IARE
INSTITUTE OF
AERONAUTICAL ENGINEERING
(An Autonomous Institute affiliated to JNTUH, Hyderabad)
Dundigal, Hyderabad - 500 043

LABORATORY WORK BOOK

Name of the Student : Racherla Santhosh

Class IT-B Semester 03

Course Code : AGSD11 Course Name : DS laboratory

Name of the Course Faculty Ms. K. Laxminarayanamma Faculty ID : IARE10033

Exercise Number : 04 Week Number : 04 Date : 30/09/2024

Roll Number									
2	3	9	5	1	A	1	2	C	3

S. No.	Exercise Number	EXERCISE NAME	MARKS AWARDED					
			Aim/ Preparation	Algorithm / Procedure Performance in the Lab	Source Code Calculations and Graphs	Program Execution Results and Error Analysis	Viva - Voce	Total
			4	4	4	4	4	20
1	4.1	Quick Sort						
2	4.2	Merge Sort						
3	4.3	Heap Sort						
4	4.4	Radix Sort						
5	4.5	Shell Sort						
6			4	4	4	4	4	20
7								
8								
9								
10								
11								
12								

Santhosh

Signature of the Student

[illegible]

Signature of the Faculty

4. Divide And Conquer.

4.1 Quick Sort :-

AIM :- Write a Program to Sort the given list of elements using Quick Sort.

PROGRAM :-

```
Public class quickSort {
 Public static void quickSort (int[] arr, int low,
                                  int high) {
    if (low < high) {
       int  pi = Partition (arr, low, high);
       quickSort (arr, low, pi - 1);
       quickSort (arr, pi+1, high);
    }
 }
 Public static int Partition (int[] arr, int low,
                                  int high) {
    int Pivot = arr[low];
    int i = low + 1 ;
    for (int j = low + 1 ;  j <= high ; j++ ) {
      if (arr [j] < pivot)  {
        Swap (arr, i, j ) ;
        i++ ;
      }
   }
  Swap (arr, low, i - 1) ;
  return  i - 1 ;
 }
 Public  Static  void   Swap ( int[] arr, int i, int j) {
    int  temp  =  arr [i] ;
    arr [i]  =  arr [j] ;
    arr [j]  =  temp ;
 }
 Public  Static  void   printArray (int [] arr ) {
   for (int  i = 0 ;  i < arr. length ;  i++ ) {
     System. out. print  ( arr [i] + " ") ;
   }
```

ROLL NUMBER :

```
    System. out. println() ;
  }
 Public  Static void  main ( String [] args) {
     int [] arr = { 7, 6, 10, 5, 9, 2, 1, 15, 7 } ;
     int n = arr. length;
     System. out. println (" Original array :") ;
     Print Array (arr) ;
     quick Sort (arr, 0, n-1) ;
     System. out. println (" Sorted array :") ;
     Print Array (arr) ;
  }
}
```

RESULT : -

INPUT : arr = [7, 6, 10, 5, 9, 2, 15, 7]

OUTPUT : arr = [2, 5, 6, 7, 7, 9, 10, 15]

4.2 Merge Sort :-

AIM :- Write a Program to Sort the given list of elements using Merge Sort.

PROGRAM :-

```
class MergeSort {
  void merge (int arr[], int l, int m, int r)
  {
    int n1 = m-l+1;
    int n2 = r-m;
    int L[] = new int[n1];
    int R[] = new int[n2];
    for (int i=0; i<n1; ++i)
       L[i] = arr[l+i];
    for (int j=0; j<n2; ++j)
       R[j] = arr[m+l+j];
    int i=0, j=0;
    int K=1;
    while (i<n1 && j<n2) {
```

ROLL NUMBER :

```
    if ( L[i]  <=  R[j])  {
        arr [K]  =  L[i] ;
        i++;
    }
    else {
        arr [K]  =  R[j] ;
        j++ ;
    }
    K++;
  }
  while (i< n1) {
    arr [K] = L[i] ;
    i++;
    K++;
  }
  while ( j <n2) {
    arr [K] = R[j] ;
    j++;
    k++;
  }
}
```

ROLL NUMBER :

```
void Sort (int arr[], int l, int r)
{
  if (l<r) {
     int m = (l+r)/2;
     Sort (arr, l, m);
     Sort (arr, m+1, r);
     merge (arr, l, m, r);
  }
}
Static void printArray (int arr[])
{
  int n = arr.length;
  for (int i=0; i<n; ++i)
     System.out.print (arr[i] + " ");
  System.out.println ();
}
Public Static void main (String args[])
{
  int arr[] = { 12, 11, 13, 5, 6, 7 };
  System.out.println (" Given Array ");
```

ROLL NUMBER :

```
        Print Array (arr);
        Merge Sort ob = new MergeSort ();
        ob. Sort (arr, 0, arr. length - 1);
        System. out. println ("\n Sorted array");
        Print Array (arr);
    }
}
```

RESULT :-

INPUT : arr = [12, 11, 13, 5, 6, 7]

OUTPUT : arr = [5, 6, 7, 11, 12, 13]

4.3 Heap Sort :-

AIM :- Write a Program to Sort the given list of elements using Heap Sort.

PROGRAM :-

```
Public class HeapSort {
    Public void Sort (int arr[]) {
```

ROLL NUMBER :

```
int  N = arr.length;
for (int i=N/2 - 1; i>=0; i--)
    heapify (arr, N, i);
for (int i=N-1; i>0; i--) {
    int temp = arr[0];
    arr[0] = arr[i];
    arr[i] = temp;
    heapify (arr, i, 0);
  }
}
void heapify (int arr[], int N, int i)
{
  int largest = i;
  int l = 2*i + 1;
  int r = 2*i + 2;
  if (l<N && arr[l] > arr[largest])
   largest = l;
  if (r<N && arr[r] > arr[largest])
```

```
largest = r ;
if ( largest ! = i ) {
  int swap = arr [i] ;
  arr [i] = arr [ largest ] ;
  arr [largest] = swap ;
  heapify (arr , N, largest) ;
 }
}
static void printArray (int arr[])
{ int N = arr. length ;
  for ( int i = 0 ; i < N ; ++i )
     System. out. print (arr [i] + " ") ;
  System. out. println () ;
}
Public static void main( String args [] )
{
  int arr []  =  { 12, 11, 13, 5, 6, 7 } ;
  int N = arr. length ;
```

ROLL NUMBER :

```
        HeapSort ob = new HeapSort();
        ob. Sort (arr);
        System. out. println (" Sorted array is");
        PrintArray (arr);
    }
}
```

RESULT :-

INPUT : arr = [12, 11, 13, 5, 6, 7]

OUTPUT : Sorted array is 5, 6, 7, 11, 12, 13 .

4.4 Radix Sort :-

AIM :- Write a Program to Sort the given list of elements using Radix Sort .

PROGRAM :-

```
import. java. io. * ;
import java. util. * ;
class Radix {
  State int getMax (int arr[], int n)
  {
```

```
int mx = arr [0] ;
for (int i= 1; i< n ; i++)
 if (arr [i] > mx )
    mx = arr [i] ;
return mx ;
}
Static void  count Sort (int arr [], int n, int exp)
{ int output [] = new int[n] ;
 int i ;
 int Count [] = new int [10] ;
Arrays .fill ( count, 0) ;
for ( i = 0 ; i<n ; i++)
  count [ (arr [i] / exp ) % 10 ] ++ ;
for ( i = 1 ; i < 10 ; i++)
  count [i]  + = count [i-1] ;
for ( i=n - 1 ; i>=0 ; i--) {
 output [count [(arr [i] / exp) % 10] - 1] = arr[i];
```

ROLL NUMBER :

```
    count [ (arr [i] / exp ) % 10 ] -- ;
  }
  for (i = 0 ; i < n ; i++)
     arr [i]  =  output [i] ;
 }
 Static void  radix Sort (int arr [], int n)
 {
   int m = get Max (arr, n) ;
   for (int exp = 1 ; m / exp > 0 ; exp * = 10 )
      countSort (arr, n, exp) ;
 }
 Static void print ( int arr [], int n)
 {
  for (int i = 0 ; i < n ; i + +)
     System. out. print (arr [i] + " ") ;
 }
Public Static void main ( String [] args )
{
  int arr [] = { 170, 45, 75, 90, 802, 24, 2, 66 } ;
  int n = arr. length ;
  radixsort (arr, n) ;
```

ROLL NUMBER :

```
    Print (arr, n);
  }
}
```

RESULT :-

INPUT : arr = [170, 45, 75, 90, 802, 24, 2, 66]

OUTPUT : arr = [2, 24, 45, 66, 75, 90, 170, 802]

4.5 Shell Sort :-

AIM :- Write a Program to Sort the given list of elements using Shell Sort.

PROGRAM :-

```
class ShellSort
{
  Static void printArray(int arr[])
  {
    int n = arr.length;
    for (int i=0; i<n; ++i)
        System.out.print (arr [i] + " ");
    System.out.println();
  }
```

```
int Sort (int arr[])
{
  int n = arr. length ;
  for ( int gap = n/2 ; gap >0; gap /= 2)
  {
    for (int i= gap ; i<n ; i += 1)
    {
      int temp = arr[i] ;
      int j ;
      for ( j = i ; j >= gap && arr [j - gap] >
            temp ; j = gap)
        arr [j] = arr [j - gap] ;
        arr [j] = temp ;
    }
  }
  return 0;
}

Public Static void main (String args[])
{
  int arr[] = { 12, 34, 54, 2, 3 } ;
```

```
    System.out.println(" Array before Sorting");
    Print Array (arr);
    Shell Sort ob = new Shell Sort();
    ob • Sort (arr);
    System.out.println(" Array after Sorting");
    Print Array (arr);
  }
}
```

RESULT :-

INPUT : arr = [12, 34, 54, 2, 3]

OUTPUT : arr = [2, 3, 12, 34, 54]

[illegible]

VIVA VOCE :-

1) What is Sorting Algorithm ?

A) A Sorting Algorithm is used to rearrange a given array (or) list elements using to Compare Operator on the elements. the Comparison Operator is used to decide the new Order of element in the respective Data Structure.

2) Explain about Merge Sort ?

A) Merge Sort is a general - Purpose Sorting technique purely based on Divide & Conquer Approach. In the Divide & Conquer Technique, The elements are divided into smaller parts of lists. Then the appropriate function is applied to each half of the main input list. Further, the halves are Merged together to get the Result.

3) How does Quick Sort Work ?

A) Quick Sort is Divide & Conquer Sorting Algorithm. It chooses a pivot element & rearrange the Array so that elements are smaller than the Pivot are on the left and elements greater are on the right.

4) What is Radix Sort ?

A) Radix Sort is the Sorting Technique which is based on the Value. In this method, The elements Can be Sorted by making use of either of the two methods LSD (Least Significant Digit) MSD (Most Significant Digit).

[illegible]