Assignment - 05

Name :- J. Lokesh

Regno :- 192311216.

SUB :- D.S

SUB CoDe :- CSA0389.

Write the algorithm for insertion Sort and Sort the following Sequence: 3, 1, 4, 1, 5, 9, 2, 6, 5

ii) Explain the procedure for merge Sort and perform the merge Sort for the following inputs. Also, Show the result for each Step for iteration 64, 8, 216, 512, 27, 72, 9, 0, 1, 343, 125.

Sol: Algorithm for Insertion:-

1) Begin with the Second element in the List.
2) Compare the Current Element to the previous elements.
3) Shift all larger element one position to the right.
4) Insert the Current element into its Correct position.
5) Repeat Steps 2-4 for each element in the List until the entire List is Sorted.

Sorting the Sequence:-

Sequence: 3, 1, 4, 1, 5, 9, 2, 6, 5

3	1	4	1	5	9	2	6	5

Compare 3 & 1, 3 > 1
Swap 3, 1

1	3	4	1	5	9	2	6	5

1	3	1	4	5	9	2	6	5

1	1	3	4	5	9	2	6	5

1	1	3	4	5	2	9	6	5

1	1	3	4	2	5	9	6	5

1	1	3	2	4	5	9	6	5

1	1	2	3	4	5	9	6	5

1	1	2	3	4	5	6	9	5

1	1	2	3	4	5	6	5	9

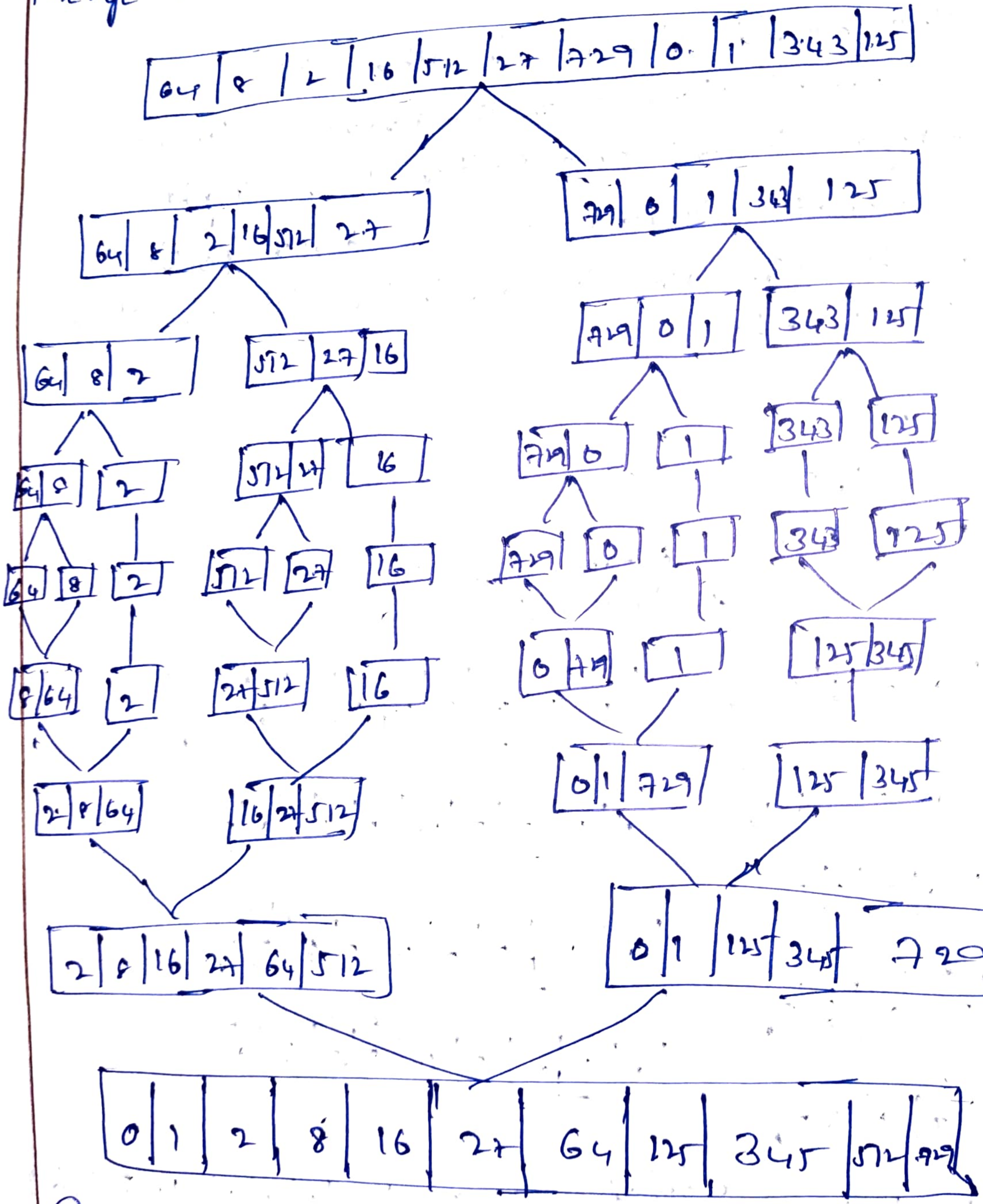
1	1	2	3	4	5	5	6	9

Sorted Sequence:- 1, 1, 2, 3, 4,

Merge Sort Procedure:-

* Split the list into halves until each sublist has one element.
* Combine the sublists to produce new sorted sublists until there is one sorted list.

Merge sort with 64, 8, 2 16, 512, 27, 729, 0, 1, 343, 125.



Sorted list:- 0, 1, 8, 27, 64, 125, 216, 343, 512, 729.

Drao the Conpcept of quick Sort

Step1:- Choose the highest index value has pivot

Step2:- take two variables to point left and right of the list excluding pivot

Step3:- left points to the low index using elements your own

Sol:- Algorithm:-

*Select the element at the highest index as the pivot

* Set 'left' to the low index and right to the high index-1

* move 'left' rightwards and 'right' leftwards until left is greater than or equal to 'right', swapping elements as the needed.

* Swap the pivot with the element at the 'left' pointer position.

* Return the index of the pivot element.

Program:-

```
#include <stdio.h>
int main() {
 int arr[ ]={64,8,216,512,27,729,0}
 int n= sizeof(arr)/sizeof(arr[0]);
 int low=0, high=n-1;
while (low < high){
 int pivot= arr[high];
 int left=low;
 int right =high-1;
while(left<= right){
 while(left<=right && arr[left]<
                            pivot){
   left++;
 }
     while(right>=low && arr[ri
                    >pivot){
            right--;
     }
 if (left <right) {
 int temp= arr[left
   arr[left] =arr [right];
   arr [right]= temp;
     left++;
    right--;
  }
  }
int temp = arr[left];
arr[left] = arr[high];
arr[high] = temp;
high = left - 1;
if (high < low){
  low = left+1;
  high = n-1; } }
Printf ("Sorted array: ");
for (int i=0; i<n; i++){
   Printf ("%d", arr[i]);
}
 Printf ("\n");
   return 0;
}
```

utput :-

Sorted array:- 0, 1, 8, 27, 64, 125, 343, 512, 729,,