GDB Tutorial

Purpose

The purpose of this tutorial is to introduce you to GDB and expose you to a new form of debugging. GDB will be your best friend in this course. Learn it and use it frequently!

How to Access Tutorial

Make sure to log in to your Linux environment before you begin. The tutorial materials are available on GitHub. You can clone a copy of the tutorial with the following command:

$ git clone https://github.com/CSC-225/gdb-tutorial.git

You will then be asked to enter your GitHub username and access token. If you've cloned the repository successfully, you will now have a new directory with a copy of the tutorial materials, which you can then move into using:

$ cd gdb-tutorial

Part 1:

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

$ gcc225 -o phase1 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 *(struct Word*)secret_msg	
(gdb) next	
(gdb) print *(struct Word*)secret_msg	

(gdb) continue	
(gdb) step	
(gdb) list unscramble	
(gdb) list unscramble	
(gdb) next (do this twice)	
(gdb) print ltr	
(gdb) print isAlpha(ltr)	
(gdb) continue	
(gdb) step	
(gdb) backtrace	
(gdb) continue (do this twice)	

If you entered all of the commands above, the phase1 program should have finished and printed the secret message. If you want to start the tutorial again (i.e., restart the program) without leaving gdb, you can simply use the "run" command to run the program again.

BONUS:
What does **unscramble(struct Word *msg)** do? And how?

What does **reverse(struct Word *msg)** do? And how?

What does **toggleCase(struct Word *msg)** do? And how?

Part 2:

You are given phase2.c, a file that contains buggy code! Try compiling and running the file.

```
$ gcc225 -o phase2 phase2.c
$ ./phase2
```

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

```
$ gdb ./phase2
(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 and printing values. This is a great time to test what you've learned! It might be useful to call functions such as List_print using:
$ print List_print(L)

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

TIPS:
Try using gdb's help command to get more information about each command. For instance,

(gdb) help break

This tutorial was adapted from similar materials used in CMU's 15-213 course.