2-001

5	Answer to the Buestion No-1:00 and purities
	65 92 93 medication traces
	To setup the recurrence relation for the
(141)0=	worst-case time complexity of a 3-way search,
	we need to first analyze the process. The
	Pseudocode for 3-way search is given below:
	function three Way Search (area, left, right, key):
	if left 7 right:
	return -1; 11 key not found.
	(m) = (m) = 1
*	mid1 = left + (right-left)/3
*	nid2 = right - (right - left)/3
	10 10 10 10 10 10 10 10 10 10 10 10 10 1
	if arr [mid1] == key:
	Inompreso returno mida e fatrogai tros quet
	if orth [mid2] == Ley!
c2.25	return mid 2
	Test 1 feb er 6 wednesdag
	if key anritmide]:
	return three Way Searich (art, left, mid1-1, key)
	else if key > arr[nid2]:
	return three WaySearich (artr, mid2+1, right, key)
	else:
	return three Way Search (arm, mide+1, mid2-1, key)
	d 1-p

ta	the function splits the input array into three
	parts and recursively searches in the appropriate
6)	parts and recursively searches in the appropriate sogment.
	dreader it a some new or - to French through
	Now, each iteration makes two comparisons at position
	M/3 and 2n/3. The function then recursively searche
	one of the three parity, each of 5:2e n/3. So, the
	recurrence relation is:
Luo Cu-	So the worst copertine complexity of B
	MAY MEAN T(n) = T(n/3) + O(1) = NOTROS
	where, T(n) is the time complexity and O(1) refers
	to the comparisons.
	returned a formal the same way for the way .
	We solve the recurrence using the iterative expansion
	method as shown below:
	T(n) = T(n/3) + O(1)
	T(n/3) = T(n/9) + O(1)
	Same of which (Ava) on the
	Therefore, T(n) = T(n/g) + O(1) + O(1)
	= +(n/6) + 20(1)
	T(n/9) = T(n/27) + O(1)
	similarly, T(n) = T(n/27) + 30(1) = T(n/3) + 30(1)
	So, after k aterations, T(n) = T(n/3k) + k O(1)

The recursion stops when 1/3k = 1, solving that my = 1 La construction frompost. => 3k = n went cook stere from menting xox porusons at main end and and therefore, T(n) = 0 (logg) and at so and recurrence nelation is in So, the worest case time complexity of 3-way search is O (loggn) I where , I'm is the time complexity and CO raters We solve the recurrence using the stonative exponsion method as shown below. T(n) = T(n/a) + O(1) T(n/n) = T(n/n) + O(1) Therefore, T(n) = T(n/g) + O(1) + O(1) = +(00)+20(1) (D) = T (M) = + (D) + + + (D) + = (M) + 300) So ofter k stocations, Tompart (Mg) + k c (1)

	Answer to the Question No. 72: AM-011081 priso
01	20 30 30
9)	To compare of BUILD-MAX-HEAP and BUILD-MAX-HEAP
	creates the same heap, we need storate through
	both processes. BUILD-MAX-HEAP constructs a max-heap
	by Stariting from the last non-leaf node and calling
- [0]	MAX-HEAPIFY to push elements down in the heap.
	BUILD-MAX-HEAP constructs the beap incrementally
	by inserting each element one at a time using
	MAX-HEAP-INSERT Is for ab GAGH-XAM-OJIUG
	some has for
	BUILD-MAX-HEAP and BUILD-MAX-HEAP' do not
	always create the same heap for the same
9A3H-X4	input array to al que south set seplero of d
MANTE	we need to understand that the origin
New	Cet, an input array -> [10, 20, 30, 40]
341	Using BUILD-MAX-HEAP,
	Initial: MAX-HEAPIFY (A12) on 20:
from	319 NO HZO-2N30 (M.CIJA, A)492N309A2H-XAM
qui	+: who the say one to sent to
-	and the stragand from the wintries of
(34)	MAX-HEAPIFY (A, 1) on 10: MAX-FEAPIFY (A,2) on 10:
7-0-2-	10 30 man son ent to set of
	with n elements is along so an insertion
	20 ent 10 port sevet
	Final heap -> [40,20,30,10]

Using BUILD-MAX-HEAP, sit and suf of reward40 20 30 30 1940H-MAN-MINE 100 70100 7020T creates the same neap, we need storate through final heap -> [40, 30, 10, 20] by Starting from the last non-keef rede and culling So, BUILD-MAX-HEAP produced [40, 20, 30, 10] while BUILD-MAX-HEAP' produced [40,30,10,20]. Thus, it is proven that, BUILD-MAX-HEAP and BUILD-MAX-HEAP do not always generate the same heap. BUILD-MAX-HEAP and BUILD-MAX-HEAP do not almons create the same him for the some 6 to analyze the fine complexity of BUILD-MAX-HEAP, we need to underestand that the algorithm calls MAX-HEAP-INSEFT on times, where each call inserts an element and maintains the as heap properly - XAM MAX-HEAP-INSERT (A, A [i], n) -> inserts an element at the end of the heap and bubbles it up to maintain the heap property. In worst case, an element moves from the last level to the root. The maximum height of a heap with n elements is O(logn), so an insertion takes O(logn) time. Final heap > [210,20 30,10]

Therefore, 6-on not essel sut a - recent T(n) = E O(logi)recentive calls to steel f the SUICKSORT algorithm of the last wife to good sit to so with the last will be so so with the country of t replaced it with an iterative approach.

= n logn - n +1 2 n log n -> O(n log n) the PARTITION (A.P.M) -> reconsciouses the element such that, BUILD-MAX-HEAP' runs in O(nlogn) to build an n-element heap in the works + case scenario. formadie the ent store planes in to sweet and turn updates the P value. Each Horation convactly partitions and sorts the left subarray, then apporter p to proceeds the right Subornay iteratively. The TRE-SUICKSORT Offertion performs the sout in the same manner as GUICHE Therefore, TEE BUTEKSOET DERMECTIN SONTS TIL

No. of the last	Answer to the Question No-3:
75 N	
9)	From the section 7.1, which contains two
_	recuresive calls to itself, it is proven that
	the QUICKSORT algorithm connectly sonts the armay
	A. TRE-BUICKSORT differs from QUICKSORT in only
_	the last une of the loop. In the TRE-QUICKSORT,
	we as eliminated the second recurrière call and
-	replaced it with an iterative approach.
-	The same was I + N - NEW ON =
	if pzre, the firstion terminates
r	the PARTITION (A,P,M) -> reastranges the element such
- 0+	that elements in A[Pig-1] are less than or equal to
-	the pivot and A[q+1:17] are greater than or equal to
	the pivot.
	The function recursively sorets the left subannay
	and then updates the p value.
	Marine Committee of the
	Each iteration correctly partitions and sorts the
	left subarray, then updates p to process the right
	subarray iteratively. The TRE-QUICKSORT effectively
	performs the sort in the same manner as QUICKSOFT.
	Therefore, TRE-BUICKSORT connectly sonts the
	arereay A.

67	The worst case seenanio for stack dept o	courcs
- forma	when the partitioning 95 highly unbalance	d. For
0-	example, in an already sorted on reverse	sonted
170	input, the paratition picks the smallest con	largest)
- Lunn	element as the pirot so, we get and	
	FRE-SUICKSORT (A, 1, N)	
	TRE-BUICKSORT (A, 1, N) TRE-BUICKSORT (A, 1, N-1)	
	TRE-BUICKSORT (A, 1, n-1) TRE-BUICKSORT (A, 1, n-2) TOTAL MODERATE STATE OF THE S	
	THE-BUICKSORF MOD (N.P. R)	
	71.24 21.14 2	
	TRE-BUICKSORT (A) 171) TITHAT = NO	
	:(12-12-16)+1	100
	This results in n recursive calls, leading in a	stack
	depth of O(n).	
	Fore, an arriay [1,2,3,4,5], the graphical	
	Fore, an array [1,2,3,4,5], the graphical representation will be	
ادرد	$T(5) \rightarrow T(4) \rightarrow T(3) \rightarrow T(2) \rightarrow T$	r(i)
Date 1	smaller portion and sterates over the lang	
of milities	n=5, two eforce 5 Steps to complete the rec	cursion.
	on walf tre elements (at most), the depth tollows a	
	This confirms that in the works + case the	Stack
	depth of TRE-BURKSORT will be O(n).	
المد	Therestone, beeping the running time at company	
Must 13	wonst case stack depth for the modified ale	
	will be Octogn).	

0	to ensurce a maximum stack depth of O(wgn).
JATON J	while maintaining the o(nlogn) expected running
Sonsted	time of the algorithm, we need to make sure to
(tespera)	periform the recursive operation on the smaller
	subarray and îterate on the larger subarray.
	TRE-BUICKSOFT (A) TROSTOFT
	The modified TRE-BUICKSOFT will be:
	TRE-SUICKSORT (A. 1. n-2)
	TRE-BUICKSORT-MOD (A, P, T)
	while per:
	9 = PARTITION (A, P, r) 2 2010 8-397
	7f(9-PLR-9):
Jack.	
	else: (1)8 to ital
	TRE-BUICKSORT-MOD (A, Q+1, TE) NO NO TO
	R=Q-1 ad lieu nother maringer
	00 000 000
(i)	This modified algorithm applies recupsion on the
	smaller portion and iterates over the larger one
moldmi	in each îteration. Since, each recursive call operate
Received to the second	on half the elements (at most), the depth follows a
ام ام ال	logarithmic pattern. The runtime will be,
	(O(nlogn) + O (logn)) = O(nlogn) 1+gab
	Therefore, keeping the running time at O(niogn), the
	worst case stack depth for the modified algorithm
	will be Ollogn).