CS 135

Computer Science I

Fall 2022

Assignment 6

A picture containing calendar

Description automatically generated

School of Computer Science

Howard Hughes College of Engineering

University of Nevada, Las Vegas

# **Assignment summary**

In this assignment, you will create a crossword game that reads the levels from data files. The user will select a level to play, and the application will load the level if the data file is available. User’s input is to be validated and the game needs to tell the user if the level’s data file is missing. The user can have 5 incorrect guesses before losing the game. All correct guesses are going to be displayed on the board and the application will keep track of all user guesses. Upon winning or losing a level, the user will be asked if they want to play again.

1. **Sample Run**

Figures below show a sample application run. Note that all figures below are screenshots of the same run.

Calendar

Description automatically generated

Calendar

Description automatically generated

Figure 2:Sample Run - Screenshot 2

Figure :Sample Run - Screenshot 1

Calendar

Description automatically generated

Figure 3: Sample Run - Screenshot 3

1. **Selecting and Loading Levels**

Levels are read from text files that are included with this assignment’s files. User’s input is to be validated whenever the user selects a level. The application will also check if the selected level file is available or not. Different messages will be displayed accordingly. See screenshots below for reference:

Text

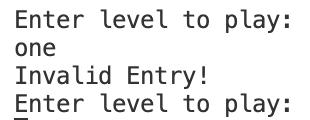
Description automatically generated

Figure 4: Invalid Level

Figure 5: Sample Run - Screenshot 3

1. **Displaying Levels and Playing the Game**

The application will read the level’s data as characters into a 2-D vector and will display it on the screen. Letters will be displayed as underscores until guessed by the user, i.e., when the user guesses a letter that should be on the board, the letter will be displayed in the correct locations. Guessing a correct letter will not affect the number of remaining guesses, however, that number will still be displayed on the screen. Locations that do not contain letters will be displayed as # character.

Text, letter

Description automatically generated

Figure 6: Guessing a Correct Letter

Upon guessing a number that’s not on the board the application will tell the user that the letter is not on the board and will decrease the number of available guesses. The board will be displayed as well.

Calendar

Description automatically generated with medium confidence

Figure 7: Guessing an Incorrect Letter

The application will also keep track of all guessed characters. When the user re-enters a character that they have already guessed, the application will display a message to the user and display the board again. The number of remaining guesses will not be affected but it will be displayed with the board.

A picture containing text

Description automatically generated

Figure 8: Guessing an Already Guessed Letter

The application will not care about the letter case, i.e. The user can enter either uppercase or lowercase letters and this should not affect how the application functions.

Calendar

Description automatically generated

Figure 9: Entering an Uppercase Letter

Calendar

Description automatically generated

Figure 10: Entering a Lowercase Letter

1. **Ending the Game**

The game ends when the user guesses all letters on the board or when they run out of guesses. A different message will be displayed depending on the scenario.

Calendar

Description automatically generated with medium confidence

Figure 11: Winning the Game

A picture containing text

Description automatically generated

Figure 12: Losing the Game

The user will have the option to play again or quit the game. The user’s response can be an uppercase or lowercase. Any letter other than Y, y, N, or n is considered an invalid input. If the user selects to play again, the game will restart. The game’s logo will not be displayed again but everything else will run normally.

If the user selects not to play another round, the application will terminate, and a message will be displayed. See screenshot below for reference.

Text

Description automatically generated

Figure 13: Ending the Game

1. **Error Handling**

Every user input will be validated to make sure that the user enters correct data. No need to worry about the user entering numbers or special characters for guesses though. However, you need to make sure that the user enters a number for the level number and validate the user’s input for the application’s termination.

Notes: when the user enters any number below 1 for the level, the message should be “**Invalid Entry!**”. If the number is above zero but the level files are not found, then the message needs to be “**Level files could not be found!**”.

The user could enter a word instead of a letter when guessing the letters. The application will only take the first letter and discard the rest.

1. **Submission**

Submit a C++ file named main.cpp to CodeGrade via Canvas. No need to worry about submitting the levels as they’re already uploaded to CodeGrade.