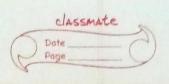
Tutorial-3 Search (arr, n,x)
if (arr (n-1)z=x) return "found"; int bockup zarr[n-1];
arr[n-1]=x; for (intizo; i++) if (ass[i]zzx) arr(n-1] z backup if (ixn-1) seturn found"; return "Not found"; Bubble sort, Insertion sort, selection sort are inplace sorting algorithm Bubble of Insertion sort can be skeptied as stable algorithm. Merge sort sort is a stable but not an inplace algorithm.

Insertion sort is also used to o online sorting. Quick sort is not stable but is an in-place algorithm. Selection sort is also an online sorting algorithms - Time Complexity:

BCZO(1) 00) Avg. Care = O(n log n) Worstz T(n)=T(n/2)+c Recurrence relation T(n) of C if i>zf T(n/2)+C 6therwise Recursive 7 7 0) int binary (int arr [], int low, int high intx) Ef (1000 > high) seturn -1", 9 nt mid= [low + tugh]/2;
if (X = z arr [mid])
return mid;



else if (x < a > x [mid])

setus n binary (a > x, low, mid-1, x);
else return binary (arr, mid+1, high,x);

Complexity 3 Algorithm Worst Best. Aug. 0 (n2) $Q(n^2)$ (n2) Selection 0(n) 0(n) 6 (n2) 52(n) Bubble 0 (12) 52(n) Insertion O (nlgn) 6 (nlogn s(nlogn) Heap O (nlgr O(nlogn) SL(n log(n)) ywick. O (nlogn) Se (nlog(n)) O(nlog n) Merge

Quick sort is best in terms of practical ene as most wide sorting algorithms at present (8)

-> Has sunning time of O(n2) Chat makes.

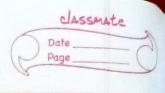
It suspects be in applications

-> Most after than not rules at O(nlogn)

-> High space efficiency by executing

Po place.

-) quest utility in Chrogramming language is powered quick sorts





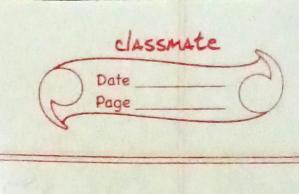
Woost Case: O(n2) - O(n2)

The coort case occurs when the partition process picks up greatest or smallest element as pivot.

Best Core :

- position process a picks the middle element as pivot.
- Read 24B data using 26BB RAM:
 Read 24B of data in main memory
 4 sort it by quick sort a
 Write sorted data to dick
 - Repeat it all contil all data is in sorted 24 B in e 4/2 = 2 chunks:
 - · Which now need to be merged in a single file.

Perform 2 way energe & store the result in buffer [needs 200 m B] from each gorted chunk into input I allocation for output buffer.



(120, icn; 1++)

for (jzo; jcn; jr+)

a[i] + a[j] ok

