UNIVERSITI TEKNOLOGI PETRONAS

EEB1043/EDB1023/EFB1023 STRUCTURED PROGRAMMING AND INTERFACING JANUARY 2024

TEXT TWIST GAME PROJECT

Contents

1. OBJECTIVES

The topic of this project is *"Text Twist"*. You are asked to develop a word puzzle game known as Text Twist using C language. Students need to work in pairs and **must use the *"gfx"* graphic library** to create the graphics of the game. Text Twist game is described is in **Section 2.**

2. TEXT TWIST BACKGROUND

Text Twist is a word puzzle game where the player is given a string of letters. Players can form as many English words as possible from these letters to make words of three letters or more. To advance to the next round the player must find the 'bingo' word. The bingo word is a word that uses all the letters (there may be more than one).

Figure 1 shows an example of a Text Twist puzzle.

In this project, you need to create a graphical user interface (GUI) based Text Twist game. In other words, it must contain the gameplay as well as the GUI. The GUI should consist of the following items:

- letters boxes for the guessed letter
- clear/delete button
- enter button
- the letters that the user guessed should appear on the GUI.
- the user must use a mouse to choose the letter, delete the letter or enter the letter during the game play.
- An option to exit the game and return to the main page should be available while the player is playing. In Figure 1, MENU is the button to return to the main page.
- The twist and last buttons are OPTIONAL.

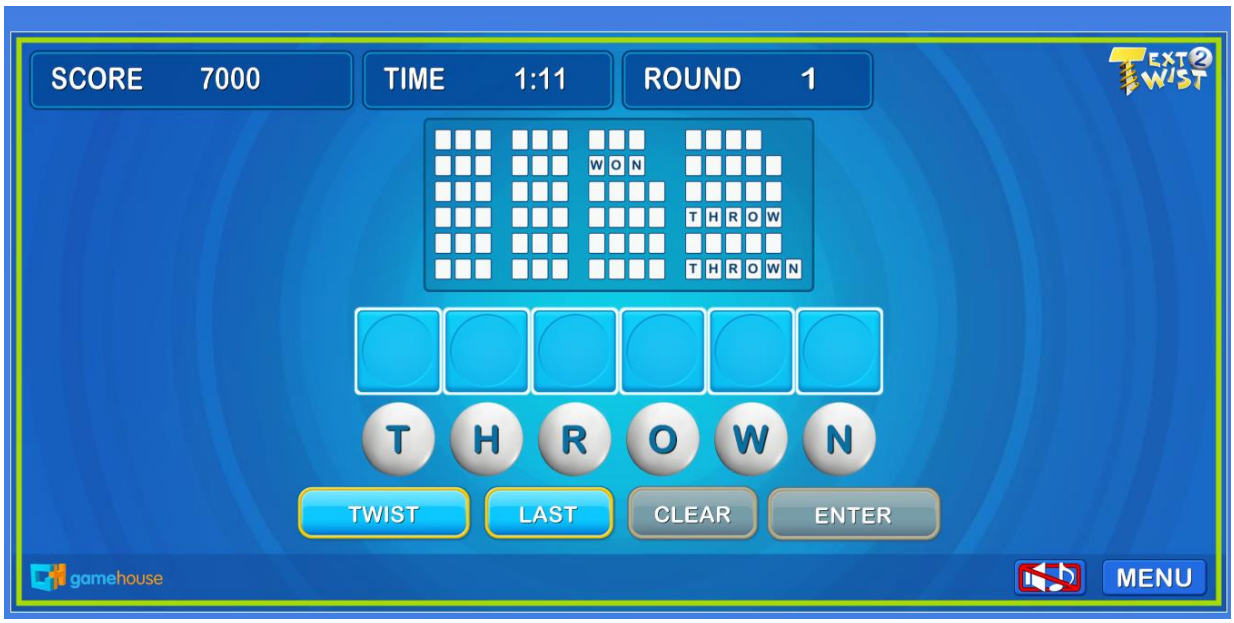

Figure 1: GUI for Text Twist game and example

What are the rules of this game?

The rules are:

- All words must be at least 3 letters long.
- You do not have to use all the letters. However, the longer the words, the more points you score.
- In each game, you can find at least one bingo. A bingo is a word that uses all the given letters. You may find more than one bingo.
- You cannot make the same word more than once.

Examples of the game

In Figure 1, the letters "T, H, R, O, W, N" is given.

- You can make the word "WON" because it is at least 3 letters long.
- We could not make the word "THR" because the word does not exist.
- The word "THROWN" is the Bingo word! Because all letters are used

Play Text Twist Game

You can play the game from this website:
https://zone.msn.com/gameplayer/gameplayerHTML.aspx?game=texttwist2

3. PROJECT DELIVERABLES

Students must start the game design with Level 1. You can add the Graphics to Level 1 to achieve Level 2 only after the completion of Level 1. Similarly, implement Level 3 after the completion of Level 2. The marks will be awarded based on how many levels have been achieved and evaluation criteria described in this document. All three levels must be implemented in one C program (not separately).

Level 1: Text-based Text Twist

The text-based user interface that will do the following:

1. In the Start Screen, the user will be given three choices:
   a. New game – if the user chooses this, a new Text Twist game will start.
   b. Get Help on how to play the game – if the user chooses this, a new screen will appear with instructions on how to play the game. Pressing Enter will return the user to the Start Screen
   c. Exit – the program exits.

```
Welcome to Text Twist! Choose:
1: New Game
2: Help
3: Exit

If user chooses option 2, display the instructions on how
to play game on new window and pressing enter will navigate
user to the main page (start screen). If option 3 is chosen,
exit the game. If user chooses option 1, game will begin.
```

2. When a new game is started (option 1 is chosen), a letter puzzle is given such as thrown in Figure 1. The user is asked to input his guess, and the guess will be compared to the Text Twist dictionary of valid words e.g. :

```
Here is your puzzle:

THROWN
Guess a word or END to quit: TOW
TOW: Valid word
```

3. The user inputs his word guess and presses enter. Your program needs to check for the following invalid guesses:
   a. Consists of numbers/symbols
   b. Already found – word has been captured or found
   c. Invalid word – word is not in the dictionary
   d. Too shorts – the word is less than 3 letters

   As for the valid guess, program needs to check for *bingo*
   a. Bingo – the word uses all the letters

4. If a guess is invalid, the program displays an informative message based on the invalid condition and asks the user to enter another guess eg.:

```
Here is your puzzle:

THROWN


Guess a word or END to quit: TOW
TOW: Valid word

THROWN
```

```
You made the following words so far:
TOW
Guess a word or END to quit: TOW
You already made that word.

THROWN

You made the following words so far:
TOW
Guess a word or END to quit: TOWN
TOWN: Valid word

THROWN

You made the following words so far:
TOW TOWN
Guess a word or END to quit: WON
WON: Valid word

THROWN

You made the following words so far:
TOW TOWN WON
Guess a word or END to quit: ON
ON: Too short

THROWN

You made the following words so far:
TOW TOWN WON
Guess a word or END to quit: ON5
ON5: Invalid symbol/number 5

THROWN

You made the following words so far:
TOW TOWN WON
Guess a word or END to quit: THW
THW: Invalid word

THROWN

You made the following words so far:
TOW TOWN WON
Guess a word or END to quit: THROWN
THROWN: Bingo


THROWN

You made the following words so far:
TOW TOWN WON THROWN
```

```
Guess a word or END to quit: END
```

5. If a guess is valid, it will display the output (Valid word) as shown in the example above and display all the correct guesses so far.
6. The program keeps asking for a guess until the user quits the game or finish the game. Both options will return the user to the start screen.
7. At the end of the game, the correct word is shown, and the user returns to the start screen.

Level 2: Basic GUI-based Text Twist

You are required to write a C program that will have the **same functionality as Level 1** but with a GUI interface similar to Figure 1. This includes the following features:

1. User input is from mouse clicks of the GUI letter boxes ONLY. Keyboard inputs are not allowed.
2. The guesses/words entered are displayed on the screen.
3. Clear/Delete, Enter and Exit options must be available in the gameplay. Exit button will navigate user to the start screen.
4. The feedback on invalid guess (such as too short, invalid word etc.) and bingo are shown during the game play.
5. Buttons and mouse clicks are used for game options: New Game, Help and Exit.

Level 3: Advanced GUI-based Text Twist

You are required to write a C program that will have the same functionality as Level 2 plus the following extra components:

1. For each game, different set game of letters is included in the game and the gameplay option is randomized. This is different from Level 1 in which only 1 game set is used.
2. The dictionary of valid words is taken from a text file and not hard-coded in the program.
3. The scores of the game are recorded and displayed. The player's updated score should be displayed regularly and when the game ends, the total score is displayed.
4. Any other useful feature that is found to be acceptable such as rotate option, sound feature, timer to end the game or additional navigation.

Here are the score points:

- **Each letter is worth 2 points each.**

4. REQUIREMENTS

You can design the appearance of your game however you choose according to your creativity. Here are some requirements for the game:

a) You are required to write multiple user-defined functions for your game using C language. Your `main()` function should have a loop that plays the game. Inside this loop, multiple functions you create in your design should be called in the appropriate sequence.

b) Each function should be **meaningful** (i.e., not a minor task).

c) The code can run without syntax error.

d) Descriptions of each function are available before the start of each function. Comments must be included in the code where necessary.

e) Command for compiling your code with the details of file names must be stated in the header of your code.

f) Codes must be neat and easy to read. Use good programming practices and formatting.

g) You must use the `GameSet.txt` and `wordlist_Game.txt` files provided.

h) You must use the `gfx graphics library` for this project and other graphics libraries are prohibited.

i) Game must implement Level 1 and this level cannot be skipped. Level 2 or (and) Level 3 will not be considered if Level 1 is missing. Student must also do Level 2 before implementing Level 3. Level 1 is the engine/gameplay of your game.

k) Your graphics window and texts for the game choice must be large enough to play.

Examples of possible functions

Here are possible functions you may implement but they are not the exhaustive list of functions for the game:

1. Choose a random game from the given GameSets (puzzle)
   - When the function is called, one of the game sets will be randomly chosen and the possible words available in the dictionary of valid word list for the game set will be chosen.
2. Ask for Guess
   - This function asks for the user to enter the guessed word.
3. Check validity of guess
   - This function checks if the guess entered is a valid or invalid guess.
4. Give feedback on guess
   - This function generates the feedback on the guess i.e. if the word is correct and if it is not correct, give the suitable messages.
5. Main menu
   - This function generates the main menu graphics.
6. Play Game
   - This function is the main function for game play.
7. Help
   - This function is the function to provide instructions on how to play game.
8. Clear Guess
   - This function is to clear the guesses.

You can have other user-defined functions apart from the ones mentioned above, which are the basic requirements. Efficient use of user-defined functions will give your project a better grade.

5. RESOURCES

You will be provided with the following 2 types of files for testing your game. Here are the two types of files:

- `GameSets.txt` file - A list of possible Text Twist game sets or puzzles. In this project, 3 game sets or puzzles are provided. For Level 1, you may choose one of the Text Twist game sets. For Level 3, you need to use all the game sets and the selection of the game set must be done randomly.
- `wordlist_Game1.txt, wordlist_Game2.txt and wordlist_Game3.txt.` These are the files that contain a list of valid words that can be formed from the 3 Text Twist game sets in the `GameSets.txt`. Hint: You may combine all the wordlists into 1 file for Level 3 implementation.

6. PROJECT DEMONSTRATION

Each student must take part in the project demonstration. Although this project is a group project, each student will be evaluated individually.

- In Week 9, students will need to demonstrate Level 1 achievement.
- Final demonstration is due on Week 12. Schedule will be provided.
- During demonstration, each student must present and demo the project. You can discuss and plan with your partner on task distribution during demo. For example, you can start the presentation by playing the game and explain the features and achievement of your game. You may also highlight the implementation and design of your game.
- Please adhere to the time limit of the demonstration as specified in the schedule.
- Attend the demo early and set-up your computer prior to demonstration to avoid any delay.
- Students should demonstrate their involvement and contribution in the project development and show that they know the project in and out.
- Examiner will ask questions to each student to evaluate individual contributions.
- Examiner may also ask you to explain the code.

7. SUBMISSION

Each group must submit the following deliverables by the **deadline**:

1. **Project code** is written in C and the code must use gfx library. Only 1 member from each group to submit the code. Week 12, ULearn.
2. **Video** of the game with the duration between 2 to 3 minutes. Week 12, Ulearn.
3. **Level 1** of the game must be demonstrated in week 9.
4. **Project Demonstration (Final implementation)** will be conducted physically in week 12 in SPI Lab**.**

The submission of your codes and video must be done via ULearn under the folder -> "Submission of Project File" folder for respective group.

The code must be submitted on ULearn by week 12 with the following format:

- Save your project filename as GroupName.c. For example: **"GroupPower.c"**
- Submit the project code together with other files you use to run the code such as gfx.c etc.
- The maximum files you can upload is 6 (lesser is preferable). You can submit multiple files such as main file, or your own header file. Exclude the GameSet.txt, wordlist.txt, gfx files unless there are changes in these files.
- Use the following header in your main code. Follow the format below:

```
/*****************************************************************
  Group Name              :    GroupPower.c
  1st Member Name         :    ???????????????? <--- put your name here
  1st Matric Number       :    123456789
  2nd  Member Name        :    ???????????????? <--- put your name here
  2nd Matric  Number      :    123456121
  Semester                :    Jan 2024
```

-Specify command to compile the project (modify and include additional files if exist):

gcc filename.c gfx.c -o fileoutput.o -lX11 -lm
**********/

8. EVALUATION CRITERIA

The project is worth 40% of the course mark.

Your project will be assessed based on the following components:

1. *Deliverable of your project (in this case the Game you created and the completion of your level).*
2. *Individual contribution to the project*. Individual presentation and your answers during demonstration are used to evaluate individual components. Students who have not been actively involved in the program development usually will have problem to explain the code and game in detail.
3. *C code*. We will look at your code quality, the overall design, and the performance of your program.
   - Is the implementation clear? (i.e., concise, and simple, compared to unnecessarily complicated).
   - Is there good documentation within the code? (i.e., consistent documentation of functions and comments added where they are required).
   - Is the code robust and can handle errors or wrong input?
   - Does the code run correctly?
4. *Video of the game*. Make the video brief, visually attractive, concise, and informative. Duration 2-3 minutes.

9. PENALTY

There will be a penalty for late or no submission of code and absenteeism during the demo. Students who were absent from project demonstration will be awarded "0" mark.

WARNING: Do not share your code with other groups. Plagiarism will result in a loss of marks for all students involved and anything suspicious will be investigated.