Assignment #3

Topic(s): Dynamic Memory, Valgrind, Reading from Files
Date assigned: Monday, September 24, 2018
Date due: Wednesday, October 3, 2018
Points: 30

You should now have a correctly working generic static stack. Now, we have decided to change from a static stack to a dynamic stack. If we do this to a user of our static stack, we must be able to do so without the user having to alter their code in any painful way. They really shouldn't even know we've changed the implementation!

For this assignment you will:

1) create a project called **GenericDynamicStack** using:
    a. an include file called **stk.h**, which is the stack interface
    b. a source file **stk.c** which is the stack implementation
    c. a source file **stkdriver.c** which is the stack driver
    d. a make file called **Makefile** that is used to build all object files and executables for the project

2) A copy of stk.h exists on zeus in **/home/CS300Public/2018/03Files**. You are to copy stk.h from 03Files on zeus and implement each function prototype specified in stk.h in a file called stk.c. Do not modify stk.h in any way or you will lose significant points.

Here is a simple driver that tests some of your stack functions.

```c
#include "../include/stk.h"

/*************************************************************************
 Function:    main

 Description: test all the functionality of the stack

 Parameters:  none

 Returned:    Exit Status
 *************************************************************************/
int main ()
{
  Stack sTheStack;
  char ch = 'a',
       popValue;

  puts ("Program Start");

  stkCreate (&sTheStack);

  stkPush (&sTheStack, &ch, sizeof (char));

  while (!stkIsEmpty (&sTheStack))
  {
    stkPop (&sTheStack, &popValue, sizeof (char));
    printf ("Data = %c\n", popValue);
  }
  stkTerminate (&sTheStack);

  puts ("Program End");

  return EXIT_SUCCESS;
}
```

To successfully complete this assignment:

1. Create a new project called **GenericDynamicStack**. Make sure that you create a new C project->Makefile project->Empty Project->Linux GCC. Add the directories (**src**, **include**, **bin**, and **datafiles**).
2. Implement each of the functions for stk.h one at a time in a file called stk.c. You can use your testdriver(s) from GenericStackStack to test.
3. Create a **Makefile** for the project GenericDynamicStack. Watch the Makefile videos and look at the Makefile for GenericStaticStack to help you create the project.
4. Once you have implemented each function, you are to write a driver (stkdriver.c) that reads single words from a data file called palindromes.txt found in a folder called datafiles. The driver will print the word followed by [palindrome] or [not palindrome]. The driver is mostly written in 03Files and is called stkdriver.c. All you have to do is write the function isPalindrome correctly. **You can write this driver right now and test it with your GenericStaticStack.**
5. You are to submit a tarball called **cs300_3_PUNetID.tar.gz** that contains your Eclipse project (please clean) GenericDynamicStack after extensive testing on zeus.
6. You must be using Subversion. In particular, you must show me (in person) at least 5 commits of GenericDynamicStack on at least two different days by the time this assignment is due. Failure to do so will cost you 5 points. At any time, you can just show me the history of GenericDynamicStack that has at least 5 commits on two different days. DO NOT WAIT UNTIL THE LAST MINUTE!!!!!
7. You must add a valgrind target to your Makefile as shown below. The easiest way to test for

valgrind errors is using the Eclipse profiling tools but make sure to test from the command line also. Right click on the executable, then Profiling Tools, then Profile with Valgrind. You need to use valgrind often as you are coding each function

```
valgrind: bin/stkdriver
  valgrind –v --leak-check=full --show-leak-kinds=all bin/stkdriver
```

8. List how many hours you worked on the assignment in the header comments of stk.c.

If you've done this assignment correctly, you should be able to take your static stack driver and use it as a driver for the dynamic stack. No other changes are necessary. Now how cool is that!!!! ☺

If you find any mistakes or you think there are discrepancies, please email me ASAP. I will check into your issue, fix as necessary, and email the entire class if changes are made.

How to submit your project

1) clean your project and then create a tarball called **cs300_3_punetid.tar.gz** (that is "your" punetid all lowercase) that contains all files for correctly compiling your program on zeus. How to do this:
    a. at the level of GenericDynamicStack (after cleaning your project), type the command: **tar czf cs300_3_punetid.tar.gz GenericDynamicStack** OR use **make tarball** if you have a proper tarball target set up in your Makefile
2) use scp to transfer the tarball to zeus for testing
3) extract the tarball and test
4) once you are sure your program works on zeus from the command line, then submit your tarball as you did in assignment #1