Divyansh Mittal , 2020CS10342 Computer Architecture cs1200342@iitd.ac.in

1. Assignment 1

a) Stage 2

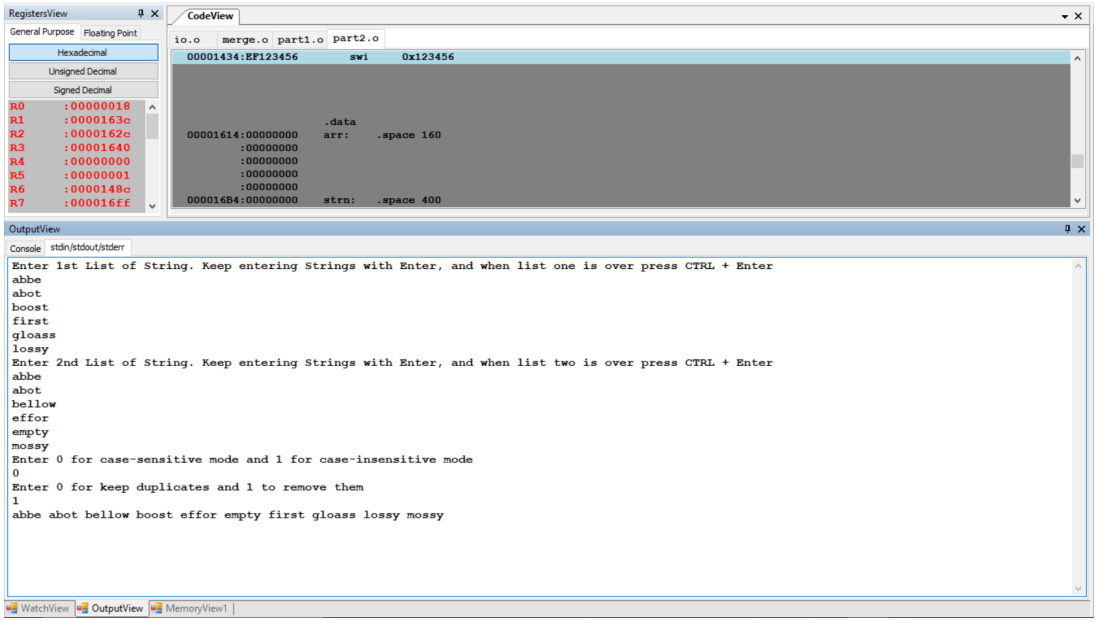
- **Input:** To call the merge function, r0 is start of list 1, r1 has size of it r2 has pointer to list 2, r3 has size of it, r4 can be used to specify comparison mode, and r5 for removal of duplicate.

  When running directly , run main_ part2.s Enter a list by separating strings by enter. When done with list 1, press Ctrl + Enter [1]. Do the same for list 2. Then enter 0 or 1 for case sensitivity mode and similarly for removal or not of duplicates.

- **Output:** The r0, r1 has the locn and size of merged list. The merge uses temporary space to sort, but replaces the original list too, hence can be easily used for merge sort

  If directly running you will get the sorted list printed. [2]

Working: It works via a two pointer method. In the function merge, initially r0 points to list 1 start, r1 points to list 1 end, r2 points at list 2 start and r3 points to list 2 end. String at r0,r2 pointer are compared. Whatever is smaller, r6 places pointer at a temp locn , advances and the one which is placed is also advanced. This loop continues till r0 becomes more than r1 or r2 becomes more than r3. In this case the rest of pointers are copied at the temp locn. If we have to remove duplicate, and a duplicate occours, the val at r0 is placed at r6 and both r0,r2 advance together. The final merge list is coppied back at the original r0 that was passed and r1 holds size of sorted list.



[1] For windows, pressing enter sends '\ r' char and pressing ctrl + enter registers as '\ n'. The io replaces '\ r','\ n' with the null character

[2] The IO does not add any "\ n" or space char. So enter space with the word to get spaced out output. In the memory the strings however are null terminated

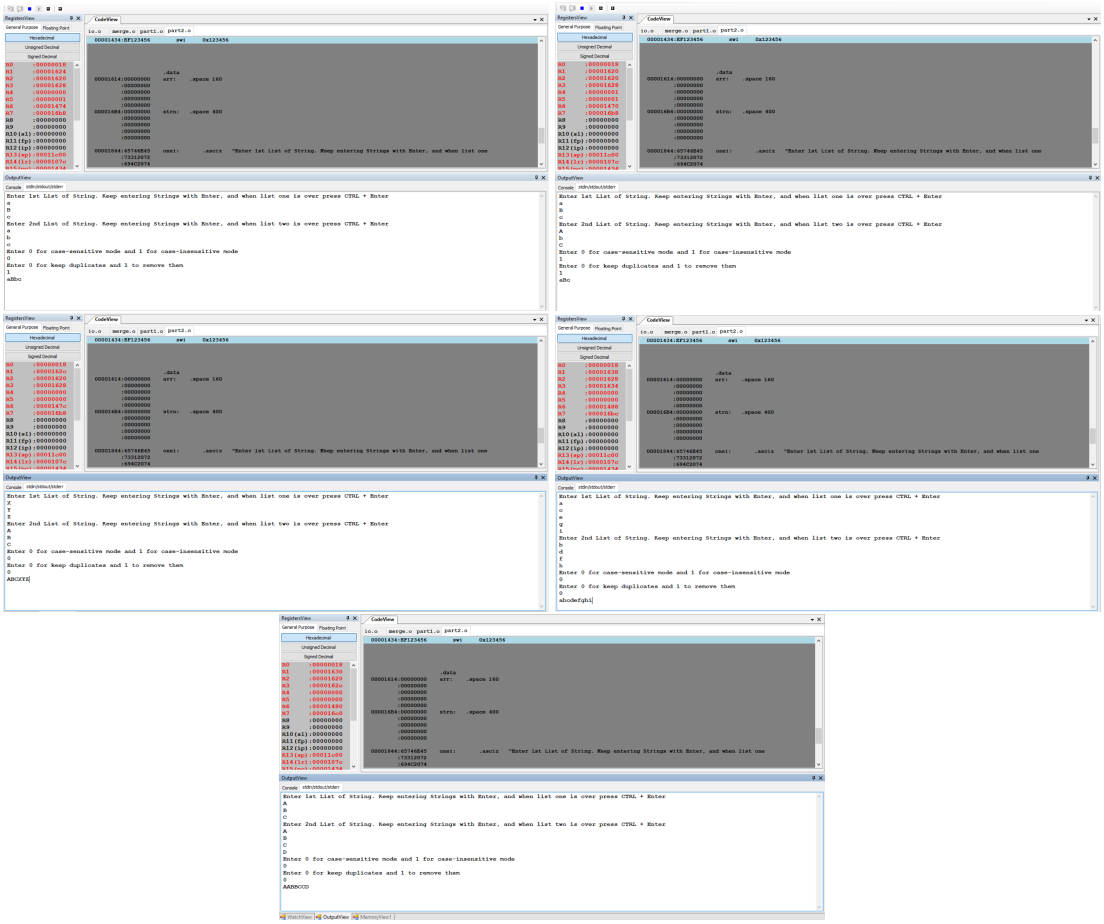


Figure 1: Various test cases

b) Stage 1

- **Input:** To call the comparison function, r1 and r2 can be used to pass pointer to string 1 and string 2. r3 can be used to specify comparison mode. Place 0 for case-sensitive mode and 1 for case-insensitive mode

  If directly running part1.s, enter string 1, string 2, and comparison mode (0 or 1) via stdin[3]

- **Output:** If you call the comparison function, result is stored in r0. -1 is returned if string2 > string 1, 0 if string1 = string2 and 1 if string1 < string 2

  If directly running part1.s, stdout text to tell which string is bigger

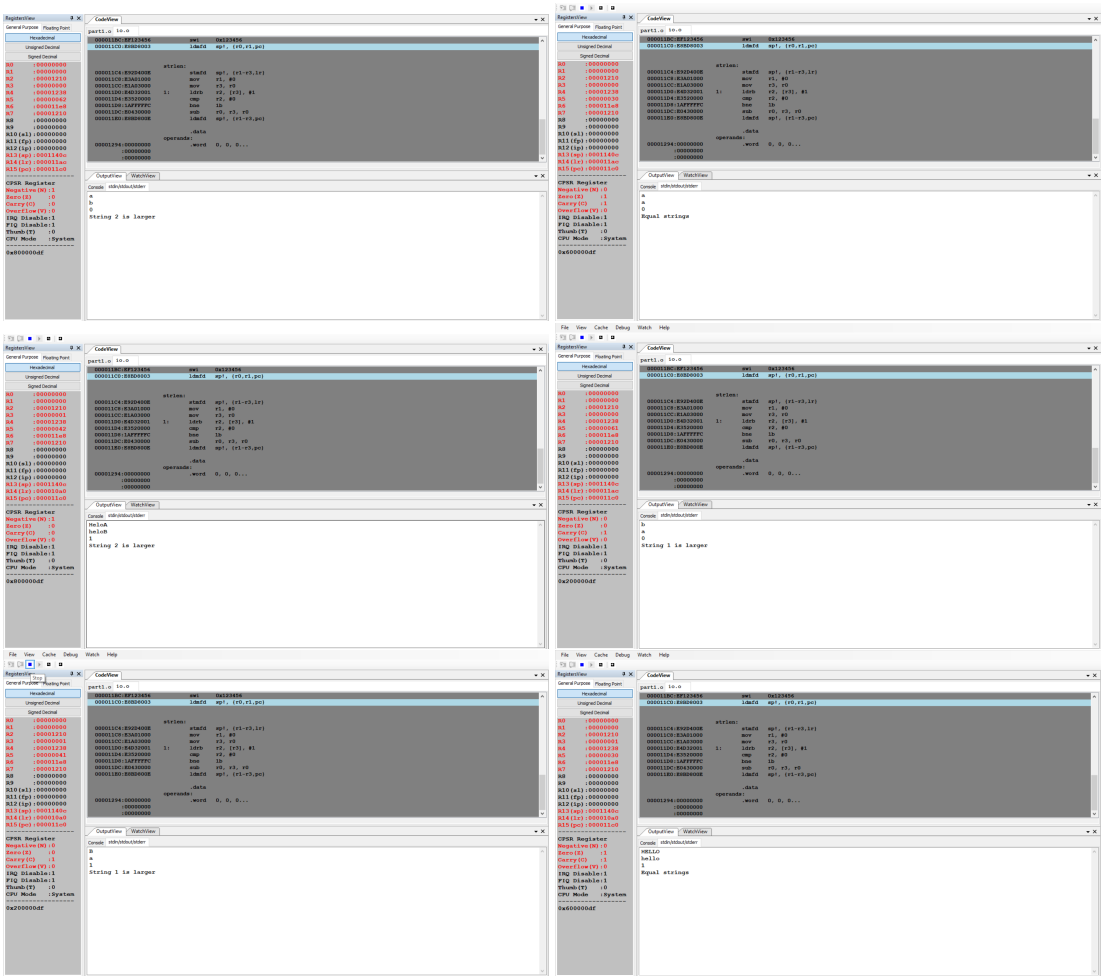


Figure 2: Various test cases

[3]In direct run r0 will get 0 as print is called