merse aluce Heap 23145 32415 43251 forcturn=1 turnentuntt) fur ci=0 li< 2 if carreil sarreill public static void Sort(int[] arr) { // TODO Auto-generated method stub '**for** (**int** turn = 1; turn < arr.length; turn++) for (int i = 0; $i \not \leq 4$, i++) { **if**(arr[i]>arr[i+1]) { int temp=arr[i]; arr[i]=arr[i+1]; arr[i+1]=temp; 1=3 public static void Sort(int[] arr) { // TODO Auto-generated method stub for (int turn = 1; turn < arr.length; turn++) {</pre> for (int i = 0; i < arr.length - turn; i++) {</pre> if (arr[i] > arr[i + 1]) { int temp = arr[i]; arr[i] = arr[i + 1];arr[i + 1] = temp;123 5 4 12384 12381:5 int[] <u>arr</u> = { 4, -1, 5, 3, 1, 2 }; mini= idx=2 For Ci = idx+1 i < an.lon i++) &
if carrei) < an cmini) 12

mini=i
3 i dx = 1 idx= 4 idx = 5public static void Sort(int[] arr) { for (int i = 0; i < arr.length; i++) {</pre> jdx =1 int idx = Minimum_Index(arr, i); rint temp = arr[idx]; (idxii) idx=4 arr[idx] = arr[i]; arr[i] = temp;-112479 itom=arrcej >2 j= i-1 arrce)>iton 2 arrcjH)=arcj) 2 arrcj)=jten [2,-1,3,5] j=0 { 4, 1, 3, 4, 5, 9, 6, 2 }; public static void Insert Last Element(int[] arr, int i) {// i->
plast index int item = arr[i];
 int j = i - 1;) int j = i - 1;
while (j >= 0 && arr[j] > item) { arr[j + 1] = arr[j];2 244569 // j+1 correct index; Co-1) 2 ans= 700 forci=o ican-lon_i+1/c som+=arci) and= max(ans,form) 23-79 - 21 23-74-121 23-74-18+9 ans=2 public static int Maximum_Sum(int[] arr) { int ans = Integer.MIN_VALUE; int sum = 0;for (int i = 0; i < arr.length; i++) {</pre> sum += arr[i]; Sum=2 +7 ans = Math.max(ans, sum);13 if c sum 2018 1=2 Sm= 3-7 SUm =0 - fu-(2) -1 gr= 4-1 = 11-13=2 Given an array nums of size n, return the majority element. The majority element is the element that appears more than [n / 2] times. You may assume that the majority element always exists in the array. C=27377 if carroi)==ell 6 = au (0)

Lec-8

bupble Serv-