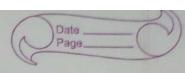
Julorial 3 Sols unt Linear Search (int * arr, int n, int key)
for i > 0 to n-1 if arr [i] = key velturn i return -1 iterative unsertion sort: void unsertion_sort (intarr [], int n) ent i, temp, j; for i = 1 to n temp = arr [i] while (j >= 0 AND arr [j] > temp) arr [j+1] = arr [j] J = J = - | 1 arr [j+1] - temp execursive unsertion word: void insertion_sort (int arr [], int n) uf (n = 1) unserlian_sort (arr, n-1) ·last = arr [n-1] while (j >= 0 & f arr [j] > last) arr [j+13 - arr (j) ara [j+1] = lad



insertion sort is called online sorting because it does not need to know anything about what values it will sort and the information as suggested while the algorithm is summing.

serie complerity: lest case: - O(n²); worst case = O(n²)

space u: O(1)

Ic: best cost = Bist case: O(n); worst case = O(n²)

Space complexity = O()

merge sort:
Jime complexity: Best case: 0 (n logn); worst case = 0 (n logn

Jace " : 0 (n)"

Duck sort:
time complexity: Best case = O(nlogn); worst case = o(n²)

where is a complexity of the contract case = o(n²)

Time complexity: Best case = O(n2); worst case = O(n2)

space (= O()

time complicity: Bust case = O(nlogn); worst case = O(n)

space 11 : O(1)

istable online unplace Sol 4 - Sorting VIV selectionisort unsertion wort merge word quick wort heap wort dribble isort Ad S. Herature lunary search (intare [], int I, int & intz) While (1 < de) { unt m < (0+x)/2; if (ars [m] = x)
veeturn m; if (arr [m] < n) ulse m+1; $m \in m-1$; June complimity: Best Case: D(1)

Average of: D(logn)

Worst 11: O(logn) of Care [mid] = 2)

elle if (arr Emid) > x)

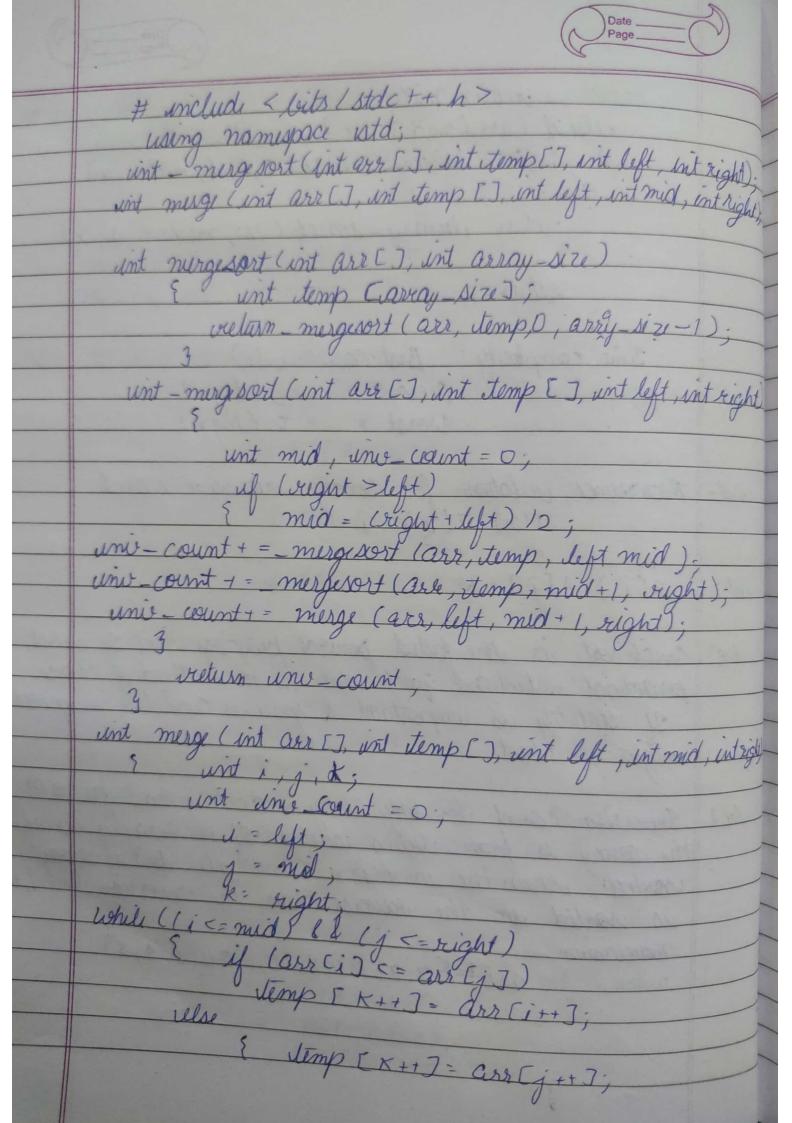
elle if (arr Emid) > x)

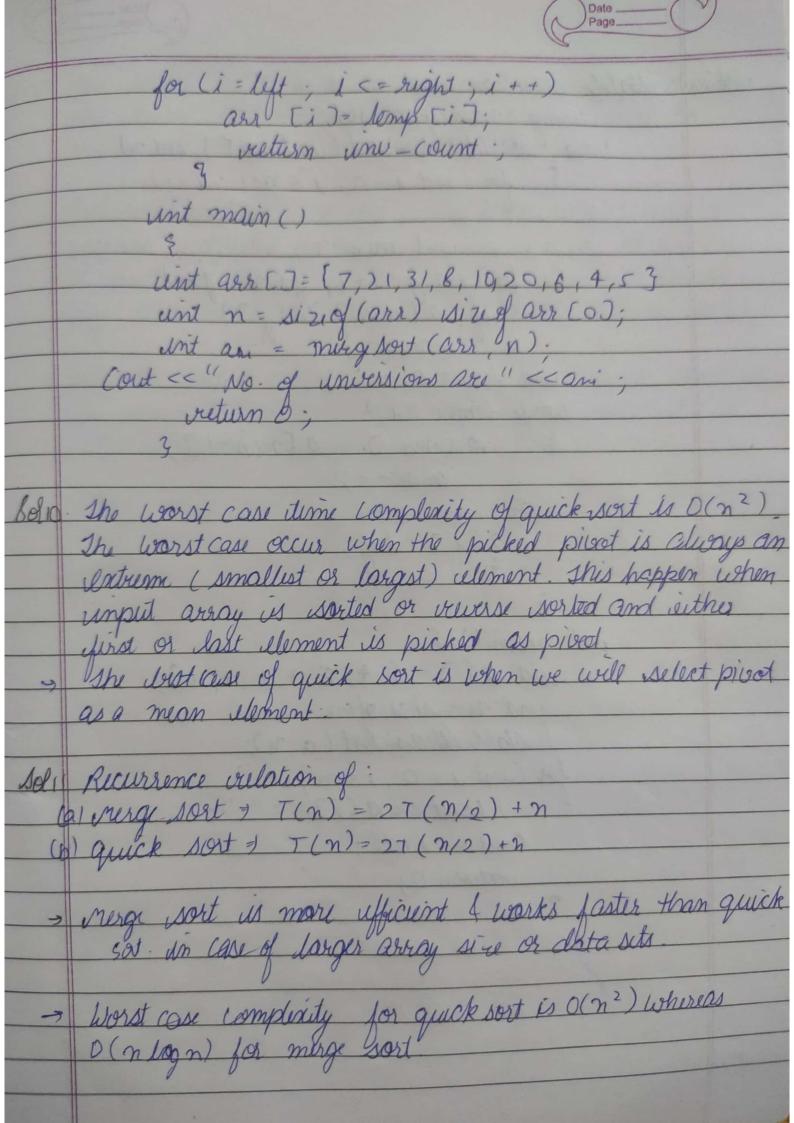
return linary-search (arr, l, mid-1, x)

elle veeturn elemany-search (ars, mid + 1, tr, x) yestur-1; Jime complexity: Best Case = 6(1)

Overage 11 = 0(logn)

Worst 11 = 0 (logn) Recurrence relation for dunary recursive slorch $T(n) = T(\frac{n}{2}) + 1$ M7 ACIJACJJ=K processed situations quick wort is the method of choice of stability is important & space is available messesort might be best the array is from being warted. If the array is already sorted, then the inversion count is 0, but if array is sorted in the reverse order, the universion count is maximum arri= 87,21,31,8,10,1,20,6,4,53





Sol 12 Stable setection wort using namespace std; for lint i=0; i=n-1; i++) unt min = 1; fa (int j=i+1; j<n; j++)

if (a [min) >a[j]) min=j; unt key = Q[min]; While (min >i) a [min] = a [min-1] Q (i) = key; unt main () § unt a[] = { 4, 5, 3, 2, 4, 13; unt n-size of a) stable selection sort (a, n). for (anti=0; icn; itt) cout « a lé] < " " Cout coundl. valur 0;

