Note : The questions which have some modifications are listed below. The updations are highlighted using different colour.

4. Given an array ***A*** of ***n*** integers, sort the array in descending order using insertion sort. Print the number of comparisons, number of shifts (any change in position of an element is considered as a shift), and the elements of the array in each iteration. Assume that $1 \leq n \leq 100$.

Your program should implement the following functions:

main() **-** repeatedly reads a character ***'r', 's'*** or ***'d'*** and calls the sub-functions appropriately until character ***'t'*** is entered.

read(A, n) – read ***n*** integers and store it in an array ***A***.

sort(A, n) – perform insertion sort on array ***A*** of size ***n***, and print the number of comparisons and number of shifts in each iteration. Use display() function to print the array contents at the end of each iteration.

display(A, n) – display ***n*** elements of the array ***A***.

INPUT FORMAT:

First line contains a character from {***'r', 's', 'd','t'***} followed by zero or one integer.

• Character ***'r'*** is followed by an integer ***n***, the size of the input array. In this case, the second line gives the list of ***n*** integers separated by a space, to be read into the array.

• Character ***'s'*** is to sort the given array using insertion sort.

• Character ***'d'*** is followed by the integer ***n***, the size of the output array. In this case, the next line print the list of ***n*** integers separated by space.

• Character ***'t'*** terminates the program.

OUTPUT FORMAT:

For option ***'s'***, print the output for insertion sort at the end of each iteration. Display the number of comparisons and the number of shifts in the same line (space separated) followed by the elements of the array in the next line.

For option ***'d'***, print ***n*** elements of array separated by space in a new line.

Sample Input:

```
r 7
12 35 59 60 7 90 43
d
s
t
```

Sample Output:

12 35 59 60 7 90 43
1 1
35 12 59 60 7 90 43
2 3
59 35 12 60 7 90 43
3 4
60 59 35 12 7 90 43
1 0
60 59 35 12 7 90 43
5 6
90 60 59 35 12 7 43
4 4
90 60 59 43 35 12 7

6. Given an array ***A*** of ***n*** characters , write a recursive program to sort the array in ascending order using selection sort. Assume that ***1<= n <=100***.

Your program should implement the following functions:
main() - repeatedly reads a character ***'r', 's'*** or ***'d'*** and calls the sub-functions appropriately until character ***'t'*** is entered.
read(A, n) – read ***n*** characters and store it in an array ***A*** .
recursive_Selectionsort(A,n) – perform selection sort on array ***A*** using recursion. Use display() function to print the array contents after sorting.
display(A, n) – display n elements of the array

INPUT FORMAT:
First line contains a character from {***'r', 's', 'd' ,'t'***} followed by zero or one integer.
• Character ***'r'*** is followed by an integer ***n***, the size of the input array. In this case, the second line gives the list of ***n*** integers separated by a space, to be read into the array.
• Character ***'s'*** is to sort the given array using selection sort.
• Character ***'d'*** is followed by the integer ***n***, the size of the output array. In this case, the next line print the list of ***n*** integers separated by space.
• Character ***'t'*** terminates the program.

OUTPUT FORMAT:
For option ***'s'*** , print the output for selection sort.
For option ***'d'***, print ***n*** elements of array separated by space in a new line.

Sample Input:
r 7
I A M g o o d
d 7
s
t
Sample Output:
I A M g o o d
d g o o A I M

7. Given an array ***A*** of ***n*** integers, sort the array in ascending order using bubble sort, only non-negative integers are sorted in ascending order. Print the elements of the array after sorting. Assume that ***1<= n <=100***.
Your program should implement the following functions:
main() - repeatedly reads a character ***'r', 's'*** or ***'d'*** and calls the sub-functions appropriately until Character ***'t'*** is entered.
read(A, n) – read ***n*** integers and store it in an array***A***.
bubbleSort(A,n) – perform bubble sort on array ***A***, only non negative integers are sorted in ascending order. Use display() function to print the sorted array.
display(A, n) – display ***n*** elements of the array ***A***.

INPUT FORMAT:
First line contains a character from {***'r', 's', 'd' ,'t'***} followed by zero or one integer.
• Character ***'r'*** is followed by an integer ***n***, the size of the input array. In this case, the second line gives the list of n integers separated by a space, to be read into the array.
• Character ***'s'*** is to sort the given array using bubble sort.
• Character ***'d'*** is followed by the integer ***n***, the size of the output array. In this case, the next line print the list of ***n*** integers separated by space.
• Character ***'t'*** terminates the program.
OUTPUT FORMAT:
For option ***'s'***, print the output for bubble sort.
For option ***'d'***, print n elements of array separated by space in a new line.

Sample Input:
r 9
3 -1 5 -2 1 2 7 -3 4
d 9
s
t

Sample Output:
3 -1 5 -2 1 2 7 -3 4
1 -1 2 -2 3 4 5 -3 7

8. Given an array ***A*** of ***n*** integers that stores account numbers of all employees in an organization (account numbers have equal number of digits), sort the array in ascending order using radix sort.
Print the elements of the array after sorting. Assume that ***1<= n <=100***.
Your program should implement the following functions:
main() - repeatedly reads a character ***'r', 's'*** or ***'d'*** and calls the sub-functions appropriately until Character ***'t'*** is entered.
read(A, n) – read ***n*** integers and store it in an array ***A***.
radixSort(A,n) – perform radix sort on array ***A***. Use display() function to print the sorted array.
display(A, n) – display ***n*** elements of the array ***A***.

INPUT FORMAT:
First line contains a character from {***'r', 's', 'd' ,'t'***} followed by zero or one integer.
• Character ***'r'*** is followed by an integer ***n***, the size of the input array. In this case, the second line gives the list of ***n*** integers separated by a space, to be read into the array.
• Character ***'s'*** is to sort the given array using radix sort.
• Character ***'d'*** is followed by the integer ***n***, the size of the output array. In this case, the next line print the list of ***n*** integers separated by space.
• Character ***'t'*** terminates the program.

OUTPUT FORMAT:
For option ***'s'***, print the output for radix sort.
For option ***'d'***, print n elements of array separated by space in a new line.

Sample Input:
r 6
582 675 591 189 900 770
d 6
s
t

Sample Output:
582 675 591 189 900 770
189 582 591 675 770 900