	Page No
	Jutorial - 3
1.	int linearcearch (int cour(), int n, int ky) {
	for (int i=0; i <n; i+t){<="" th=""></n;>
	if (aur [i] == key)
	return i;
	3
	eleturn -1;
	3
2.	iterative insertion sort
	wold insertionsort (int aver[3, int n){
	int i, j, t = 0;
	for (i=1', i <n; i++){<="" th=""></n;>
	t = aver [i];
	j = î-1;
	while (1> = 0 88 + < our (1)){
	are (j+1) = are (j);
	1'
	3
	alex (j+1) = t;
-	3
	3
	recursive insertion sort.
	noid insertionsout (int aver (), int n){
	if (n<=1)
-	gretwen;
	insectionsout (aux, n-1);

	rage No.
	last = aver [n-1];
	j=n-2;
	while (ix>=0 88 aver(j)>last)
	our [j+1] = our [j];
	g;
	3
	auer [3+1] = last;
	J
	Insertion sout is called online sorting
	because it does not need to know
	anything about what values it will
	sout and the information is requisted
	While the algorithm is surmaing.
3.00	bubble sout-
(4)	
	Time complexity—Best case = 0(n2) Worst case = 0(n2)
	space complexity = O(1)
(ii)	Selection sout -
	Space complexity = 0(1) Selection sout - Time complexity - Best case - 0(n2)
	Woult case - O(n2)
	space complexity = O(1)
(iii)	Merge sort -
	Time complemy - best case - o (neogn)
	Space complenity = O(1) Merge sout - Time complenity - best case - o(nlogn) Worst case - o(nlogn)
	space Complexity - O(n)

Date.

			Date. — Page No.	
(v)	Insertion	sout -		
	Time con Space con	plenity - Be plenity - Ol	st Can - Ol	en), woust
V)	Quick Se	Dect -		
	Time con	relevity - P	best case -	- O(nloan)
		1	lout care	- O(n2)
	Space a	pert- replenity-fr omplenity-	- O(n)	
w'	Heap s	out - mplenity - mplenity -		
	Time co	suplenity -	Best Case -	O(nlogn)
		- V	loust case	- O(neogn)
	space co	implenity -	-0(1)	0 /
Me	souting	inplace	Stable	Online
	Selection			
	Mution			
	Merge			
	Reap			
	Bubble			
5	Heratine	Dimary son	0.00	
	int bing	vypearch (inst and Fo	0
_		Joseph Control	land ford	ent e, but it,
_	ne	hile (l<=91)	5	ky) &
		1		

んべん

Page No.
int m = (l+2)/2;
if (aur [m]== kly)
seetween m;
if (aur [m] < key) T.C.
l= m+1; Best Care-O1)
else Arg. (ax = O(logn)
9=m-1; Well Car =
3 O(logn)
requen -1;
3
recursive binary search
ent binauxeauch (int aus [], int l, int a, int
4 (47-2) 8
int m = (l+91)/2;
if (our (m) = 2 key)
seteen m;
elle if (aur [m] > key)
elle if (aver [m] > key) giehren binarystatch (aver, l, mid-] elle kieg);
elle (cig);
netven binarysearch (aug, midtl, 9, key);
)
T.
report -1; T.C.
Best Case = O(1)
 Mg. Case = O (logn) Worse Case = O (logn)
would case 2 (wg)

Date. -

		Page No
		Linear Search T.C.
		Best Case: O(1)
		Avg. Case: O(n) Worst lase: O(n)
_		Worst (are 10(n)
_		
)_	61	Recurrence delation for binary
_		recuerère search - T(n) = T(n/g) +1
		(1) (1/2)
	4	A[i] + A[i] = K
-		
	8.	Quick Sout is the fastest general-
		The war procural
		signations quecksout is the method
		of church is stability is impose tout
		merge sort
		might be leest.
_	9,	Inversion count 104 111
_		indicates - how you (or close)
,—		indicates - how fair (or close) the array is from being sorded. It
_		the acreay is already souted, then
,		the inversion count is o, but if the
		7

Date. -Page No. array is sorted in the reverse order, the inversion court is the maximum. aver [] = 27,21,31,8,10,1,20,6,4,5} # include < bits / stdc + +. 4 > using namespace std; int merge-soit (int area (), int temp (), int left, int right); int merge (int our [], int temp() int left, int mid, int right; int mergesort (int aur [], int auray, size) !
int temp [auray size];
eltern merge-sout (aur, temp, o, aceray-size-1); int merge-sort (int arce[], int temp[], int left, int right) & int mid, in-count =0; if (right > left) { mid = inght left + (right-left)/2; Inv-count + = merge_sort (aver, temp, left, mid); in count + = merge sort (accer, temp signt);
in count + = merge (accer, temp mid+1, night);
left, mid+1, night); section in count;

Page No.

int merge (int arr [], int temp[], int left, int mid, int right) & int i, j, k, inv-count =0; i = left; j= med; K = left; whèle ((iz=mid-1) & (j<= right)) { if (aur (i) <= aur[j]) temp [1<+] = ava [i++]; temp[[t++] = ava[j++]; in-count = in-count +
(mid-i); while (i<= mid -1) temp [k++] = arr[i++]; while (j <= right) temp [k++] = aver [j++]; for (i= left; jc= night; i++) geteen in-count; int main (){ int aur [] = 27, 21, 51, 8, 10, 1, 20, 6, 4, 5}; int n = size of (arr) (size of (arr [0]);

	Date. ————————————————————————————————————
	int ans = merges out (ascr, n);
	cout « " no g'inversion are " «
	eleheren O';
	4
10 ,	The worst case the complexity of
	quick sout is O(n2). The worst care
	occuers when the picked pirot is
	alway in entereme & smallest or largest
	element. This happen when input
	avay is souted or reverse souted
	and either first or last element
	is picked as perot
	The feest case of quick sout is when
	we will select pivot as a mean
	dement.
11.	Recurrence relation of:
a)	Menge Sout = T(n) = 2T (n/2)+n
b)	Menge Sout => T(n) = 2T (n/2)+n Duick Sout => T(n) = 2T (n/2)+n
-	Merge sout is more efficient 8 nous
	faster than quick sout in case of
	larger array size or datalets.
1	In all to all the second of th
	is o(n2) whereas o(neogn) for
	is $O(n^2)$ whereas $O(n \log n)$ for neige sout.

	Page No.
19	Stable selection Spert
10,	
	void stableselectionspert (int aver [],
	ℓ int n) ℓ
	for (int i=0; i <n-1; i++){<="" th=""></n-1;>
	Ent mén = 1;
	for (int j=i+1; j <n; j++){<="" th=""></n;>
	if (are [min] > are [i])
	min = j;
	5
	int key - arr (min);
	while (min>i) &
	aver [min] = a [min-1];
	nin;
	J. Per Co.
	all [i] = (cey;
	1
	int main () &
	int aser [] = \$4,5,3,2,4,13;
	int n = size of (avoi) / size of (aur [o]);
	Gtableselectionsout (aver, n);
	jou (int i=0; l <n', i++).<="" th=""></n',>
	cout « aur [i] « " ",
	cout « endl;
	repen 0',
	3

Date

	Page No
13.	The easiest way to do this is to we
	sorting. We devide our sound
	file into temperary temporary files of
	file into temperory temporary files q Size equal to the size of the RAM & first sout these files.
	& jour soit suise files,
0	Gentlynal Sperting: If the input datain
	such that is count adjusted in the
	memory entirely at once it needs
	to be solved in a hand disk, floopy
	to be sorted in a hard disk, floppy disk or any other storage device. This is called external 80sting.
	is called external sorting.
. •	Internal socitions: If the input data
	I've such that it can calleted in the
	main memony at once it is called internal souting.
	internal souting.