Program 2: Search and Replace

CSCI 2212, Fall 2015

1 Goals

- To read a file and write a file.
- To use a c string for input.
- To use functions from the C string library.
- To produce work that can COPE: that is, be clear, organized, precise, and economical.

2 Specifications

Scope:
Write a program to search a text file for a keyword and write out a new text file with that keyword replaced by a different word.

Detailed Problem Statement:

- Input: Four things to read from the keyboard.
  - The name of an input file that contains the text to be processed.
  - The name of an output file to contains the finished text.
  - key : A word to search for.
  - replacement: A different word to substitute for the key.
- Outputs:
  - The keyboard inputs should be echo-printed.
  - An output file named "P2out.txt" that is just like the input file except that the key word has been replaced by the replacement word.
- Constants: `#define` two symbolic constants, MATCH and NOMATCH.

3 Submission

Use your mouse to capture the output from two test runs and put it into a text file. Use plain text always. DO NOT put any part of your code or your output into a Word file or an rtf file.

Put all parts of your code, input file, output file, and screen output into a folder. Give the folder a name of the form P2 followed by your name. Zip it up and email it to Dr. Fischer's home address or submit it on Dr Alsmadi's Blackboard site.

4 Assessment and Grading

This 10-point program is due Friday, September 18. Your work will be assessed, marked up, and graded, as follows:

1. Clarity: 1	–All variables and functions have reasonable names, written in camelCase. –Every function starts with a brief comment that identifies its main purpose. –No line of code or comment exceeds 80 characters. –Error reports clearly identify the reason for aborting execution.
2. Organization: 2	–The required functions are present. –Every function starts with a visual dividing line // ——————————— –No line of code or comment exceeds 80 characters. –Code is formatted reasonably and indented properly, according to the instructor's standards.
3. Precision: 6	–The results in the output are correct, given the corresponding input. –File opening errors are handled. –End of file is handled appropriately –The program processes a file according to the above instructions. –Correct output is submitted for two test cases. –Required error checks are made in appropriate places so that the program does not attempt to process malformed or missing data.
4. Economy 1	–Comments do not repeat the obvious. –Variable and function names are not excessively long. They begin with the important information. –Code does not contain long redundant areas. –Code is written as simply as possible.