Assignment 2

CS392 Spring 2019

Task Details

Implement a shared library libcs392string.so, which provides the APIs for memory copy and string length calculation.

Requirement

1. Your project should include two source files “cs392_memcpy.c” and “cs392_strlen.c”.

2. The “cs392_memcpy.c” file should define a function “void * cs392_memcpy (void * dst, void * src, unsigned num)”. It copies “num” bytes of data from the memory pointed to by “src” to the memory pointed to by “dst”. This function returns “dst”.
    a. Hint: “dst” and “src” are of type “void*”, which cannot be directly dereferenced. Consider pointer casting.
    b. Hint: when you access an element with pointer dereference, you will be accessing the whole element. Hope this help you determine which pointer type to cast to.

3. The “cs392_strlen.c” file should define a function “unsigned cs392_strlen(char *str)”. It calculates the length of the string pointed to by “str”, and returns that length.
    a. Hint: the end of a string is marked by the null byte ‘\0’.
    b. Hint: we do not consider ‘\0’ when calculating the length of a string

4. You project should include a header file “cs392_string.h” to provide prototypes (i.e., declarations) of the above two functions.

5. Your project should include a “Makefile”, which builds your source code to “libcs392string.so”.
    a. You are expected to include explicit rules to compile the “.c” file into object files (“.o” files)
    b. You are expected to include explicit rule to link the objects files to generate libcs392string.so
    c. You need to include a “clean” target in your Makefile, which, if executed, can remove the object files and the libcs392string.so.

6. Zip all the files into a cs392_ass2.zip and submit the .zip file only

Grading

1. Cannot compile or run: **Will not be graded** (I mean it this time)
2. No statement of “I pledge my honor that I have abided by the Stevens Honor System.” as comment in the beginning of your code: **Will not be graded** (I mean it this time)
    a. Place the honor statement in the beginning of each “.c” file and “.h” file as comment
3. Pass the test case: +50
    a. Pass cs392_memcpy (25)
        i. You will not get this by calling “memcpy” inside “cs392_memcpy”
    b. Pass cs392_strlen (25)
        i. You will not get this by calling “strlen” inside “cs392_strlen”
4. Include the header file: +10
    a. Correct prototype for cs392_memcpy (4)
    b. Correct prototype for cs392_strlen (4)
    c. Avoid duplicated inclusion (2)
5. Makefile +40
    a. Correct explicit rule to compile cs392_memcpy.c into object file (10)
    b. Correct explicit rule to compile cs392_strcpy.c into object file (10)
    c. Correct explicit rule to link the objects files to generate libcs392string.so (10)
    d. Correct “clean” rule (10)

Testcase:

https://drive.google.com/file/d/1vh7QaEcTtLGZ_WbNRJGvZvbDPz3ikNQB/view?usp=sharing

Download the testcase, compile it, and link it with your libcs392string.so.
How to compile, link and test:
[1] Place the testcase in the same folder as your libcs392string.so and cs392_string.h
[2] Run: **gcc cs392_ass2_test.c -o test libcs392string.so**. This will generate a program “test”
[3] Run: **export LD_LIBRARY_PATH=$PWD**
[4] Run: **./test**

If you made everything correct in your libcs392string.so, you will be able to see:

“Congratulations, you passed the memcpy task
Congratulations, you passed the strlen task”