Assignment #2

Topic(s): C, Makefiles, Writing modular code, Stack ADT, Multiple Projects
Date assigned: Wednesday, September 16, 2015
Date due: Wednesday, September 23, 2015
Points: 25

Part A

We have discussed the Stack ADT and looked at a static implementation of the Stack ADT. The purpose of this assignment is to have you (A) implement the Stack ADT using static memory, and (B) use the stack that you just created in a palindrome checker.

Here is what you need to do:

- Create a project called StaticStack with the folders (include, src, bin) to contain:
    - include file stk.h shown on the next page and on zeus in /home/CS300Public/2015/02Files. Copy the contents of this file into your project. Do not modify stk.h in any way!
    - src files:
        - stk.c: This will contain the implementation of all of the functions in stk.h. You are to completely write this file. Implement each function one at a time and test it using stkdriver.c as you go.
        - stkdriver.c: This will contain a driver that extensively tests each of the functions in your program. Part of your grade will be based on how well your driver tests each and every function listed above. The simple driver shown on the next page is not good for testing your project.
    - Makefile: Download MakefileSS from zeus in /home/CS300Public/2015/02Files. Copy its contents into Makefile

```c
/*****************************************************************************
File name:    stk.h
Author:       Computer Science, Pacific University
Date:         September 15, 2015
Class:        CS300
Assignment:   2 Static Stack
Purpose:      Header for a static stack with common stack operations
Hours Worked: TODO
*****************************************************************************/
#ifndef STK_H_
#define STK_H_

#include <stdbool.h>

//****************************************************************************
// Constants
//****************************************************************************
#define NO_ERROR 0
#define ERROR_STACK_EMPTY 1
#define ERROR_STACK_FULL 2
#define ERROR_NO_STACK_CREATE 3
#define ERROR_NO_STACK_TERMINATE 4
#define ERROR_NO_STACK_MEMORY 5

#define MAX_STACK_ELEMENTS 100

//****************************************************************************
// User-defined types
//****************************************************************************
typedef short int ERRORCODE;
typedef char DATATYPE;

typedef struct Stack *StackPtr;
typedef struct Stack
{
  int size;
  DATATYPE data[MAX_STACK_ELEMENTS];
  int top;
} Stack;

//****************************************************************************
// Function prototypes
//****************************************************************************
extern ERRORCODE stkCreate (StackPtr psStack);
extern ERRORCODE stkTerminate (StackPtr psStack);
extern ERRORCODE stkIsFull (const StackPtr psStack, bool *pbIsFull);
extern ERRORCODE stkIsEmpty (const StackPtr psStack, bool *pbIsEmpty);
extern ERRORCODE stkPush (StackPtr psStack, DATATYPE value);
extern ERRORCODE stkPop (StackPtr psStack, DATATYPE *pValue);
extern ERRORCODE stkPeek (const StackPtr psStack, DATATYPE *pValue);
extern ERRORCODE stkSize (const StackPtr psStack, int *pSize);

#endif /* STK_H_ */

```

```c
/*****************************************************************************
 File name:  stkDriver.c
 Author:     Computer Science, Pacific University
 Date:       9.8.14
 Class:      CS300
 Assignment: Static Stack
 Purpose:    Static stack driver to test the stack operations
 *****************************************************************************/

#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>
#include "../include/stk.h"

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

 Description: Simple driver for the stack module

 Parameters:  none

 Returned:    EXIT Status
 *****************************************************************************/
int main ()
{
  Stack sStack;
  DATATYPE data;
  bool bIsEmpty;

  if (NO_ERROR == stkCreate (&sStack))
  {
    stkPush (&sStack, 's');

    stkIsEmpty (&sStack, &bIsEmpty);
    while (!bIsEmpty)
    {
      stkPop (&sStack, &data);
      printf ("Data = %c\n", data);
      stkIsEmpty (&sStack, &bIsEmpty);
    }
    stkTerminate (&sStack);
  }

  printf ("Process Complete\n");

  return 0;
}
```

Part B

Now that you have a working implementation for the Stack ADT, you are to use this code to implement a palindrome checker. I have written all of the code in /home/CS300Public/2015/02Files. All you have to do is put everything together to produce the executable.

Here is what you need to do:

- Create a project called PalindromeChecker with the folders (include, src, bin, testcases) to contain:
    - src file:
        - palindromeChecker.c: Copy the contents of the file palindromeChecker.c on zeus into palindromeChecker.c in your project. You will not need to modify anything.
    - testcases:
        - palindrome1.txt: Copy the contents of the file palindrome1.txt on zeus into palindrome1.txt in your project. You will not need to modify anything.
        -
    - Makefile: Download MakefilePC from zeus in /home/CS300Public/2015/02Files. Copy its contents into Makefile
- Build and run the project. If all goes well, you will see:

```
PALINDROME CHECKER
---------- -------

mom [palindrome]
palindrome [not palindrome]
racecar [palindrome]
rotator [palindrome]
computer science [not palindrome]
Madam I'm Adam [palindrome]
```

Submit your Assignment

Now it's time to submit your solution. You first must create a tarball called **cs300_2_punetid.tar.gz** that contains both Eclipse projects: StaticStack and PalindromeChecker.

At the level of both folders, the command is:

```
tar czf cs300_2_punetid.tar.gz StaticStack PalindromeChecker
```

scp the file over to zeus, extract and test!!!!!

Once you are sure the tarball extracts correctly and works properly, submit the tarball as you did for assignment #1.

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 post any changes to Piazza.