Session 3 Lab

Purpose

The purpose of this lab is to provide an introduction to GDB and expose students to a new form of debugging. GDB will be your best friend in this course. Learn it and use it frequently.

How to Access Lab

Make sure to login to a shark machine before you begin. Follow the steps below to copy the lab into your private directory. You can obtain access to the lab by navigating to the following location on the afs servers: `/afs/cs.cmu.edu/academic/class/15213-f20/bootcamps/lab3_handout.tar`. You should copy this tar file to a location within your protected Andrew directory in which you plan to do your work.

- Navigate to a private directory where you want to complete the lab and type:
  ```
  $  cp /afs/cs.cmu.edu/academic/class/15213-f20/bootcamps/lab3_handout.tar .
  ```
- To decompress the tar file, type:
  ```
  $  tar -xf lab3_handout.tar
  ```

Part 1:

Follow the instructions to step through the phase1.c code.

```
$ gcc -o phase1 -g -std=c99 phase1.c
$ gdb ./phase1
```

Command	What did you observe?
`(gdb) break main`	
`(gdb) break unscramble`	
`(gdb) break reverse`	
`(gdb) break toggleCase`	
`(gdb) info break`	
`(gdb) run`	
`(gdb) next` (do this twice)	
`(gdb) print *(word_t*)secret_msg`	
`(gdb) next`	
`(gdb) print *(word_t*)secret_msg`	
`(gdb) continue`	
`(gdb) step`	

`(gdb) next` (do this twice)	
`(gdb) print ltr`	
`(gdb) print isAlpha(ltr)`	
`(gdb) watch ltr`	
`(gdb) continue`	
`(gdb) step`	
`(gdb) backtrace`	

BONUS:
What does `unscramble(word_t *msg)` do? And how?

What does `reverse(word_t *msg)` do? And how?

What does `toggleCase(word_t *msg)` do? And how?

Part 2:
You are given phase2.c, a file that contains buggy code! First try compiling and running the file.

```
$ gcc -std=c99 -o phase2 -g phase2.c
$ ./phase2
```

What did you observe? The next part of this lab focuses on finding out where the crash happened using GDB.

```
(gdb) run
```

What does GDB tell you about the error? Try backtracing to obtain further information.

```
(gdb) bt
```

This should reveal information about where the error is coming from. Your next step is to use GDB commands to further investigate why it is happening.

Hints: Utilize gdb commands discussed in phase 1 such as breakpoint, watchpoints to specific variables, and printing values. This is a great time to showcase your learnings from this session. It might be useful to call function such as stack_print by:

```
$ print (void) stack_print(S)
```

BONUS:
If the previous two phases were easy for you then feel free to change the code to fix the bug!

TIPS:
Talk about the help command and the shortcuts for each command.