CS 2123-001 Data Structures

Instructor Dr. Turgay Korkmaz

Homework 3
Due date: check BB Learn
!!!! NO LATE HOMEWORK WILL BE ACCEPTED !!!

(Command-line Arguments, Files, Abstract Data Type - Library)

You are asked to write a program to check if the HTML tags in a given file are nested correctly or not. This problem is similar to the example of proper matching of `{ ( [ ] ) },` but now you will deal with HTML tags and read them from a file. So you need to use stack ADT as we discussed in class.

[First get `booklib.zip` and `08-Abstract-Data-Types.zip` from the class web page. For this, click the link "programs from the textbook" under online materials. Then click 00-zip-files. Get README.txt and read it. Accordingly, get other files as described in it]

The implementation in 08-Abstract... has the stack library implementation and a driver program (rpncal.c) for RPN calculator. During the recitation you will modify it. But for this homework, you need to write a new application program (say htmlchecker.c), which will use the stack lib as rpncal.c does. So you will use the exisitn implementation of stack library, but in your new application since you will push/pop html tags (strings), you need to make sure stack.h has `typedef void *stackElementT;` // or
`typedef char *stackElementT;`

Background:
HTML files consists of regular text and **tags** enclosed in angle brackets, the **<** and **>** symbols. Tags are used to identify the structure of a document.

Most tags come in pairs: a beginning tag and a closing tag. For example, the tags `<title>` and `</title>` are the beginning and closing tags. There are several such tags including `<b> </b> <i> </i> <h1></h1>` etc. Tags may have several attributes and such attributes will be in the beginning tag for example `<a href="ff.html"> link </a>`.

HTML allows two-sided tags to be nested without overlapping, as in the example of proper matching of `{ ( [ ] ) }`.

Some tags are single-sided (e.g., `<img src="a.jpg" />, <br />, <hr />)` and they appear alone. They will not affect the nesting but you still need to process them since they might be in the file. You can simply ignore the tags start with < and end with />
Also the file may contain HTML comments such as

```
<!-- comments contain <p> <\p> -->
```

Your program should ignore everything between `<!--` and `-->`

For example, if an HTML file contains

```
<title><b> THIS FILE </b> USES CORRECTLY NESTED TAGS
</title>
<h1><i> First <b class="c1"> header </b> text <img
src="pic.jpg" /> </i></h1>
<!-- comments contain <p> <b> <\p> -->
<p id=par1"> Some other text </p>
```

Then YES, all the tags are nested correctly. But if an HTML file contains

```
<title> <b> THIS FILE </title> IS </b> NOT NESTED
CORRECTLY.
<p> <b>  some text is not nested correctly </b>
```

Then NO, the tag `<B> </title>` violates the proper nesting!

Your program must accept HTML input file name as a command line argument and process it for proper nesting of HTML tags. So we will run your program as follows

```
  main212> driver sample1.html
```

You can stop the program when you detect the first tag that violates the proper nesting structure and print NO and the tag that violates nesting on the screen. Otherwise, your program will print YES, all the tags are nested correctly.

As always, make sure you release (free) the dynamically allocated memories if you allocate any memory in your programs. So, before submitting your program, run it with `valgrind` to see if there is any memory leakage… Also if you need to debug your program, compile your programs with –g option and then run it with `gdb` and/or `ddd`.

What to return: !!!! NO LATE HOMEWORK WILL BE ACCEPTED !!!

1. Create a directory, say LASTNAME_hw3, and do all your work under that directory.
2. If needed, you will get stack library from the slides and use it along with other libraries in your driver/client program, `driver.c`.
3. To easily compile the stack library and driver program, you must have a `Makefile` and use "`make`" to compile your code.
4. After compiling, run your program a few times with different data files, then save the output (using script) into `output.txt` file. So you will have around 6-7 files in your LASTNAME_hw3 directory.
5. Go to parent directory of LASTNAME_hw3, and use

```
> tar -cf  LASTNAME_hw3.tar  LASTNAME_hw3
```

This will create a new file called LASTNAME_hw3.tar and it contains all of your files. So just submit this .tar file.

6. Go to BB Learn, and just submit LASTNAME_hw3.tar as **attachment** before the deadline. DO NOT submit other .h or .c files individually.

/* Don't forget to include comments about the problem, yourself and each major step in your program! */

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

1) All assignments must be submitted as either a zip or tar archive 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.
4) 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.____