Name : K.vamsi krishna yadav.

Reg no : 192324165.

Course : Data structure.

Course code: CSA0389.

Submission : 21/08/2024.

Assignment no: 03.

write the algorithm for insertion sort and sort the following sequence: 3, 1, 4, 1, 5, 9, 2, 6, 5.

Sol Algorithm for insertion.

i) Begin with the second element in the first.

ii) compare the current element to the Previous elements.

iii) shift all Larger elements one Position the right.

iv) insert the current element into its correct Position.

v) RePeat steps for each elements.

Sorting the 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

compare 4 & 1, 4 > 1
swap 4, 1

1	3	1	4	5	9	2	6	5

compare 3 & 1, 3 > 1
swap 3, 1.

1	1	3	4	5	9	2	6	5

compare 9 & 2 9 > 2
swap 9, 2

1	1	3	4	5	2	9	6	5

compare 5 & 2 5 > 2
swap 5, 2.

1	1	3	4	2	5	9	6	5.

compare 4 & 2 com 4>2
swap 4,2

1	1	3	2	4	5	9	6	5

compare 2 & 3 3>2
swap 3,2

1	1	2	3	4	5	9	6	5

compare 6 & 5 , 6>5
swap 6,5

1	1	2	3	4	5	9	5	6

compa 9 & 5 9>5
swap 9,5

1	1	2	3	4	5	5	9	6

compare 9 & 6 9>6
swap 6,9

1	1	2	3	4	5	5	6	9

sorted.

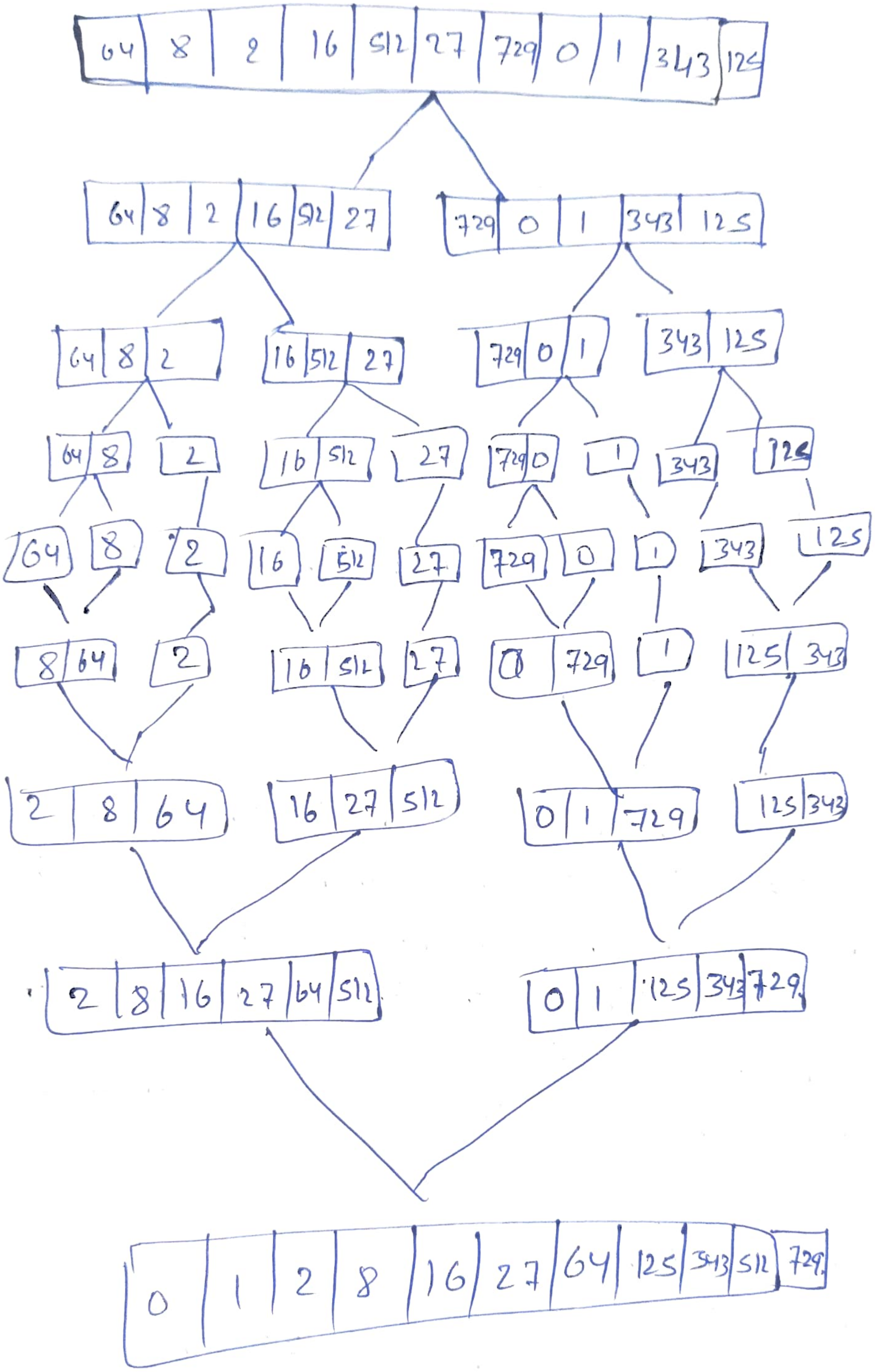
merge sort Procedure:

* split the list into halves. untill each sublist has one element.

* combine the sublists untill there is one sorted list.

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

2



Sorted list = 0, 1, 2, 8, 16, 27, 64, 125, 343,
512, 729.

(2) Draw the concept of quick sort.

step1: choose the highest index value has pivot.

step2: rake 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.

*more 'Left" right words' and 'right' Leftwords. until Left is greater than or equal to 'right', swapping elements as the needed.

* swap the pivot with the elements at the 'Left' pointer position.

* Return the index Of the pivot element.

Program:

```
#Include <stdio.h>
 int main() {
 int arr[] = {64, 8, 216, 5, 2, 27, 729, 0, 1};
 int n=  size of (arr)/ size of (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 (right] > Pivot )}

          ~~ring~~ 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;
 }
```

OutPut :

```
storted array: 0,1,8,27,64,125,343,512,
               729.
```