CS160 ARM Programming Assignment 3

STRING SUBSTITUTION

Problem description

In this lab you will write an ARM assembly language subroutine for substition of a target string with a replacement string inside a text. You will also write a C program for tying everything together plus handle the input and output. See your textbook and ARM lab manuals for details of ARM assembly language techniques.

Getting more specific

Your code must do the following:

Assignment Write a C program called `edit.c` that reads in a file (whose name is specified as argument) and stores it as an array of bytes `text[]` terminated by a zero byte (null character). Your C program should then read in pairs of strings from the user, one string per line, storing them in two null-terminated arrays `target[]` and `replace[]`. Your program should then call the routine `strsub(char *text_in, char *text_out, char *target, char *replace)`. The routine takes as an input a null-terminated array `*text_in` and produces its result in another array `text_out[]`. The array `text_out[]` should take the value of `text_in[]` with all occurances of `target[]` replaced by `replace[]`. The return value is the number of substitutions made.

Your program should print the stored text before the first call to `strsub()` and after every call, and should also print the number of substitutions made by each call.

Your C program may use two buffers to store the input and output of `strsub`, but it should not copy data between them.

The maximum size of the stored text is 1000 bytes, and the maximum size of each of `target[]` and `replace[]` is 10 bytes. Your program should use these limits to ensure that your program's data structures never exceed the memory allocated to them.

Hints It is possible that two instances of `target` can overlap, as in this example: `text = ''ababaa'', target = ''aba'', replace = ''d''`. When two instances of `target` overlap, the first should be replaced and not the second. So in this case the result should be `''dbaa''`.

The result of a replacement may introduce another instance of `target`, as in this example: `text = ''hellollo'', target = ''hello'', replace = ''he''`. Any instance of `target` which is introduced as the result of a replacement should not be be replaced again in the same call. So in this case only one replacement would occur, and the result should be `''hello''`.

Due date and submission policy

You have two weeks to finish this lab. Submit `edit.c`, and `strsub.s` to your lab GTA no later than 5pm on Sunday, December 3, 2005.