Page No.:
Date: / /

U20 CS110 Krishna Pandey

Criven the following algorithms answer the

- Linear search

Binary search

selection sort

1. Analyse the time complexity of above algorithms and write recurrance relative for the same.

Sinear search: The idea behind linear search is to search given element in

linearly in the given array

A recursive approach to linear search.

first search the element in the 1st loca

if not found it recursively calls the

linear search with the modified array

without the 1st element

when, n=1 T(1)=1

T(n) = 1+ T(n-11

= 1+ (1+H1 -- T(1))

T(n) = 1+n-1

Page No.;			1
Date:	1	1	

T(n)=O(n)

Binary search

The approach is to check weather A(n)=n.

If x < A [n] then consider lower half
of the array or else upper half of the

After every iteration problem size reduces

Recurance rela is T(n) = T(n)+1

T(n) - time required for binary search in an array of size n

T(n) = T(n)+1

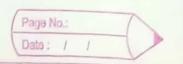
 $T(n) = T\left(\frac{n}{2^{\kappa}}\right) + [+1+...]$

Since T(1) T(1)=1

When $n = 2^K$ T(n) = T(1) + K

 $T(n) = 1 + \log n$ $K = \log n$ 2

 $T(n) = O(\log n)$



Selection 80xt: We need to 80xt an array using selection 80xt, for that we bare to find index of minimum element. Each individual iteration takes a const. time.

the no. of iteration of then loop is n in first call then n-1 & so on

The recurrance sell is

T(n) = T(n-1) + n

T(n) = T(n-2) + (n-1) + n

T(n) = T(o) + T(1) + T(2) + -- n-1+h

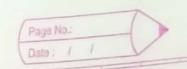
T(n)= D(n+1)

 $T(n) = \frac{n^2 + n}{2}$

we know that, there are a calls to swap and each call takes const- amound of time.

using asymptotic notation, O(n2) term in most significant, so running time of sol soft is O(n2)

T(n) = 0 (n2)



2-	
0	complexity of above algorithm. Can you predict the same based on your Implementation of above algorithm.
sd-	O linear search on worst care we have to compare a elements of array, so it takes O(n) operations by implementation
	3 Binary search By implementation in each recursive call the search get reduced by half of the array So, for a elementa, there are logar recursive calls.
	3 Selection sort O(n2) become are 2 nexted loop.
	from above we can say that we can predicte
(i)	Do they match with theoretical time complexity Yes No Yes
	If yes then write the time complexity of all if No, then write the difference
	lineary search = O(n) Binary search = O(logn) Solution sort = O(n2)