Bradley Grose

CSCE 212 Fall 2020

Dr. Wang

Project 2

1.0 Project Description

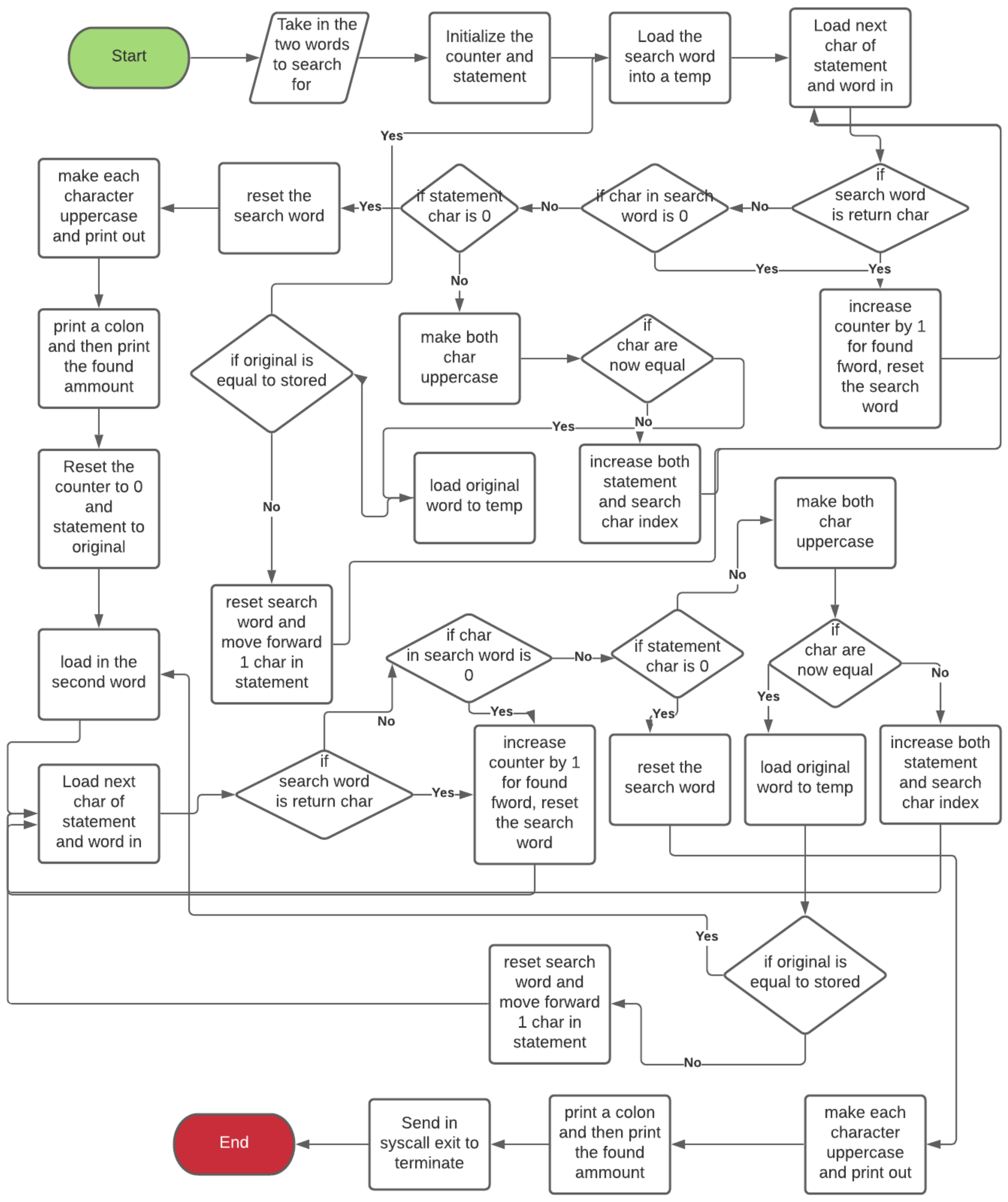
For this project, I utilized MIPS assembly language in order to search a statement for two words the user will input into the code. The statement it will search is as follows:

"The South Carolina Gamecocks football program represents the University of South Carolina in the sport of American football. The Gamecocks compete in the Football Bowl Subdivision of the National Collegiate Athletic Association (NCAA) and the Eastern Division of the Southeastern Conference. Will Muschamp currently serves as the team's head coach. They play their home games at Williams-Brice Stadium. Currently, it is the 20th largest stadium in college football"

The words input by the user should not be case sensitive and capitalization should not matter. The code, in exchange, will return the number of times those words are found in the above statement, even if it is none.

2.0 Program Design

To Start my code, I have the statement as well as the prompts hard coded in in the data section. I also have word memory space set to hold the value of searching words. To start the code, I use syscall to prompt the user and collect the two strings I will be searching for. Then, I load in the statement as well as initialize a counter. The following steps are run twice, however one time with the first search word and one time with the second search word. I will then proceed to move the search term into a temp variable. I will then load the next character of the statement and search word into another temp. If the search word next character is zero or new line, it will increase the counter as that means the word was found. If the statement next char is null, then the whole statement has been searched, so it will go to the end process. If none of those are true, each character will be made uppercase using a jump and link that checks if it is lowercase and if it is, it will subtract 32 from the value making it uppercase. This is then returned. If those characters are not equal, the code will reload in the search word and move forward in the statement and jump back to the character parse. If they are the same, then the code will move forward one index on both of the terms to look for the next character to match. If the end case is called, the original search word will be loaded into the temp again. Then, one by one, the code will use the uppercase algorithm to make the character uppercase and print it out. Once the sentence is printed, the count is also printed. Once both cases are run, syscall is used to exit the code.


Start
Take in the two words to search for
Initialize the counter and statement
Load the search word into a temp
Load next char of statement and word in
if search word is return char
if char in search word is 0
if statement char is 0
reset the search word
make each character uppercase and print out
print a colon and then print the found ammount
Reset the counter to 0 and statement to original
load in the second word
Load next char of statement and word in
if original is equal to stored
reset search word and move forward 1 char in statement
make both char uppercase
if char are now equal
load original word to temp
increase both statement and search char index
increase counter by 1 for found fword, reset the search word
if char in search word is 0
if statement char is 0
if search word is return char
reset the search word
load original word to temp
increase both statement and search char index
if original is equal to stored
reset search word and move forward 1 char in statement
make each charcter uppercase and print out
print a colon and then print the found ammount
Send in syscall exit to terminate
End
Yes
No

3.0 Symbol Table

Registers	Purpose/Where Its Used
statement1	An asciiz value that holds the string of the statement to be searched.
prompt1	An asciiz value that holds the prompt for the first input word.
prompt2	An asciiz value that holds the prompt for the second input word.
colonspace	Ab asciiz value that holds ": " to be used when printing results.
firstWord	A .word value of size 10 that holds the string of the first search word.
secondWord	A .word value of size 10 that holds the string of the second search word.
newline	An asciiz values that holds a newline command to be used when printing results for cleanliness.
$v0	Used to store the syscall int value and to take in the values for syscall.
$a0	Sends the value in to a function or to be printed out by syscall.
$a1	Used to set the character limit when intaking strings into the code.
$t0	Used to store the statement and parsed statement throughout the code. Also, when printing, the original string is loaded into $t0.
$t1	Used to store the firstWord for search and the parsed firstWord as I run comparisons.
$t2	Used to store the secondWord for search and the parsed secondWord as I run comparisons.
$t3	Used as a counter for whenever the match was found to be printed as a result.
$t4	Used to store the comparison character for the search word.
$t5	Used to store the comparison character for the statement.
$t6	Used to load in the search word to compare it to the parsed word if the words do not match to reload in original.
$ra	Return address for jump and link command.
main	Label used for the user input section of the code.

start1	Label that loads in the first search word into temp variable.
searchFirstWord	Label that runs checks to see if the word is a match in addition to calling other labels to be used for comparison
nextChar1	Label that is called if the words do not match, it will reset the original word and move forward in statement.
increase1	Label is called when a match in words is found and adds one to the counter.
end1	Label is called after statement is searched. This resets to the value of the original word for printing.
count1Loop	This label goes through and uppercases each character and then prints it out.
print1	This prints out the separator for the string and then prints out the resulting times it was found.
start2	This Label reinitializes the counter as well as the statement. In addition, it loads in the second word.
searchSecondWord	Label that runs checks to see if the word is a match in addition to calling other labels to be used for comparison.
nextChar2	Label that is called if the words do not match, it will reset the original word and move forward in statement.
increase2	Label is called when a match in words is found and adds one to the counter.
end2	Label is called after statement is searched. This resets to the value of the original word for printing.
count2Loop	This label goes through and uppercases each character and then prints it out.
print2	This prints out the separator for the string and then prints out the resulting times it was found. It also uses syscall 10 to terminate the program.
toUpper	This label is called to make a sent in character uppercase.
return	This label returns said character to where it was linked to.

4.0 Learning Coverage

Some technical topics learned and used in this project are below

1. Using jal to jump and link to a function
2. Using addi to change a letter from upper case to lower case by subtracting 32
3. Creating methods using Labels and jump functions
4. Creating return values using jr command
5. Setting a character limit and using that to go through byte to compare

5.0 Test Plan

I set up this test plan so that the code will run with a few different conditions. The first prompt is the one given as the example which looks for words found 2 and 4 times with sometimes that same capitalization, sometimes not. The second one checks that capitalization works by sending in 2 different variation of gamecocks with different capitalization to verify. Finally to show it works on more word, I used of which is the most common word and a random group of letters to make sure it returned 0.

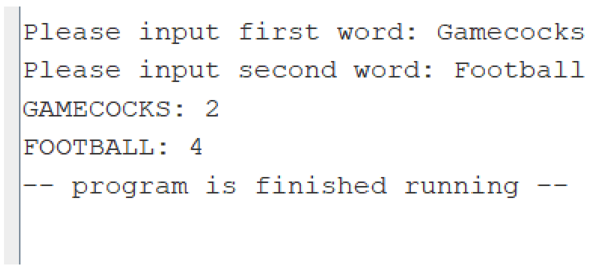
Plan 1: Run the given example, sending in “Gamecocks” and “Football”. Should return 2 and 4 respectively

Plan 2: Run using “gAmEcOcK” and “gameCOCK” to confirm that both return 2 no matter capitalization

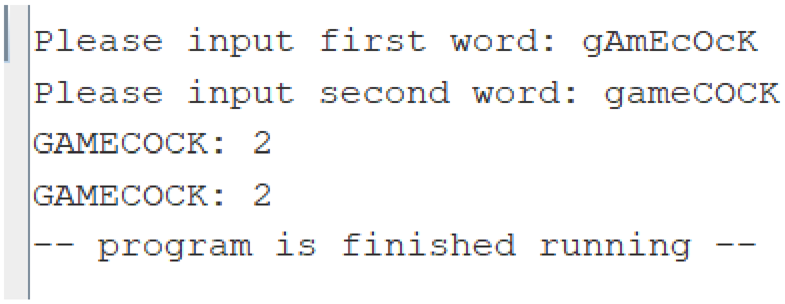
Plan 3: Run using “the” and “asadfg” which should return 13 and 0 respectively.

6.0 Test Results

Test1: Successful

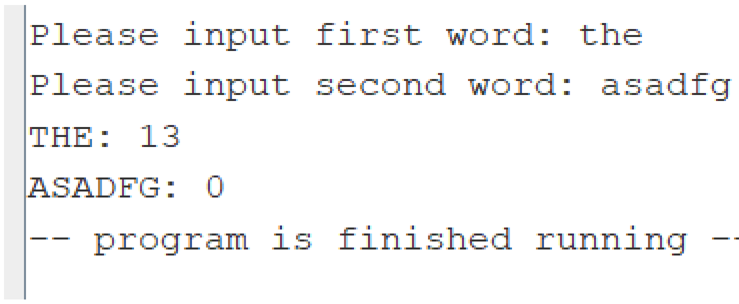

```
Please input first word: Gamecocks
Please input second word: Football
GAMECOCKS: 2
FOOTBALL: 4
-- program is finished running --
```

Test2: Successful



```
Please input first word: gAmEcOcK
Please input second word: gameCOCK
GAMECOCK: 2
GAMECOCK: 2
-- program is finished running --
```

Test3: Successful



```
Please input first word: the
Please input second word: asadfg
THE: 13
ASADFG: 0
-- program is finished running --
```