ASSIGNMENT 1 STAGE 2 REPORT

Sreemanti Dey

January 2022

1 Objective

An assembly program that merges two sorted lists of strings, maintaining the sorting order. It supports the two comparison modes - case-insensitive and case-sensitive and gives an option for removing the duplicates.

2 Assumptions

ARMSIM VERSION - 2.0.1
ANGEL SWI

3 Design Decisions

I have allocated a space in the merge function for the merged list of pointers, and the caller is aware of the context i.e. it knows that the two input sorted lists of strings are contiguously stored so the caller (here it is the main function) copies the merged list of pointers back to the place of the original input list of pointers and thus disposes off the space that had been allocated to the merged list of pointers.

4 Implementation details

4.1 Details of merge.s file

1. Merge routine performs merge of 2 sorted lists of strings. I have passed the pointers to the starting addresses of the memory spaces that contain addresses of our string lists, in r0 and r1. And I have allocated a common static space called MergedListOfPointers that has the sorted list of pointers.

2. In my merge.s file, I have assumed sizes of the strings are not zero.

3. I have stored the pointers r0-r3, which are being used by the callee, i.e. the compare routine so that the information does not get lost.

4. In the case when duplicate removal option is 1, I consider the duplicates to be placed in my final merged list from the first list, for eg, if my first list has ABC and second list has ABC, then I will keep the ABC from my first list in my final list, similarly if the case-insensitive option is turned on, and I have abc in my first list and ABC in my second list, I keep abc from the first list in my final list, as is also evident from the test cases that I have added in my report.

4.2 Details of UsefulFunctions.s file

I have used UsefulFunctions.s for I/O handling, after making few corrections as suggested on piazza and I have also added support for backspace key. I added if character read in is of ASCII 0x08 then we should delete the previous character from the buffer and continue reading.
I used the atoi, itoa, strlen, fgets and prints routines from the UsefulFunctions.s file.

4.3 Details of main.s file

1. My main.s file calls merge routine and displays appropriate output based on the user input.

2. I have preserved register values in my main.s file before calling any function, mainly those which are used by callee and lie within r0-r3.

3. Since we have to take in so many inputs, I have assigned a memory space called auxiliaryinfo that stores the sizes of the 2 string lists, comparison mode and duplicate removal option.

4. To conserve as much space as possible, I have assigned a space for my string list and another space for my list of string pointer, each pointer can be assumed to be fit inside 4 bytes, since pointer stores address of my strings. Also, I am storing all the strings contiguously in one space, each taking space equivalent to just the length of the string and I have also given 2 extra bytes for each string to accomodate null character as well as carriage return character which is added sometimes when we give a newline for users using the Windows OS.

4.4 Error Handling

1. If the comparison mode entered is anything other than 0 or 1, I print Invalid Input and exit my program.

2. If the duplicate removal option entered is anything other than 0 or 1, I print Invalid Input and exit my program.

3. If user inputs sizes of a string as 0, I give an error, saying Invalid Input and exit my program.

5 Results

Test Inputs:

1. The user gets a prompt for every input he/she has to make. The input prompt gives a "Enter the size of first list of strings:" prompt for entering the size of the first list of strings, "Enter the first list of strings:" prompt for entering the first list of strings, "Enter the size of second list of strings:" prompt for entering the size of the second list of strings, "Enter the second list of strings:" prompt for entering the second list of strings, "Enter comparison mode (0 for case-insensitive and 1 for case-sensitive):" prompt for entering comparison mode, "Enter the duplicate removal option (0 if duplicates not to be removed and 1 if duplicates are to be removed):" prompt for entering duplicate removal option.

2. Comparison mode has 0 as case-insensitive mode and 1 as case-sensitive mode.

3. Duplicate removal option takes in 0 if duplicates are not to be removed and 1 if duplicates are to be removed.

Test Outputs:

1. First the size of the merged list of strings is printed.

2. The merged list of strings are printed one by one,each in a new line

6 TestCases

1. Enter the size of first list of strings:
   3
   Enter the first list of strings:
   A
   B
   C
   Enter the size of second list of strings:
   3
   Enter the second list of strings:
   a
   b
   c
   Enter the comparison mode(0 if case-insensitive and 1 if case-sensitive):
   0
   Enter the duplicate removal option (0 if duplicates not to be removed and 1 if duplicates are to be removed):
   0
   The size of final list of strings:

6
The final list of strings:
A
a
B
b
C
c

2. Enter the size of first list of strings:
3
Enter the first list of strings:
ABC
pq
Z12#
Enter the size of second list of strings:
2
Enter the second list of strings:
abc
def
Enter the comparison mode(0 if case-insensitive and 1 if case-sensitive):
0
Enter the duplicate removal option (0 if duplicates not to be removed and 1 if duplicates are to be removed):
1
The size of final list of strings:
4
The final list of strings:
ABC
def
pq
Z12#

3. Enter the size of first list of strings:
2
Enter the first list of strings:
abc
def
Enter the size of second list of strings:
3
Enter the second list of strings:
ABC
pq
Z67%666
Enter the comparison mode(0 if case-insensitive and 1 if case-sensitive):

1
Enter the duplicate removal option (0 if duplicates not to be removed and 1 if duplicates are to be removed):
0
The size of final list of strings:
5
The final list of strings:
ABC
abc
def
pq
Z67%666

4. Enter the size of first list of strings:
   4
   Enter the first list of strings:
   abc
   bgh
   jki
   poi
   Enter the size of second list of strings:
   1
   Enter the second list of strings:
   cfg
   Enter the comparison mode(0 if case-insensitive and 1 if case-sensitive):
   0
   Enter the duplicate removal option (0 if duplicates not to be removed and 1 if duplicates are to be removed):
   1
   The size of final list of strings:
   5
   The final list of strings:
   abc
   bgh
   cfg
   jki
   poi

5. Enter the size of first list of strings:
   3
   Enter the first list of strings:
   aaaaaaaaa
   aaaaaaaaab
   bgh12345
   Enter the size of second list of strings:

2
Enter the second list of strings:
aaaaaaaaa
bbbbbbbbb
Enter the comparison mode(0 if case-insensitive and 1 if case-sensitive):
1
Enter the duplicate removal option (0 if duplicates not to be removed and 1 if duplicates are to be removed):
1
The size of final list of strings:
4
The final list of strings:
aaaaaaaaa
aaaaaaaaab
bbbbbbbbb
bgh12345

6. Enter the size of first list of strings:
5
Enter the first list of strings:
abd
ABE
CDF
efghi
mpq
Enter the size of second list of strings:
2
Enter the second list of strings:
CDf
MPQ
Enter the comparison mode(0 if case-insensitive and 1 if case-sensitive):
0
Enter the duplicate removal option (0 if duplicates not to be removed and 1 if duplicates are to be removed):
1
The size of final list of strings:
5
The final list of strings:
abd
ABE
CDF
efghi
mpq

7. Enter the size of first list of strings:

2
Enter the first list of strings:
cDf
MPq
Enter the size of second list of strings:
5
Enter the second list of strings:
abd
ABE
CDF
efghi
mpq
Enter the comparison mode(0 if case-insensitive and 1 if case-sensitive):
0
Enter the duplicate removal option (0 if duplicates not to be removed and 1 if duplicates are to be removed):
0
The size of final list of strings:
7
The final list of strings:
abd
ABE
cDf
CDF
efghi
MPq
mpq