LAB FOUR

FILES, HEAP MEMORY

CS2263, Fall 2021

LEARNING OUTCOMES

At the conclusion of the lab, students should be able to

- Write C programs that read and write numbers and strings to and from files
- Write C programs that perform elementary heap memory management

RESOURCES

This lab will build on your forNextDay() Strings.c/.h module

EXERCISE ZERO

Copy your existing String module into your lab source code location and build/modify a Makefile for it.

EXERCISE ONE

Consider how to write a string to a file such that you can easily read it back in, remembering that

1. You will need to allocate the memory for the string
2. When you begin reading the string you don't know how long it will be.

This seems like an unsolvable problem. While there are solutions to this, we haven't learned enough yet. In the meantime, how could you write the string out so that the program reading it in would know its length when it started?

Write the function to do this:

```
int fputString(FILE* pFOut, char* s);
```

that writes the string to the file, preceded by the length of the string as an integer, e.g.

```
8 I love C!
```

Now write a test driver program that reads two strings (assume less than 100 characters each) from standard in using fgets() and writes them to an open file (file name from the command line arguments) using your function. Of course you should add this as a new target in your Makefile before you get started.

SUBMIT:

- A screen shot of the make command at work, test of the program and the text file that is produced.

EXERCISE TWO

Using your own functions from your Strings module to help, write a function

```
char* fgetString(FILE* pFIn);
```

that reads the string length from an open file, allocates the memory, reads the string into it and returns it.

Now write a test driver program that opens a file (file name from the command line arguments) and reads all strings from a file and writes them, one string per line, to standard out. Of course you should add this as a new target in your Makefile before you get started.

Careful:

1. You need to make sure that the existing functions in your Strings module work correctly. If they don't – fix 'em!
2. Don't forget that the memory for the string will need to be `free()`'d when you're done with it!
3. You may wish to run the valgrind utility to check for potential memory leaks. See Chapter 9 as well as Chapter 5.4 of your text for more.

SUBMIT:

- A screen shot of the make command, test of the program and the text file that is produced.

EXERCISE THREE

Push your modified Strings module files to the FCS git

SUBMIT:

- the modified source code, including the Makefile
- the screenshot of you pushing the program source to the FCS git

SUBMISSION

Before the due date for this lab, students should submit a single zip or tar file (named *LastName_FirstName_Lab4.zip or LastName_FirstName_Lab4.tar*) online to the lms containing:

- the required material for each question (use the headings indicating the question number) in a single pdf file (named *LastName_FirstName_Lab4.pdf)*
- Your source code directory:
    - This should include all of your source files, including any test programs.
    - This should not include object (.o) files and executables. Nobody needs to see those.