**CMPE 102: Fall 2021**

**Program #3: Search (100 Points)**

**Due Date: Nov 16, 2021**

Description:

You are responsible to write an assembly function to perform searching of a list of names in stored in ascending order. This function will use the linear search algorithm. The names could be stored in mixed case. Therefore your assembly function must do the comparison in a ***case insensitive***manner. This function will use the C function signature but everything within this function should be assembly code using the ASM block similar to the assembly example shown in class.

Program Specification:

long search\_by\_name (char\* list, long count, char\* token);

list – the starting address of the list of structures to be searched

count – total number of names in the list

token – name to be search in the list

This function returns the Student ID of the student record in the list. While the list was constructed, the ID was populated in sequence starting with 1001. If the name is not found, then it returns 0 and print an error message on the console.

Additional Information:

* We do NOT support partial search, it means that it is all or nothing.
* Both the file name and search token are being passed by command line arguments.
* The output is display on the screen.
* You MUST have detailed comments within the ASM block for your algorithm (per line).

Program Checklist:

For program #3, you will submit your p3.cpp and a file named readme.p3 text file with any comments (things that work or not work about your code besides the required header (name, student ID, OS type, 32/64 bit, and compiler info). You MUST be able to compile and run the program from the command line). If we can't test your program, you're not going to get much credit for your work!!!

Given the same test data, the output of your ASM function should match exactly the output of the output from the given C function. There are at least different scenarios you need to test.

#1 Test existing elements with exact match

#2 Test case insensitivity

#3 Test partial search & non-existent elements