CS 2123 Data Structures Recitation - Exercise 06

Use random.h interface and random.c implementation – Makefile to compile - Debugger

In this exercise, you will implement a new client/driver program using random.h. But before that, let's see how to use random.h and random.c files with the simple hilo.c client/driver program that we discussed in class. Also we will use Makefile.

First follow the "recitation" link in the class web page and download the files under `rec06-ch08a-ch03-ex01-random-h` directory or simply download its zipped version and unzip it to get all at once!

Now go into that directory (i.e., `> cd rec06-ch08a-ch03-ex01-random-h`), and execute

```
> make
```

This will compile all the programs for you and create executable files. Then you can run hilo program and play the game…

NOW, you are asked to implement another client/driver program using random.h library. Specifically, your program will allow user to play the 4 digit guessing game that I mentioned in the class. Instead of implementing everything in main() function, you will implement certain parts as functions and call them in your main() function as shown in the next page. You don't need to implement the random.h and random.c. You can simply use the functions from random library, as we did in hilo.c game.

Also add the necessary statements into the Makefile so we can compile your new program using make.

For more information about Makefile and make utility, check `ch08a-ch03-interface-lib-user-defined.ppt` slides. Also TA will help you.

Here is what your new application program should do at the high level.....

Do the followings in main():

1. Call Randomize();
2. Computer generates a 4-digit random number between 1000 and 9999 until all digits are different.
    a. secret = RandomInteger(1000, 9999);
    b. Write a function that finds the digits of secret and stores these digits in an array, say s_a.
    c. Write another function to check if all the digits are different in s_a or not.
    d. If not, go to 2(a).
3. User tries to guess the secret number within 40 attempts:

a. Ask user to enter a 4-digit number, say ans.
b. Call the same function in 2(b) to find digits of ans and store them in another array, say a_a
e. Call the same function in 2(c) to check if all the digits are different in a_a or not.
c. If not, go to 3(a).
d. Increase number of attempts.
e. Compare s_a and a_a to determine the number of digits that are in_place, out_of_place, not_exist.
f. Print these values, and go to 3(a) until secret==ans or 40 attempts are made.

Compile it using make
You can debug it using gdb or ddd

Here is an example:
Suppose computer generated 4359
If user enters, 3457, your program should say
1 digit is in place
2 digits are out of place
1 digit does not exist

Have fun!

Try your best to finish it. Even if your program is not correct, you will still better understand how to use a library, compile multiple file etc....

Also try to learn about `gdb` and `ddd` to debug your programs..

If needed, TA will go over `gdb` and `ddd` next time....

What to return: !!!! NO LATE RECITATION ASSIGNMNET WILL BE ACCEPTED !!!

DO ALL YOUR WORK UNDER **abc123-rec06** folder using your own abc123

1. First implement your program which can be named as rec06.c
2. Then compile and run it. Copy/paste the result in an output file, say out06.txt.
3. Remove executables from **abc123-rec06** folder and then Zip that folder
4. Then go to BB Learn, and submit your **abc123-rec06.zip** as an attachment before the deadline. Make sure your zip file contains all your files!

You must submit your work using Blackboard Learn and respect the following rules:

1) All assignments must be submitted as a zip file unless it is a single pdf file.
2) Assignments must include all source code.
3) Assignments must include an output.txt file which demonstrates the final test output run by the student.

If your assignment does not run/compile, the output.txt file should include an explanation of what was accomplished, what the error message was that prevented the student from finishing the assignment and what the student BELIEVES to be the underlying cause of the error.