Group No.:19

Student IDs: 20607620,20610102,20697931,20700473,20717304,20595032

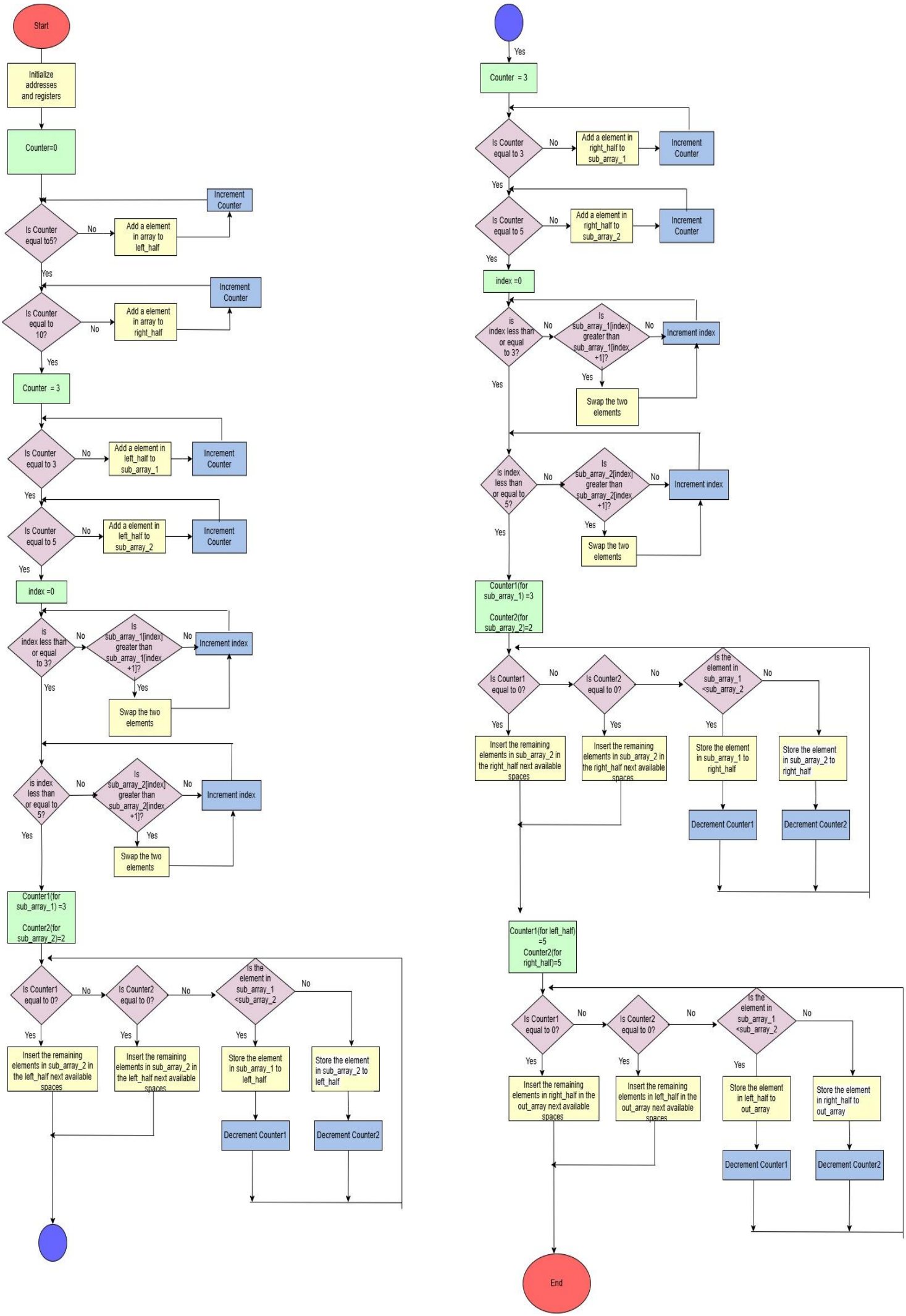
Firstly in the merge sort code the addresses and registers are initialized.The register sp holds the address of the stack pointer stored in stk_ptr, register r0 holds the address of the original array(init_array), register r1 holds the address of the left half of array(left_half), register r2 holds the address of the right half of array(right_half) , register r3 holds the address of the output array(out_array), registers r4 and r5 hold the address of sub arrays (sub_array_1 and sub_array_2) which will be used to store the further split arrays.

Then the array is split into two, a right half and a left half . The register r6 will be used as a counter and r7 is used to hold the midpoint(check split_init_array).By comparing r6 and r7 each time inside the loop , the left half of array is formed(check left_half_split). Similarly,by comparing r6 to the number 10 each inside the loop, the right half of the array is formed(check right_half_split).

After this, the left half of the array is then split further. Again r6 is used as a counter and r7 hold the midpoint. By comparing r6 and r7 each time inside the loop , the left half on the left half array is formed and is stored in one of the subarrays (check left_half_1). Similarly, by comparing r6 to the number 10 each inside the loop, the right half of the left half array is formed is stored in the other subarray (check right_half_1).Once done ,the left half is sorted and the right half is sorted(check Sort_Left_1 and Sort_Right_1).

Then the merging of the two subarrays of the left half array begins(check merge_sub_arrays). The registers r6 and r7 are now used to hold the midpoints of one of the subarrays respectively. R8 is initialized with the first element in sub_array_1 and r9 is initialized with the first element in sub_array_2. In main_loop_1 r6 is compared to 0 where if r6 is 0 jump to subarr_1_empty_1 occurs and r7 is also compared to 0 where if r7 is 0 jump to subarr_2_empty_1 occurs. Subarr_1_empty_1 and . subarr_2_empty_1 is used in case if one of the subarrays are exhausted , therefore the remaining elements in the other subarray are stored to the left_half. If r6 and r7 are still not 0 the elements stored in r8 and r9 are compared . If r8 is smaller than r9 the element in r8 is stored in the output array . If r8 is greater that r9 the element in r9 is stored in the output array .This continues until one of the subarrays exhaust. Now the left half of the array is completely sorted. This whole process is repeated for the right half of the array starting from splitting the right half further to sorting the subarrays to merging the subarrays.

After the right half is fully sorted the right half and left half of the array has to merge back to one whole array. The registers r6 and r7 hold the number of elements in left half and right half respectively. The register r8 and r9 are initialized to the first elements of left array and right array respectively. Firstly r6 is compared to 0 . If r6 is equal to 0 jump to left_empty occurs . r7 is also compared to 0 . If r7 is equal to 0 jump to right_empty occurs. Just like before left_empty and right_empty is used for example when left_half are exhausted the elements in right_half is stored to the out_array and vice versa. If r6 and r7 are not equivalent to 0 the elements in r8 and r9 are compared. If r8 is less than r9 , r8 is stored in out_array. Else r9 is stored in out_array. This repeats until either left_half or right_half exhausts.Once done the array has been fully sorted using the merge sort algorithm.


Start
Initialize addresses and registers
Counter=0
Is Counter equal to5?
No
Add a element in array to left_half
Increment Counter
Yes
Is Counter equal to 10?
No
Add a element in array to right_half
Increment Counter
Yes
Counter = 3
Is Counter equal to 3
No
Add a element in left_half to sub_array_1
Increment Counter
Yes
Is Counter equal to 5
No
Add a element in left_half to sub_array_2
Increment Counter
Yes
index =0
is index less than or equal to 3?
No
Is sub_array_1[index] greater than sub_array_1[index +1]?
No
Increment index
Yes
Swap the two elements
is index less than or equal to 5?
No
Is sub_array_2[index] greater than sub_array_2[index +1]?
No
Increment index
Yes
Swap the two elements
Counter1(for sub_array_1) =3
Counter2(for sub_array_2)=2
Is Counter1 equal to 0?
No
Is Counter2 equal to 0?
No
Is the element in sub_array_1 <sub_array_2
No
Yes
Insert the remaining elements in sub_array_2 in the left_half next available spaces
Insert the remaining elements in sub_array_2 in the left_half next available spaces
Store the element in sub_array_1 to left_half
Store the element in sub_array_2 to left_half
Decrement Counter1
Decrement Counter2
Yes
Counter = 3
Is Counter equal to 3
No
Add a element in right_half to sub_array_1
Increment Counter
Yes
Is Counter equal to 5
No
Add a element in right_half to sub_array_2
Increment Counter
Yes
index =0
is index less than or equal to 3?
No
Is sub_array_1[index] greater than sub_array_1[index +1]?
No
Increment index
Yes
Swap the two elements
is index less than or equal to 5?
No
Is sub_array_2[index] greater than sub_array_2[index +1]?
No
Increment index
Yes
Swap the two elements
Counter1(for sub_array_1) =3
Counter2(for sub_array_2)=2
Is Counter1 equal to 0?
No
Is Counter2 equal to 0?
No
Is the element in sub_array_1 <sub_array_2
No
Yes
Insert the remaining elements in sub_array_2 in the right_half next available spaces
Insert the remaining elements in sub_array_2 in the right_half next available spaces
Store the element in sub_array_1 to right_half
Store the element in sub_array_2 to right_half
Decrement Counter1
Decrement Counter2
Counter1(for left_half) =5
Counter2(for right_half)=5
Is Counter1 equal to 0?
No
Is Counter2 equal to 0?
No
Is the element in sub_array_1 <sub_array_2
No
Yes
Insert the remaining elements in right_half in the out_array next available spaces
Insert the remaining elements in left_half in the out_array next available spaces
Store the element in left_half to out_array
Store the element in right_half to out_array
Decrement Counter1
Decrement Counter2
End