***NAME : Himanshu Dixit***

***ENROLL NO. : 21103262***

***BATCH : B11***

***Data Structure [15B11CI311]***

***Tutorial Sheet***

***Week 5***

***Q.1.*** *Given a test case Arr[] = 12 14 19 23 25 29 40 43 86 99, write the steps of bubble,  selection and insertion sort. Can these be optimized further? If yes, how? Also compare these  sorting techniques based on*

*∙ Number of comparisons*

*∙ Number of swaps*

*∙ Extra Space Requirement.*

*∙ In Place sorting*

***Solution :***

*Bubble Sort:*

*Procedure bubbleSort(int arr[], int n)*

*for (int i=0; i<n-1; i++)*

*for (int j=0; j<n-i-1; j++)*

*if (arr[j]>arr[j+1])*

*swap(arr[j], arr[j+1]);*

*Selection Sort:*

*Procedure insertionSort(int arr[], int n)*

*for (int i=1;i<n;i++)*

*int ind=i;*

*int temp=arr[i];*

*for (int j=i+1;i<n;i++)*

*if (arr[ind]>arr[j])*

*ind=j;*

*swap(arr[i], arr[ind]);*

*Insertion Sort:*

*Procedure insertionSort(int arr[], int n)*

*for (int i=1; i<n;i++)*

*int j=i-1,temp=arr[i];*

*while (arr[j]>temp && j>=0)*

*a[j+1]=a[j];*

*j=j-1;*

*arr[j+1]=temp;*

*Bubble sort can be optimised further by checking if no swap takes place while checking by introducing a flag variable.*

*Comparing all the sorting techniques using given example(assuming optimisation in Bubble Sort) on the basis of :*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Bubble Sort** | | **Selection Sort** | | **Insertion Sort** |
| **No. of comparisons** | n-1 | n(n-1)/2 | n(n-1)/2 | |
| **No. of swaps** | 0 | n-1 | 0 | |
| **Extra Space requirement** | No | No | No | |
| **In place Sorting** | Yes | Yes | Yes | |

***Q.2.*** *Apply quicksort on Arr[] given in Q.1.*

*∙ Taking pivot always as the first element*

*∙ Taking pivot always as the last element*

*∙ Taking pivot always as the middle element*

*Write number of comparisons happened in each case.*

***Solution :***

*i) No. of comparisons = n(n-1)/2*

*ii) No. of comparisons = n(n-1)/2*

*iii) No. of comparisons = n(n-1)/2*

***Q.3.*** *Given an integer array nums, move all the even integers at the beginning of the array  followed by all the odd integers. Return any array that satisfies this condition.*

*Input: nums = [3, 1, 2, 4]*

*Possible Output: [2,4,3,1]/ [4,2,3,1]/[2,4,1,3],/[4,2,1,3].*

***Solution :***

*Procedure evenFirst(int nums[], int n)*

*for (int i=1;i<n;i++)*

*if (arr[i]%2==0)*

*int j=i-1;*

*int temp=arr[i];*

*while (arr[j]%2==1 && j>=0)*

*a[j+1]=a[j];*

*j=j-1;*

*arr[j+1]=temp;*

***Q.4.*** *Given an array of meeting time intervals consisting of start and end  times[[s1,e1],[s2,e2],...](si< ei), determine if a person could attend all meetings.*

*Input: Input:*

*[[0,30],[5,10],[15,20]] [[7, 10],[2,4]]*

*Output : false Output : true*

***Solution :***

*Procedure possible(int start[], int end[], int n)*

*for (int i=1;i<n;i++)*

*int j=i-1;*

*while (j>=0)*

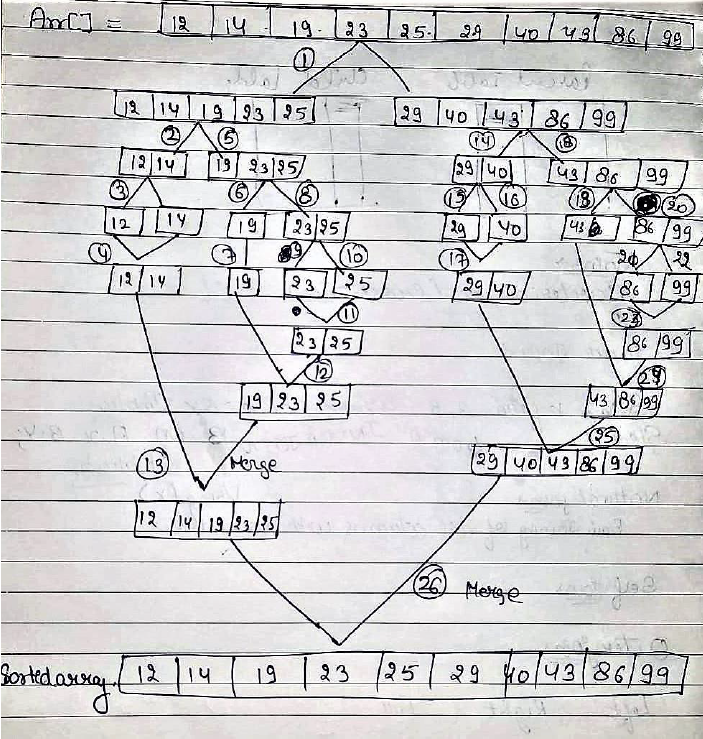
*if (start[i]<end && end[i]>start[j])*

*return false;*

*return true;*

***Q.5.*** *Apply merge sort and write all the steps on Arr[] given in Q.1.*

***Solution :***

**