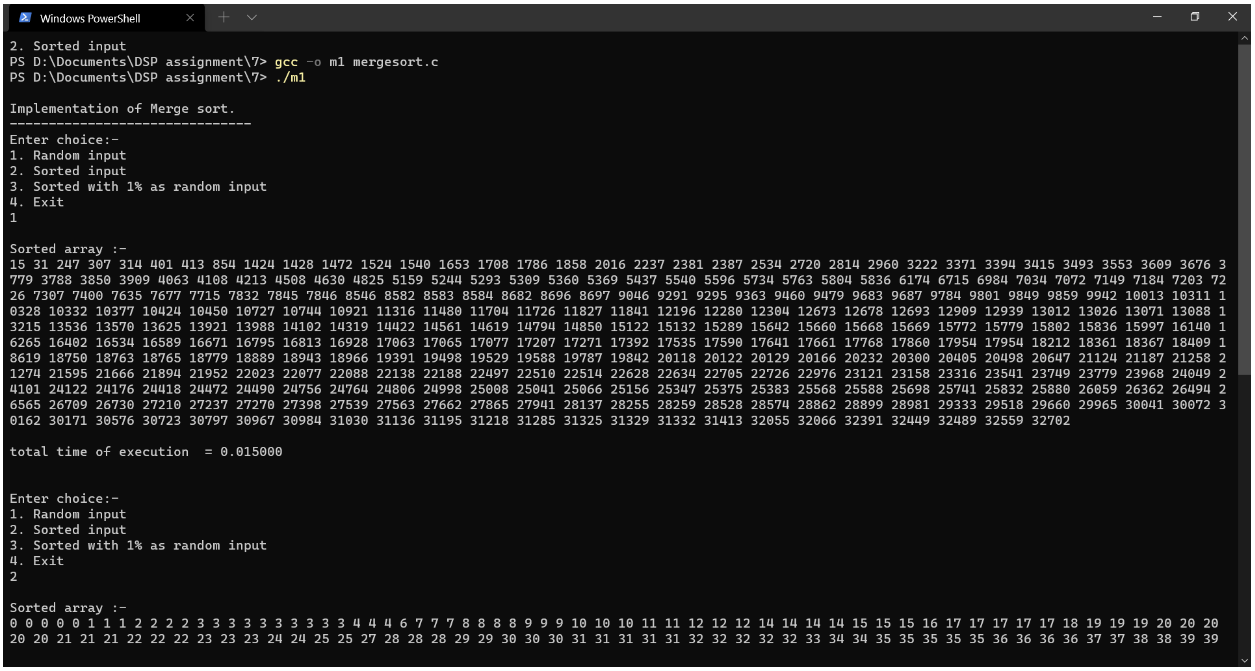
Implementation of Mergesort:-

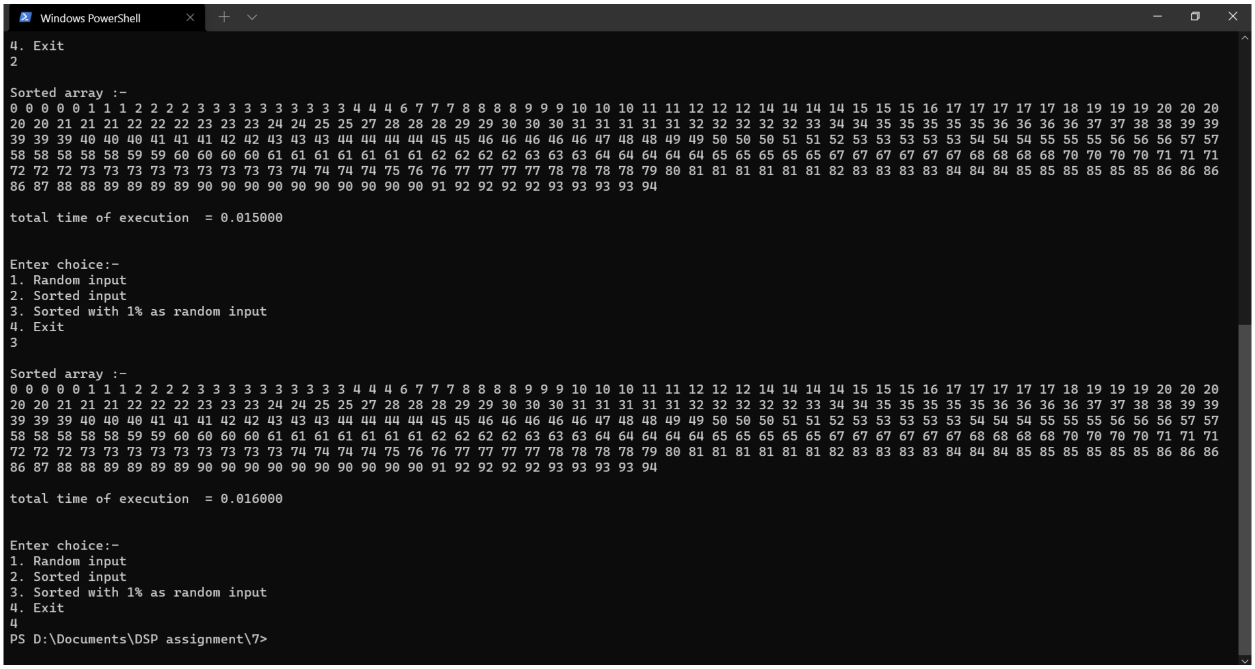
Following are the functions used in the program:-

1. merge(): This function is used to merge two subarrays. It takes two arrays as left and right, where it stores the entire array in two fragments. Then it sorts them and puts back the sorted order in the original array.
2. mergeSort(): This function is called recursively to sort the subarray.
3. main(): Program execution starts from the main function. It takes input from the files and accordingly sorts the elements in the array, as per the choice of the user. It also prints the time it takes to execute the quicksort() function.

OUTPUT:



```
2. Sorted input
PS D:\Documents\DSP assignment\7> gcc -o m1 mergesort.c
PS D:\Documents\DSP assignment\7> ./m1

Implementation of Merge sort.
-------------------------------
Enter choice:-
1. Random input
2. Sorted input
3. Sorted with 1% as random input
4. Exit
1

Sorted array :-
15 31 247 307 314 401 413 854 1424 1428 1472 1524 1540 1653 1708 1786 1858 2016 2237 2381 2387 2534 2720 2814 2960 3222 3371 3394 3415 3493 3553 3609 3676 3
779 3788 3850 3909 4063 4108 4213 4508 4630 4825 5159 5244 5293 5309 5360 5369 5437 5540 5596 5734 5763 5804 5836 6174 6715 6984 7034 7072 7149 7184 7203 72
26 7307 7400 7635 7677 7715 7832 7845 7846 8546 8582 8583 8584 8682 8696 8697 9046 9291 9295 9363 9460 9479 9683 9687 9784 9801 9849 9859 9942 10013 10311 1
0328 10332 10377 10424 10450 10727 10744 10921 11316 11480 11704 11726 11827 11841 12196 12280 12304 12673 12678 12693 12909 12939 13012 13026 13071 13088 1
3215 13536 13570 13625 13921 13988 14102 14319 14422 14561 14619 14794 14850 15122 15132 15289 15642 15660 15668 15669 15772 15779 15802 15836 15997 16140 1
6265 16402 16534 16589 16671 16795 16813 16928 17063 17065 17077 17207 17271 17392 17535 17590 17641 17661 17768 17860 17954 17954 18212 18361 18367 18409 1
8619 18750 18763 18765 18779 18889 18943 18966 19391 19498 19529 19588 19787 19842 20118 20122 20129 20166 20232 20300 20405 20498 20647 21124 21187 21258 2
1274 21595 21666 21894 21952 22023 22077 22088 22138 22188 22497 22510 22514 22628 22634 22705 22726 22976 23121 23158 23316 23541 23749 23779 23968 24049 2
4101 24122 24176 24418 24472 24490 24756 24764 24806 24998 25008 25041 25066 25156 25347 25375 25383 25568 25588 25698 25741 25832 25880 26059 26362 26494 2
6565 26709 26730 27210 27237 27270 27398 27539 27563 27662 27865 27941 28137 28255 28259 28528 28574 28862 28899 28981 29333 29518 29660 29965 30041 30072 3
0162 30171 30576 30723 30797 30967 30984 31030 31136 31195 31218 31285 31325 31329 31332 31413 32055 32066 32391 32449 32489 32559 32702

total time of execution  = 0.015000


Enter choice:-
1. Random input
2. Sorted input
3. Sorted with 1% as random input
4. Exit
2

Sorted array :-
0 0 0 0 0 1 1 1 2 2 2 2 3 3 3 3 3 3 3 3 3 3 4 4 4 6 7 7 7 8 8 8 8 9 9 9 10 10 10 11 11 12 12 12 14 14 14 14 15 15 15 16 17 17 17 17 17 18 19 19 19 20 20 20
20 20 21 21 21 22 22 22 23 23 23 24 24 25 25 27 28 28 28 29 29 30 30 30 31 31 31 31 31 32 32 32 32 32 33 34 34 35 35 35 35 35 36 36 36 36 37 37 38 38 39 39
```



```
4. Exit
2

Sorted array :-
0 0 0 0 0 1 1 1 2 2 2 2 3 3 3 3 3 3 3 3 3 3 4 4 4 6 7 7 7 8 8 8 8 9 9 9 10 10 10 11 11 12 12 12 14 14 14 14 15 15 15 16 17 17 17 17 17 18 19 19 19 20 20 20
20 20 21 21 21 22 22 22 23 23 23 24 24 25 25 27 28 28 28 29 29 30 30 30 31 31 31 31 31 32 32 32 32 32 33 34 34 35 35 35 35 35 36 36 36 36 37 37 38 38 39 39
39 39 39 40 40 40 41 41 41 42 42 43 43 43 44 44 44 44 45 45 46 46 46 46 46 47 48 48 49 49 50 50 50 51 51 52 53 53 53 53 53 54 54 54 55 55 55 56 56 56 57 57
58 58 58 58 58 59 59 60 60 60 60 61 61 61 61 61 61 61 62 62 62 62 63 63 63 64 64 64 64 64 65 65 65 65 65 67 67 67 67 67 67 68 68 68 68 70 70 70 70 71 71 71
72 72 72 73 73 73 73 73 73 73 73 73 74 74 74 74 75 76 76 77 77 77 77 78 78 78 78 79 80 81 81 81 81 81 81 82 83 83 83 83 84 84 84 85 85 85 85 85 85 86 86 86
86 87 88 88 89 89 89 89 90 90 90 90 90 90 90 90 90 90 91 92 92 92 92 93 93 93 93 94

total time of execution  = 0.015000


Enter choice:-
1. Random input
2. Sorted input
3. Sorted with 1% as random input
4. Exit
3

Sorted array :-
0 0 0 0 0 1 1 1 2 2 2 2 3 3 3 3 3 3 3 3 3 3 4 4 4 6 7 7 7 8 8 8 8 9 9 9 10 10 10 11 11 12 12 12 14 14 14 14 15 15 15 16 17 17 17 17 17 18 19 19 19 20 20 20
20 20 21 21 21 22 22 22 23 23 23 24 24 25 25 27 28 28 28 29 29 30 30 30 31 31 31 31 31 32 32 32 32 32 33 34 34 35 35 35 35 35 36 36 36 36 37 37 38 38 39 39
39 39 39 40 40 40 41 41 41 42 42 43 43 43 44 44 44 44 45 45 46 46 46 46 46 47 48 48 49 49 50 50 50 51 51 52 53 53 53 53 53 54 54 54 55 55 55 56 56 56 57 57
58 58 58 58 58 59 59 60 60 60 60 61 61 61 61 61 61 61 62 62 62 62 63 63 63 64 64 64 64 64 65 65 65 65 65 67 67 67 67 67 67 68 68 68 68 70 70 70 70 71 71 71
72 72 72 73 73 73 73 73 73 73 73 73 74 74 74 74 75 76 76 77 77 77 77 78 78 78 78 79 80 81 81 81 81 81 81 82 83 83 83 83 84 84 84 85 85 85 85 85 85 86 86 86
86 87 88 88 89 89 89 89 90 90 90 90 90 90 90 90 90 90 91 92 92 92 92 93 93 93 93 94

total time of execution  = 0.016000


Enter choice:-
1. Random input
2. Sorted input
3. Sorted with 1% as random input
4. Exit
4
PS D:\Documents\DSP assignment\7>
```