 **Weight: 7%**

Data Structures

Project 2 – Jumble Game

Contents

[Objective 1](#_Toc286820763)

[Resources 1](#_Toc286820764)

[Gameplay 1](#_Toc286820765)

[Requirements 1](#_Toc286820766)

[Milestones 2](#_Toc286820767)

[Grading Breakdown 2](#_Toc286820768)

[Standard Deductions 2](#_Toc286820769)

[Submission 3](#_Toc286820770)

# Objective

You will be responsible for creating a 6 letter word jumble game. The player will be presented with 6 letters and his objective is to think of as many words as he can that are made up of those 6 letters in a given amount of time.

# Resources

Contained in the *Class Resources* folder on the server you will find a file named engldict.txt. This file is a basic list of words that will be legal answers for your game.

# Gameplay

Each round of the game, you should find a random 6 letter word in the hash table, store and scramble the letters that form this word, and display them to the screen. The player will have 60 seconds to make as many words as possible using those 6 letters. Each letter position can be used only once (in order to use the same letter twice in a word, that letter must appear twice in the scramble). If the player’s guesses are real words (in the dictionary) and made up of the letters in the scramble, the word will be stored and points will be awarded for a correct guess (The score for a word is the factorial of its length-1).

# Requirements

* You are required to load the dictionary and store the legal words for this game in a hash table. This means you will have to research and write your own hash function.
  + The hash table must use at least 500 buckets
  + No more than 5% of your buckets should be empty
  + Items should be distributed as evenly as possible among the various buckets
    - You should not store any words shorter than 3 letters or longer than 6 letters as those words will not score points for the player
    - You should not store any word with an apostrophe (‘) in it
* You are required to display the time remaining in a round on screen. This must be a real time countdown. This means your player input should also be gathered in real time.
* You are required to find a random 6 letter word from the dictionary each round of the game and rearrange those letters to form the jumble.
* You are required to display the score to the screen, updating it in real time as the player guesses words.
* You are required to store the player’s correct guesses in a Binary Search Tree and display them to the screen, updating it in real time as the player correctly guesses words.
* You are required to show the random 6 letter word to the user, if they do not guess it by the end of the round.
* You are required to have a replay option.
* You are required to keep a table of the top 5 high scores. This should be stored in a file. The format for the data in this file and the name of the file is up to you.
  + The high score table should be displayed at the end of a game.
  + Store both the name and the score.
  + The high scores should be sorted and displayed in descending order (highest at top, lowest at bottom).

# Milestones

Day 6 – write hash function that meets above requirements. Load dictionary from file. Find random word in dictionary.

Day 7 – real-time keyboard input, timer on screen, store users correct guesses, all gameplay elements implemented.

# Grading Breakdown

Requirement Point value

|  |  |
| --- | --- |
| Hash function meets requirements / dictionary loaded and hashed | 30 |
| Real-time input | 10 |
| Random word found in dictionary | 20 |
| Display (timer, scrambled letters, score, correct guesses) | 25 |
| Replay / high score table | 15 |

# Standard Deductions

|  |  |
| --- | --- |
| Compiler errors | 100 |
| Crash | 100 |
| Memory Leak | 50 each |
| Header protection | 20 each |
| Compiler warnings | 10 each |

# Submission

Due Date : 23:59:59 PM on Day 7

To submit the Project 2 assignment:

1. Clean and build the project in Visual Studio. ensure there are no errors or warnings.

2. Run the project in debug mode with leak detection turned on to ensure that all behavior is correct and there are no memory leaks.

3. Close visual studio.

4. Navigate to the Project 2 folder (The folder that contains your Project 2 Visual Studio solution and project).

5. Delete any of the following files or folders you find (you may not have all of these, some of them are generated only in special cases and some of them are only generated by previous versions of the compiler)

**\*\*Make sure you have your Windows environment set to show you hidden files, or you may miss some of these\*\***

     a) Delete the .ncb (intellisense database) file (if you have one or more, delete them all)

     b) Delete the .suo (solution user options) file (if you have one or more, delete them all)

     c) Delete the .user (user settings options) file (if you have one or more, delete them all)

     d) Delete the Debug folder (you SHOULD have one of these)

     e) Delete the Release folder (you will only have one of these if you built in release mode)

     f) Delete the ipch folder (if you have one)

     g) Delete the .sdf (sql server database) file (if you have one)

     h) Delete the .filters file (if you have one or more, delete them all)

6. On your desktop, create a new folder with your name in the following format: "Last Name, First Name" - nothing else.

     Your Last Name - a comma - a single space - your First Name. Appropriate capitalization for proper names should be used.

     Examples : "Pollack, Joey"; "De La Paz, Christhian"; "Tjarks, Matthew".

7. Copy your Project 2 folder into the folder that you just created.

8. Compress the folder by right clicking on the folder with your mouse and selecting 'Send To->Compressed (Zipped) Folder'.

9. Submit the Compressed Folder.