ENGG*1410: “Introductory Programming for Engineers”, Assignment #7 Strings

Prof. Shawki Areibi
School of Engineering, University of Guelph
Fall 2021

Start Date: Week #7, Due Date: Week #8 (Friday, 5:00 PM) in Dropbox

1. Write a function called *intToStr()* that converts an integer value into a character string. Be certain the function handles negative integers properly.

2. If *c* is a lowercase character, the expression
   c - ’a’ + ’A’
   produces the uppercase equivalent of c, assuming an ASCII character set.

   Write a function called *uppercase()* that converts all lowercase characters in a string into their uppercase equivalent.

3. Write a function called *substring()* to extract a portion of a character string. The function should be called as follows
   substring (source, start, count, result);

   Where *source* is the character string from which you are extracting the substring, *start* is an index number into source indicating the first character of the substring, *count* is the number of characters to be extracted from the source string, and *result* is an array of characters that is to contain the extracted substring. For example, the call
   substring(“character”, 4, 3, result);

   extracts the substring “act” (three characters starting with character number 4) from the string “character” and places the result in *result.*

   Be certain the function inserts a null character at the end of the substring in the *result* array. Also, have the function check that the requested number of characters does, in fact, exist in the string. If this is not the case, have the function end the substring when it reaches the end of the source string. So, for example, a call such as:
   substring(“two words”, 4, 20, result);
   should just place the string “words” inside the result array, even though 20 characters were requested by the call.

4. Write a function called *findString()* to determine if one character string exists inside another string. The first argument to the function should be the character string that is to be searched and the second argument is the string you are interested in finding. If the function finds the specified string, have it return the location in the source string where the string was found. If the function does not find the string, have it return -1. So, for example the call

index = findString("a chatterbox", "hat");
searches the string "a chatterbox" for the string "hat". Because "hat" does exist inside the source string, the function returns 3 to indicate the starting position inside the source string where "hat" was found.