wordsToUse.at( wordIndex)); // Add this word to the list of words to be typed

cout << wordsToUse.at( wordIndex) << " "; // Display the word

correctlySpelledWords.push\_back( dictionaryForLookups.at( wordIndex)); // Store correct spelling

i++;

}

cout << endl;

// Declare a variable to hold a time, and set it to the current time

time\_t startTime = time( NULL);

int elapsedSeconds = 0;

// Store user-typed words.

cout << "Type in those words within goal of " << timeLimit << " seconds: " << endl;

string userInput;

for( int i=0; i<howManyWords; i++) {

cin >> userInput;

userTypedWords.push\_back( userInput);

}

// Validate spelling of each user-typed word, concatenating all misspelled

// words for display.

int numberOfMisspelledWords = 0;

string outputMessage = " Misspelled words: ";

for( int i=0; i<howManyWords; i++) {

if( userTypedWords.at( i).compare( correctlySpelledWords.at( i)) != 0) {

// User-typed word was not spelled correctly

numberOfMisspelledWords++;

outputMessage = outputMessage + "\n " + userTypedWords.at( i) +

" should be: " + correctlySpelledWords.at( i);

}

}

// Display misspelled words, if any

if( numberOfMisspelledWords > 0) {

cout << outputMessage << endl;

}

else {

cout << " No misspelled words!" << endl;

}

// Calculate the elapsed time as the difference between the current time and the start time

elapsedSeconds = difftime( time( NULL), startTime);

cout << " Elapsed time: " << elapsedSeconds << " seconds. ";

score = calculateAndDisplayScore( elapsedSeconds, timeLimit, wordLength, howManyWords, numberOfMisspelledWords);

} //end wordTest()

//---------------------------------------------------------------------------

// Display dictionary words within some valid range.

void displayDictionaryWords( vector <string> dictionary) // Vector to store dictionary words)

{

cout << endl

<< " Enter the start and end indices between 0 and " << dictionary.size() - 1 << ": ";

int start, end;

cin >> start >> end;

// Sanity check

if( start < 0 || end > dictionary.size() - 1) {

cout << "Start or end value out of bounds. Exiting program..." << endl;

exit( -1);

}

// Display the words

for( int i=start; i<=end; i++) {

cout << " " << i << ". " << dictionary.at( i) << endl;

}

} //end displayDictionaryWords()

//---------------------------------------------------------------------------

// Main loop that reads in words, then displays and processes menu options.

int main()

{

int menuChoice; // Store user input for menu option chosen

vector <string> dictionary; // Vector to store dictionary words

vector <string> correctSpellings; // Vector to store correct versions of common incorrectly spelled words

vector <string> incorrectSpellings; // Vector to store common incorrectly spelled words

int bestScore = 0; // Best score

int score = -1; // Current score

int wordLength = 5; // Length of words currently to display

// Read in words from the dictionary and commonly misspelled words files into vectors.

readWordsIntoDictionary( dictionary, "dictionary.txt");

readCommonMisspelledWords( incorrectSpellings, correctSpellings, "misspelledWords.txt");

// For random number generation seed the random number generator to 1, so that results

// are predictable. To make results different every time instead you would

// use srand( time( NULL)); Don't do this if you want your code to match test cases!

srand( 1);

cout << "There are " << dictionary.size() << " words in the full dictionary." << endl;

cout << "There are " << incorrectSpellings.size() << " commonly misspelled words." << endl;

// Keep looping until user selects the option to exit

while( true) {

cout << endl

<< "Select a menu option:" << endl

<< " 1. Spell commonly misspelled words (timed)" << endl

<< " 2. Type random words from full dictionary (timed)" << endl

<< " 3. Display some words from one of the lists" << endl

<< " 4. Use binary search to lookup a word in full dictionary" << endl

<< " 5. Set word length to use with full dictionary" << endl

<< " 6. Exit the program" << endl

<< "Your choice --> ";

cin >> menuChoice;

cout << endl;

// Exit if option 6 to exit was chosen

if( menuChoice == 6) {

cout << "Exiting program" << endl;

break; // Break out of enclosing infinite game-play loop

}

// Reset word length if option 5 was chosen

if( menuChoice == 5) {

cout << "Enter new wordLength: ";

cin >> wordLength;

continue; // Loop back up to retry, with new wordlength value

}

// Binary search of a word in large dictionary was chosen

if( menuChoice == 4) {

string wordToLookup;

cout << "Enter the word to lookup: ";

cin >> wordToLookup;

if( binarySearch( wordToLookup, dictionary) != -1) {

cout << wordToLookup << " was found.";

}

else {

cout << wordToLookup << " was NOT found.";

}

cout << endl;

continue; // Loop back up to display menu

} //end if( menuChoice == 4)

// Dictionary will store commonly misspelled words

if( menuChoice == 1) {

cout << "Using commonly misspelled words" << endl;

wordTest( incorrectSpellings, correctSpellings, score);

}

else if( menuChoice == 2) {

cout << "Using full dictionary" << endl;

// Do the word test, selecting random words and comparing against correctly spelled words

wordTest( dictionary, dictionary, score, wordLength);

wordLength++; // Next time use words that are 1 character longer

}

else if( menuChoice == 3) {

cout << " Displaying word list entries." << endl

<< " Which words do you want to display?" << endl

<< " A. Dictionary of all words" << endl

<< " B. Wrongly spelled common misspelled words" << endl

<< " C. Correctly spelled common misspelled words" << endl

<< " Your choice --> ";

char menuInput = ' ';

cin >> menuInput;

menuInput = toupper( menuInput);

switch( menuInput) {

case 'A': displayDictionaryWords( dictionary); break;

case 'B': displayDictionaryWords( incorrectSpellings); break;

case 'C': displayDictionaryWords( correctSpellings); break;

default: cout << "Invalid switch - case option chosen. Exiting...";

exit( -1);

break;

}

} //end if( menuChoice == 3)

// Keep track of best score

if( score > bestScore) {

bestScore = score;

}

} //end while( true)

cout << "Best score was " << bestScore << endl;

return 0;

}//end main()

**Run the code by checking the output below to see the output are correct.**

**Second, Write C++ in your own version (make the code different, even variable), add comments,s and run it again to make sure the output is the same.**

**Eventually, please write 500 words to describe why you chose to write it in that way and compare the previous code and your own code. describe if you think your version is better than the previous code.**

**Output Should be:**

At the top of this document is an extended example run on Zybooks, where the elapsed time is always 0 seconds because the input was pre-typed into the input window.  Near the bottom there is an additional example showing what it looks like if the input is provided interactively, showing the scoring and output format when time runs out.

------------------------------------------------------------------------------------

There are 263533 words in the full dictionary.

There are 114 commonly misspelled words.

Select a menu option:

   1. Spell commonly misspelled words (timed)

   2. Type random words from full dictionary (timed)

   3. Display some words from one of the lists

   4. Use binary search to lookup a word in full dictionary

   5. Set word length to use with full dictionary

   6. Exit the program

Your choice --> **4**

Enter the word to lookup: **exploration**

     1. Comparing to: mankier

     2. Comparing to: dormin

     3. Comparing to: hammerheads

     4. Comparing to: fictionizations

     5. Comparing to: entree

     6. Comparing to: exoenzymes

     7. Comparing to: fanlight

     8. Comparing to: extravaganzas

     9. Comparing to: exposture

     10. Comparing to: expellants

     11. Comparing to: explanatory

     12. Comparing to: exploring

     13. Comparing to: explodes

     14. Comparing to: exploitive

     15. Comparing to: explorative

     16. Comparing to: explorational

     17. Comparing to: exploits

     18. Comparing to: exploration

exploration was found.

Select a menu option:

   1. Spell commonly misspelled words (timed)

   2. Type random words from full dictionary (timed)

   3. Display some words from one of the lists

   4. Use binary search to lookup a word in full dictionary

   5. Set word length to use with full dictionary

   6. Exit the program

Your choice --> **4**

Enter the word to lookup: **gloopzy**

     1. Comparing to: mankier

     2. Comparing to: dormin

     3. Comparing to: hammerheads

     4. Comparing to: fictionizations

     5. Comparing to: gasolenes

     6. Comparing to: gosling

     7. Comparing to: girandolas

     8. Comparing to: gluttonised

     9. Comparing to: gliadin

     10. Comparing to: glopped

     11. Comparing to: glob

     12. Comparing to: globulet

     13. Comparing to: glomerule

     14. Comparing to: gloomier

     15. Comparing to: gloomy

     16. Comparing to: gloopiest

     17. Comparing to: gloops

     18. Comparing to: gloopy

     19. Comparing to: glop

gloopzy was NOT found.

Select a menu option:

   1. Spell commonly misspelled words (timed)

   2. Type random words from full dictionary (timed)

   3. Display some words from one of the lists

   4. Use binary search to lookup a word in full dictionary

   5. Set word length to use with full dictionary

   6. Exit the program

Your choice --> **3**

   Displaying word list entries.

   Which words do you want to display?

      A. Dictionary of all words

      B. Wrongly spelled common misspelled words

      C. Correctly spelled common misspelled words

   Your choice --> **a**

   Enter the start and end indices between 0 and 263532: **200 205**

      200. aberrants

      201. aberrate

      202. aberrated

      203. aberrates

      204. aberrating

      205. aberration

Select a menu option:

   1. Spell commonly misspelled words (timed)

   2. Type random words from full dictionary (timed)

   3. Display some words from one of the lists

   4. Use binary search to lookup a word in full dictionary

   5. Set word length to use with full dictionary

   6. Exit the program

Your choice --> **3**

   Displaying word list entries.

   Which words do you want to display?

      A. Dictionary of all words

      B. Wrongly spelled common misspelled words

      C. Correctly spelled common misspelled words

   Your choice --> **b**

   Enter the start and end indices between 0 and 113: **100 105**

      100. tommorow

      101. tommorrow

      102. tounge

      103. truely

      104. unforseen

      105. unfortunatly

Select a menu option:

   1. Spell commonly misspelled words (timed)

   2. Type random words from full dictionary (timed)

   3. Display some words from one of the lists

   4. Use binary search to lookup a word in full dictionary

   5. Set word length to use with full dictionary

   6. Exit the program

Your choice --> **3**

   Displaying word list entries.

   Which words do you want to display?

      A. Dictionary of all words

      B. Wrongly spelled common misspelled words

      C. Correctly spelled common misspelled words

   Your choice --> **c**

   Enter the start and end indices between 0 and 113: **100 105**

      100. tomorrow

      101. tomorrow

      102. tongue

      103. truly

      104. unforeseen

      105. unfortunately

Select a menu option:

   1. Spell commonly misspelled words (timed)

   2. Type random words from full dictionary (timed)

   3. Display some words from one of the lists

   4. Use binary search to lookup a word in full dictionary

   5. Set word length to use with full dictionary

   6. Exit the program

Your choice --> **1**

Using commonly misspelled words

tendancy existance accomodation posession dissapear

Type in those words within goal of 15 seconds:

**tendency existence accommodation posession disappear**

   Misspelled words:

       posession should be: possession

   Elapsed time: 0 seconds. 30 point bonus for finishing 15 seconds early.

   12 points: 4 spelled correctly x 3 points each

   6 point penalty: 1 spelled incorrectly x 6 points each

   Score: 36

Select a menu option:

   1. Spell commonly misspelled words (timed)

   2. Type random words from full dictionary (timed)

   3. Display some words from one of the lists

   4. Use binary search to lookup a word in full dictionary

   5. Set word length to use with full dictionary

   6. Exit the program

Your choice --> **1**

Using commonly misspelled words

truely assasination dissapoint comming apparantly

Type in those words within goal of 15 seconds:

**truly assassination disappoint coming apparently**

   No misspelled words!

   Elapsed time: 0 seconds. 30 point bonus for finishing 15 seconds early.

   15 points: 5 spelled correctly x 3 points each

   Score: 45

Select a menu option:

   1. Spell commonly misspelled words (timed)

   2. Type random words from full dictionary (timed)

   3. Display some words from one of the lists

   4. Use binary search to lookup a word in full dictionary

   5. Set word length to use with full dictionary

   6. Exit the program

Your choice --> **1**

Using commonly misspelled words

collegue realy collegue ocassion Phillipines

Type in those words within goal of 15 seconds:

**colleague really colleague occasion Philipines**

   Misspelled words:

       Philipines should be: Philippines

   Elapsed time: 0 seconds. 30 point bonus for finishing 15 seconds early.

   12 points: 4 spelled correctly x 3 points each

   6 point penalty: 1 spelled incorrectly x 6 points each

   Score: 36

Select a menu option:

   1. Spell commonly misspelled words (timed)

   2. Type random words from full dictionary (timed)

   3. Display some words from one of the lists

   4. Use binary search to lookup a word in full dictionary

   5. Set word length to use with full dictionary

   6. Exit the program

Your choice --> **2**

Using full dictionary

devil yipee atria layup chark

Type in those words within goal of 15 seconds:

**devil yipee atria layup chark**

   No misspelled words!

   Elapsed time: 0 seconds. 30 point bonus for finishing 15 seconds early.

   5 points: 5 spelled correctly x 1 points each

   Score: 35

Select a menu option:

   1. Spell commonly misspelled words (timed)

   2. Type random words from full dictionary (timed)

   3. Display some words from one of the lists

   4. Use binary search to lookup a word in full dictionary

   5. Set word length to use with full dictionary

   6. Exit the program

Your choice --> **2**

Using full dictionary

thinly marron biffin sysops perses

Type in those words within goal of 15 seconds:

**thinly marron bifin sysops perses**

   Misspelled words:

       bifin should be: biffin

   Elapsed time: 0 seconds. 30 point bonus for finishing 15 seconds early.

   8 points: 4 spelled correctly x 2 points each

   4 point penalty: 1 spelled incorrectly x 4 points each

   Score: 34

Select a menu option:

   1. Spell commonly misspelled words (timed)

   2. Type random words from full dictionary (timed)

   3. Display some words from one of the lists

   4. Use binary search to lookup a word in full dictionary

   5. Set word length to use with full dictionary

   6. Exit the program

Your choice --> **2**

Using full dictionary

frisker zamangs magnons syntagm cruelty

Type in those words within goal of 15 seconds:

**frisker zamangs magnons syntagm cruelty**

   No misspelled words!

   Elapsed time: 0 seconds. 30 point bonus for finishing 15 seconds early.

   15 points: 5 spelled correctly x 3 points each

   Score: 45

Select a menu option:

   1. Spell commonly misspelled words (timed)

   2. Type random words from full dictionary (timed)

   3. Display some words from one of the lists

   4. Use binary search to lookup a word in full dictionary

   5. Set word length to use with full dictionary

   6. Exit the program

Your choice --> **1**

Using commonly misspelled words

unforseen enviroment futher resistence catagory

Type in those words within goal of 15 seconds:

**unforseen environment further resistance category**

   Misspelled words:

       unforseen should be: unforeseen

   Elapsed time: 0 seconds. 30 point bonus for finishing 15 seconds early.

   12 points: 4 spelled correctly x 3 points each

   6 point penalty: 1 spelled incorrectly x 6 points each

   Score: 36

Select a menu option:

   1. Spell commonly misspelled words (timed)

   2. Type random words from full dictionary (timed)

   3. Display some words from one of the lists

   4. Use binary search to lookup a word in full dictionary

   5. Set word length to use with full dictionary

   6. Exit the program

Your choice --> **5**

Enter new wordLength: **5**

Select a menu option:

   1. Spell commonly misspelled words (timed)

   2. Type random words from full dictionary (timed)

   3. Display some words from one of the lists

   4. Use binary search to lookup a word in full dictionary

   5. Set word length to use with full dictionary

   6. Exit the program

Your choice --> **2**

Using full dictionary

music stong avoid rebut clots

Type in those words within goal of 15 seconds:

**music strong avoid rebut clots**

   Misspelled words:

       strong should be: stong

   Elapsed time: 0 seconds. 30 point bonus for finishing 15 seconds early.

   4 points: 4 spelled correctly x 1 points each

   2 point penalty: 1 spelled incorrectly x 2 points each

   Score: 32

Select a menu option:

   1. Spell commonly misspelled words (timed)

   2. Type random words from full dictionary (timed)

   3. Display some words from one of the lists

   4. Use binary search to lookup a word in full dictionary

   5. Set word length to use with full dictionary

   6. Exit the program

Your choice --> **6**

Exiting program

Best score was 45

------------------------------------------------------------------------------------

Now for an example showing what the output looks like (though not in Zybooks) when time runs out:

There are 263533 words in the full dictionary.

There are 114 commonly misspelled words.

Select a menu option:

   1. Spell commonly misspelled words (timed)

   2. Type random words from full dictionary (timed)

   3. Display some words from one of the lists

   4. Use binary search to lookup a word in full dictionary

   5. Set word length to use with full dictionary

   6. Exit the program

Your choice --> **1**

Using commonly misspelled words

**happend apparantly neccessary parralel definately**

Type in those words within goal of 15 seconds:

happened apparently necessary parrrallel definitely

   Misspelled words:

       parrrallel should be: parallel

   Elapsed time: 27 seconds. -36 points for finishing 12 seconds late.

   12 points: 4 spelled correctly x 3 points each

   6 point penalty: 1 spelled incorrectly x 6 points each

   Score: -30

Select a menu option:

   1. Spell commonly misspelled words (timed)

   2. Type random words from full dictionary (timed)

   3. Display some words from one of the lists

   4. Use binary search to lookup a word in full dictionary

   5. Set word length to use with full dictionary

   6. Exit the pr**ogram**

**Your choice -->**2

Using full dictionary

angst edema unjam joint equal

Type in those words within goal of 15 seconds:

**angst edema unjamm joint equal**

   Misspelled words:

       unjamm should be: unjam

   Elapsed time: 29 seconds. -42 points for finishing 14 seconds late.

   4 points: 4 spelled correctly x 1 points each

   2 point penalty: 1 spelled incorrectly x 2 points each

   Score: -40

Select a menu option:

   1. Spell commonly misspelled words (timed)

   2. Type random words from full dictionary (timed)

   3. Display some words from one of the lists

   4. Use binary search to lookup a word in full dictionary

   5. Set word length to use with full dictionary

   6. Exit the program

Your choice --> **6**

Exiting program

Best score was 0