Coding Assignment 5

CSE 1320

The format and content of the output are not suggestions – they are the specification given to you to follow so please follow them. Points will be lost for not following the specification. This includes using the specified functions. This is essential to the grading process. If the assignment says to use `strchr()`, then please use `strchr()` and not some other way of achieving the same result. The rubric is very specific and your code will be graded very specifically. This assignment is written the way it is to exercise the concepts we have been learning in class.

Please watch the videos showing the game being played and examine the sample output provided in this document. Understanding how to play the game before studying this specification will be helpful. After watching the game being played, read this specification and then watch the video called "Coding Assignment 5".

Be sure to rename the `GameLib-Temp.c` file to `GameLib.c`. You need to include `GameLib.h` in `GameLib.c` and `Code5_xxxxxxxxxx.c` files.

You will create 4 files and submit them as one zip file – `Code5_xxxxxxxxxx.zip`

```
makefile
Code5_xxxxxxxxxx.c
GameLib.c
PhraseBank.txt
```

Step 1 – Create your own `PhraseBank.txt` file.

I have provided an example `PhraseBank.txt` file and you will need to create your own. You must use a least 5 phrases that are 80 characters or less each and they do not need to be song titles like the example. Please be professional and keep your phrases PG. You will submit your own version of the file but the name must be `PhraseBank.txt`. Your program will be tested with your file and with a different file of my choosing. Do not hardcode anything in your program using values based on the phrases in the file. We are NOT reading this file or using UNIX redirect (<) with it. This file will be added to your program during compile time using a `#include`. More info on that in the description of the process the `StartGame()` function description.

Part 2 – makefile

Copy your `makefile` from the previous assignment and change it to use your new `Code5.c` file. Change the library module/second SRC file from `DrawTool.c` to `GameLib.c`.

Step 3 – `main()`

Create three character arrays of size `MAX_INPUT`. `MAX_INPUT` should be `define`d to be `81` in `GameLib.h`. Be sure to initialize all 3 arrays to NULL when you declare them. I will refer to them as `Phrase`, `DashedPhrase` and `UpperPhrase` in this document but you are not required to use those names.

Create a variable to keep track of the number of strikes.

Call function `StartGame()`. You are passing the empty `Phrase` and the function fills it in with the player's chosen phrase. See function specification for more info.

Call function `DashIt()`. You are passing the `Phrase` that `StartGame()` filled in and you are passing the empty array `DashedPhrase`. Function `DashIt()` will use `Phrase` to create `DashedPhrase`. See function specification for more info.

Uppercase each character of `Phrase` and store the result in `UpperPhrase` (this will require a for loop since `toupper()` only accepts a single character).

do

    If `GuessALetter()` returns false, then increment the number of strikes.

while the player has not won (`strchr()` can't find a dash in `DashPhrase`) and while the number of strikes is less than `STRIKES`

`STRIKES` should be defined in `GameLib.h` and set to 3. One of the GTA tests will be to change `STRIKES` to another value and confirm that your program still plays properly.

After the do-while completes, the player has either won or struck out.

If the number of strikes is less than `STRIKES`

    Print the winning message as shown in the output

Else

    Print the message as shown in the output for losing the game

Step 4 – `StartGame()` in `GameLib.c`

Return type : `void`

Parameters : `char` array (here called `ChosenPhrase` but you can use whatever name you want)

The first line of this function needs to be this line **EXACTLY**

```
#include "PhraseBank.txt"
```

This will cause the preprocessor to take your declaration/initialization of `PhraseBank` from `PhraseBank.txt` and insert at this point in your code. Using this technique will allow for your program to be compiled and run with different versions of the phrase bank without changing your program.

Print the message to the screen as shown in the sample output.

While `PhraseBank[i]` is not `""`

    Call `CheckPhrase()` and pass it `PhraseBank[i]`

    Call `DashtIt()` and pass `PhraseBank[i]` and an empty array created in this function called `DashPhrase`. This function will take the phrase and dash the letters.

    Print a line of the menu as shown in the sample output

    Increment i

This a while loop like we discussed in lecture that can read through a ragged 2D array (`PhraseBank` from your `PhraseBank.txt` is a ragged 2D array).

Prompt for choice and store the choice in a variable

Verify that the user entered menu choice is valid (within the range of your menu – do not hardcode this check -the number of elements in the menu can change when different versions of the text file are included).

Copy the chosen phrase into `ChosenPhrase` so that when this function finishes and control runs to `main()`, `main()` will have the chosen phrase.

Step 5 – `CheckPhrase()` in `GameLib.c`

Return type : `void`

Parameters : `char` pointer (here called `Phrase` but you can use whatever name you want)

This function checks `Phrase` to see if it contains a dash. If the string contains a dash (use `strchr()` to find out), then print the message shown in the sample output. If you find a dash, then exit the program (using `exit`). This is the only allowable use of `exit()` in the program. Use pointer arithmetic to determine the position of the dash in the string for the output.

Step 6 – `GuessALetter()` in your `GameLib.c` file

This function asks for a letter, checks if it is the chosen phrase and replaces dashes with the appropriate letters.

`GuessALetter()` takes three parameters, a character array named `Phrase` and a character array named `DashedPhrase` and a character array named `UpperPhrase`. It has a return value of type `int`.

Create 4 variables

- a character variable named `Guess`
- a character pointer variable named `FindGuess` that is initialized to `NULL`.
- a character array named `UpperPhraseCopy` of size `MAX_INPUT` (`MAX_INPUT` is defined as 81 in `GameLib.h`)
- an `int` variable named `FoundALetter` that is initialize to 0

Use `strcpy()` to copy `UpperPhrase` into `UpperPhraseCopy`.

Print `DashedPhrase`

Print "`Guess a letter : `"

Put their guess into the variable `Guess` and uppercase `Guess`. (use `toupper()`)

Use `strchr()` to find `Guess` in `UpperPhraseCopy` and store the pointer in `FindGuess`.

while `FindGuess` is not `NULL`

- Set `FoundALetter` to 1
- Use pointer arithmetic to find the distance between `FindGuess` and `UpperPhraseCopy`. Use that distance to set the element in `DashedPhrase` to that same element from `Phrase`.
- Set the element at the same location/distance in `UpperPhraseCopy` to dash. You can do this using `[]` or by dereferencing `FindGuess`. Doing this prevents an infinite loop.
- Use `strchr()` to find `Guess` in `UpperPhraseCopy` and store the pointer in `FindGuess`.

Return the value of `FoundALetter`

NOTE : Guessing the same correct letter twice has no effect (no strike) and does not need to be detected or noticed. Guessing the same incorrect letter again will count as another strike.

Step 7 – `DashIt()` in your `GameLib.c` file

The function takes in a phrase and uses `strpbrk()` to create a dashed version of it.

Return type : `void`

Parameters : `char` pointer (here called `Phrase` but you can use whatever name you want) and an empty array (here called `DashPhrase` but you can use whatever name you want)

Copy `Phrase` into `DashPhrase` using a for loop one character at a time and uppercase it character by character at the same time. Be sure to get NULL terminator on the end of DashPhrase.

Use `strpbrk()` to find all alpha characters in `DashPhrase` and replace them with a dash (hint : `while` loop). The delimiters for `strpbrk()` should be ONLY uppercase A-Z.

Part 8 - Testing

Run your `Code5.e` and confirm that your output matches the output in the assignment. Confirm that you have met all elements of the rubric. Use phrases that contain numbers and punctuation and spaces.

Output From Runs of Code5.c

```
Welcome to 3 STRIKES - YOU'RE OUT - the CSE version

Please pick a phrase from the following menu

1.      ---- ---- -- --------
2.      ----- --- ----
3.      - ----- ----- ---- --------
4.      -------'- ----- -- ---- -- ---
5.      -'--- -- ---
6.      -----' -- - ------
7.      (--- -----) ----- --- ---- ----- (-- -----)
8.      ---- -' --- -----
9.      ---'- ---- ------
10.     ---'- ----- --'- ----

Enter choice : 5

Here's the phrase you need to guess


-'--- -- ---

Guess a letter : a


-'--- -a ---

Guess a letter : e


-'e-- -a --e

Guess a letter : i


-'e-- -a -ie

Guess a letter : o

Strike 1

-'e-- -a -ie

Guess a letter : u

Strike 2

-'e-- -a -ie

Guess a letter : c


C'e-- -a -ie

Guess a letter : s


C'es- -a -ie
```

```
Guess a letter : w

Strike 3

3 STRIKES - YOU'RE OUT!!

Game over
```

```
Welcome to 3 STRIKES - YOU'RE OUT - the CSE version

Please pick a phrase from the following menu

1.      ---- ---- -- --------
2.      ----- --- ----
3.      - ----- ----- ---- --------
4.      -------'- ----- -- ---- -- ---
5.      -'--- -- ---
6.      -----' -- - ------
7.      (--- -----) ----- --- ---- ----- (-- -----)
8.      ---- -' --- -----
9.      ---'- ---- ------
10.     ---'- ----- --'- ----

Enter choice : 7

Here's the phrase you need to guess


(--- -----) ----- --- ---- ----- (-- -----)

Guess a letter : a


(--- ----a) ----- --- ---- ----- (-- -a---)

Guess a letter : e

Strike 1

(--- ----a) ----- --- ---- ----- (-- -a---)

Guess a letter : i


(--- ----a) -i--- --- ---- -i--- (-- -a---)

Guess a letter : o


(-o- -o--a) -i--- -o- -o-- -i--- (-o -a---)

Guess a letter : u


(-ou -o--a) -i--- -o- -ou- -i--- (-o -a---)

Guess a letter : y
```

```
(You -o--a) -i--- -o- You- -i--- (-o -a--y)
Guess a letter : r

(You -o--a) -i--- -or Your Ri--- (-o -ar-y)
Guess a letter : g

(You Go--a) -ig-- -or Your Rig-- (-o -ar-y)
Guess a letter : h

(You Go--a) -igh- -or Your Righ- (-o -ar-y)
Guess a letter : t

(You Gotta) -ight -or Your Right (To -arty)
Guess a letter : m
Strike 2
(You Gotta) -ight -or Your Right (To -arty)
Guess a letter : f

(You Gotta) Fight for Your Right (To -arty)
Guess a letter : p

You figured it out!!
The phrase was
(You Gotta) Fight for Your Right (To Party)
YOU WIN!!!!
```