Revision, paper ? 2. a) First I will write a function partition; which partitions an array with the frest element as a privation det partition (arr: kreay [Int], l: Int, r: Int): Kreag I Int = 1 var i = l+1; var j = 2; val prot = azz (l) MIno: arelett. in expirat & are Ejerz handenis jes 2 ( are [0.. L) = are 0 [0.. L) mare (2 - . N) = are [2 .. N) . Maranare [lane) is a permutation of arco[1.2) : while (i = j) of looks out nout receid is (1-1) rec hanger it ( acce (i) e pivot) rist = land soule set section site populsed was broom no substated Jam 24 14 agreed ion wal swap = are (j) were libere to beauty wi . e. i con ari ( july exare lin) is our our li en tressessions. are (il = swap 5 a last 1 lates dues 1, the self pound one in office and her later was gare placed and some val swap = are (i-1) ... + but a little and air (i-1) = pivot of the alimet man a rate of in arrived = swap is a constitute a truste buck i - 1 5 det sort (as: Array EInt], u: Int): lust = 1 Not be partitioned a g Sozt(a, b, u)

det a sort (a: treay EInts, lie Int, e IInt) ! Unit = 1
if(z-e >1) 1
entitod bet partitioneda, birthuil a storm Min 1 101710
a sortheigh ) a gisorthanne etter) aut alle mans no
5 1
del partition are: Kreay Etuls, P. Just, v. Inst. ): Robar I red &
voz i = l+l, voc j = r ; vol phyoloaze (E)
b) This behaviour can be intefficient bécause some
elenientstra may berranapped two Huest Suppose
we have the program from the previous question
It are (j-1) is bigger than the proof and aredilis
bigger than the proof, then air (jul) and arr (il would
swap At the next Heration, we would have to swap
the element at are (1) again thuster way this might
be inefficient is if we have many requal elements.
and a films
c)
det partition (arr: Array [Int], l: Int, r: Int): Int = 1
det partition (arr: Array [Int], l:Int, r:Int): Int=1
det partition (arr: Lreag [Int], l:Int, r:Int): Int=1  var i=l+1; var j=2; var privot=arr(l)
det partition (arr: Array [Int], l:Int, r:Int): Int=1  var i=l+1; var j=r; var privot=arr(l)  //Inv: arr[l+1i] < privot < arr[jt]
det partition (arr: Array [Int], l:Int, r:Int): Int={  var i=l+1; var j=r; var privot=arr(l)  //Inv: arr[l+1i] < privot < arr[jr]  // leriejer A arr[oe] = arr [oe] narr[rN]=
det partition (arr: Array [Int], l:Int, r:Int): Int={  var i=l+1; var j=r; var privot=arr(l)  //Inv: arr[l+1i] < privot < arr[jr]  // ler'=j=r A arr[oe) = arro[oe] narr[rN]=  // = arro[rN) A arr[lr] 15 a permi of arro[lr]
det partition (arr: Array [Int], l:Int, r:Int): Int={  var i=l+1; var j=r; var privot=arr(l)  //Inv: arr[l+1.i] < privot < arr[jt] \ n arr[r.h]  // ler'=j=r n arr[oe] = arro[oe] n arr[r.N]=  // = arro[r.N) n arr[l.r] is a permi of arro[l.r]
def partition (arr: Array [Int], l:Int, r:Int): Int={  var i=l+1; var j=r; var privot = arr(l)  // Inv: arr[l+1]:   c privot = arr[jt]: A  // lex'=j=r A arr[0t] = arr [0t] A arr[r.N)=  // = arr [0[r.N] A arr [0r] is a permi of arro[lr]  while [arr [ ] arr [ ] arr [ ] arr [ ] arrol [ ]
det partition (arr: Array [Int], l:Int, r:Int): Int={  var i=l+1; var j=r; var pivot = arr(l)  // Inv: arr[l+1i] < phvot < arr[jr] / n arr[rn]  // ler'ejer n arr[oe] = arro [oe] n arr[rn]  // = arro[rn) n arr[lr) 15 a permi of arro[lr]  where fart from front from  while (iej) i  while (iej) i  while (iej) & arr(i) < phvot) i+=1
det partiton (arr: Array [Int], l:Int, r:Int): Int={  var i = l+1; var j = r; var pivot = arr (l)  // Inv: arr [l+1i] < phvot < arr [jr]  // e=r'=j=r A arr [oe] = arr [oe] A arr [oe]  // = arr o[r N) A arr [lr) 15 a permi of arro.[lr]  while (arr // prot) fixed  while (i e j) i  while (i
det partiton (arr: Array [Int], l:Int, r:Int): Int={  var i = l+1; var j = r; var pivot = arr [l]  /[Inv: arr [l+1]: i ] < phvot < arr [j-r; v]  // ler's j = r A arr [o e] = arr [o e] A arr [v]  // = arr o[r N) A arr [l r) 15 a permi of arro.[l r)  while (arr [root] [o r)  while (i e j) i  whi

det remove (elem: Int) // Post : 524 = 524 @ / elem y /Pa: elem is in set ust otherwise or telem assert ( Declem & Mance W) do : Ash ! right : Luch : Complance = A (+wI: mals) wIel 4st / Post: returns . elem + set and thurst on C Evar a = were Record [ det ] ( +uI: wels) bloo feb ( etse to nown & Blelemble + set = +se + troy ) 2011 1 1000 = # 4 x 1 x (x 1 == + 200 1 = = There see = 1x lo (x) == tous 1 ... It is tin! that Estate port extendentality of the Lynth = state 1 + 22 + NI + 1, DX + 10 · 6 def remove ( ran 1 2 ml ) & & mlove "V sola celo man celum ( this and cepy sol sold of equal elements the spartition trunction worked the at most once. But if the accord contains in moning one of the problems - every element will be moved sort funetton will be with same. This isolves outy a self-band return late of elemented Post seet = set + letomb and return true a (1/ = 5 wap Presidence Done A22 (1-1) = HUPF Hadrey ! the tree to the relation out the (HHI) save wanted

// Post i refurus # Set with the post i technical and the feel whee

-

1

\*\*\*

-



b) We wied to rew	rite only the	add and remove from
Hous s		Fronty = (1-1) 550
/ Pre: elem & To.	. N)	amore all'a
// Post : set = set +	Telemy and ra.	furn true
// v set= and	return talse :	t elem toet
defradd (element	unt ) & Bolean 10 in	e mostment tree wer
howary od 13hu to	SINGIA PTAUL	end of the moblems
// Pre : elem + [o.	D) )	Lud . was I some . And
// Post set = set	holowit mud	return trues soups
		lse sif welen & set
det remove (elem		
		.3
<u></u>		A tollow Martin
	x tends Int Set	1 Etall + exels
		th = timi
// DTI : count = #		
11 D(3 - Walt = 11		1 h/s
/ thi to vai	Boolean	Trosta set a set to
Evar a = new L	rray [that] (N)	Anticonstal she lah
var count = 0		2000
		The street of the street
det aas cecemi, in	al about asse	rt (Ds elem & elem c N)
		धर्म तु: उन्तर्गत र द <b>र</b> भे
a (elem) = true		hat a last of the off
		Essents Swarrage to A
5	5.57	, 2,51,4 5 50 acres 154
1		
det is In. (elem: I		
if lelem 20 l	( elem ZN) fal	se actionist
a (elem)		
4		

```
det remove (elem: Int): Brolean = (" a milab. ) lines
      val oldvalue = a (elem) to assert (0 & elem 88 élème N)
      a (elem) = false
      50ld Value
   Mr 1885 I Christian Line to a fabric & process on a section of the
    det sied were teel ula met own (1 = - 2 = d) !!
 d)
 det sort las: kreay & Int ] : kreay [Int ] = 1.
     var new Array = new Array [Int] (xis. length)
  for lie Obentil. N.) Vioit Margradd (xs(i))
    The assection whether clame (0. 10) is some in the
    // function
    var i= 0; var j=0
    MInv: [0... j) are in new kray hi cooks lengtente de
   wwee (i a new texay, length) former
       wwile (bithap. a (j)) inju = 1 ... harmen ) ....
       * new teray (i) = bithape it have been to the
                              Man sa Mal America
                     while temperate dates on a
                           · extract = textract. Hour !
I rear light out for blice happe wit harme server. I note
                    that is the second
4. .
                     may receive a recovery to the
class Tree (dation: Int; left: Tree, right: Tree)
                             done was a traje . There
61
def inorder (+ : Tree) : Unit = 4
   if (+. left! = null) inorder (+. left)
```

6 6 6



print (+. datum + ") u) estante les l'entre mais 1 sus	1sl
it (terigut: = mille) inordie (tiright)	1 1 1 1
340.01 / 200	1.1.5
<u>c)</u>	161.3
det make Tree (u: keray [Int] a: Int, b: Int)	1 Tree = 4
if (b-a == 1) new Tree (ula), null, mull	Jan Joh
else i	
also 1	son to beekto
val midIndex (= 16-a)/2 me posses	M Jev
new Tree (u luid Index) make Tree (u, a).	(xs briEbin
I hnoke tree la mid Index + 1	
services were there chance 20 - 12 is done to be two	
<b>9</b>	3.00.1
d)	= / <b>*6</b> ·)
det inorder (it): Tree ): Muit outer is are fine!	1000
vac current = + //current points to the root	1,12,000
w/InviT(current) is not printed yet	D. No.
while (current!= mell) ( add 111) = 1 1 1000 1000	. · • • • • • • • • • • • • • • • • • •
if (current, left == null) 1	+ 1
print (current datume " ")	
current = current. Ugut y	<u></u>
node else 1 // make current the right child of the	rightmost
Men in the left subteen	. 1
war node = current. left	
wwile (node right != null) node = node	: right
node. right = current	
current = current. left	
4 K-valle melletie	
J	* 17
<u> </u>	

e) The code in the previous questions straverses a node at most twice wand because withis was tree where will be no cycles. Therefore, the reade rune. In this proportional to the saze of the tree. toys . . tox. verst- h bor. tirener ! wall man stren. hat (then so stren heat ) 1; at the described : String, class Node (Veount: Int, next: Node) ( met. execut or view Hode of exercise, and the exercise revised b) ( France, lengt. def add (w: Heing) = 4 val i = nash (10) // the cell in when wineeds to go var current = table (i). next // header well -// Inv: w is not in L(table(i).next, current), where // L(a,b) = [] if a = 6 / L(a,b) = a:1 L(a. next, b), if a = b wwite (current: next!=null || current. word!=10)4 XX exercity your sources sources current & current next : 11 shore : 17 ) space tease to a thinest your reactions the le off ( current word == 20) current event +1=1 else renzient: next = new Node (w ; d; nute) F ( Junior . Secret & Laurer . ) 1841. Part ward place to ward, room, and there there the C) det output (bucket: Node) : Node:= 1 .... from the 1 , 1 ... var newlist = new Wode (4?4,0, null); var last = newlist Month L'Courrent, mullipe have. That abeen added to newlist // n L (new List, next, null) is ordered

. 1		1	1	1		
- 1	IJ	d	1	ч	9	

\*\*\*\*\*\*\*\*\*



wwwled currentine	xtil zit	well ) Constrain	self oil ske	· salt (s
// Invest new !				
while ( last . e				
		and and chere		
last = last	h next	4		
if (laxt. next =	- 2 nul	1) last nexte u	ien Node (cu	erent. word
		ريون	ettent count,	null).
else 4	sh	all: fress first	. Lussoil sho	4 3381.
last.next=	new	Node ( current	- word curren	x. count
	. 6		last.u	
4		1.0	I production 1 )	the 1sh
last wew List	ر دولالم	d star add N	( ost darn = 1	100
current = time	ceuth na	(i) went to	Male - Aunter	300
everent ! where	Anne.	librated 1 wi	Av 3 ov 10	es 1
new list		2 = 30 11	23 = 14,2	1 1
9	at an of	1. (d) 1x301.2.).	lus = U,s!	3
		o Il Many = 1-to		
d1 1.56	hisas.	more (de - b	root browns 1	• 4
det merge (11: No	de, l	2: Node ) : Node	mark brown	•
var ever = 11.	next;	var unit = 12	. next	Ċ
var newlist = n		• • • • • • • • • • • • • • • • • • • •		
white ( et winder e	eiter to 1	rest, ! = need ?	& cure e next	· ! zuull ){
if ( cuerl. count	tecin	ce 1. count 14		1
laxté new l	Jode (	curr 1. word curr	1. count null	.)
last = last, u	<del>ux</del> t			•
		bil hair		
ich var slavk - chrotist	grand C	7 1 1 1 shall a	are Share	• • • •
else 4		1.45/-		
lastinext =	new	Node Ceure 2. 4	sordicerrili con	ent until
last = last. w	next.	n i Bara James	*/- ****** 1	4

ener of to and marging the is lifear	at the terms to be
- cal del 18 / del del 18 (2) o ca estrono so se	
it ( cure 1. nex fet null) know and moreon	
wwe (rue 1. next ! = well ) flast - new A	
cure is bount public son	
last = last. next	•
cutth = cuttl. next	.1
5	·
5 } - I wate of the I to I to I govern . L	pertining fob
else 1 mar out of ""	y 2   12   12   1   1   1   1   1   1   1
willel cured. next (1 = mulle) to 1 Money	0 -11.0
· last = new Node (int front cuttico	unt unti
last = last. next	
curr l = curr l. next	".0" Luing
confic to prince ( of (it)	1-511
<b>y</b>	Ŀ
y new list	
£)	(0
det arrange Mi: Node 120 4 to 13 yours : 1 how.	det teag to
par new live = new Wode ( 17 ,0 mull) ; va	
var life Den 1 1 a reformation non 1 and son	
// Inv: table [o. ii) have blein added ito new	
11 wewlist is whited ND cline N I !!	
write(i < N) 1 hardware and so in	
var templist = output (table (il)	(1 ) gert fil
merge (newlist, templist)	1200-
(M) solver of well that by I do to a first solvers were	C1 112.41
\$ newlish	

-

-

99999

4

16

6

6

-



1) On average, each bucket	LANGE TO STANKE STANKER	Tree AV DOW
net would take o( =)2)		
to the number of words	- G	•
overall the average time	Halland Bury D.C.	23
The word lease is when	Ash bushed to	p 3 wodes
and they are in descend	Ing' worder 1991	4,
	Lyan . Mal	· Frai
ι.	ت فيدوار برديد	
۵)		,3
det printFrag (d, Array 2 Int	] & ;: Int. k: I	int   =1 '
print (" " + " ") //T	wo space	. s.1.
for lic 0 & cente j) pri	700	
for line jourth looked up		
•	tase. Sast	
print (" 0. ")		
for (i - 0 until k) print		Ė
4		Į.
		Mil man
<b>b</b> )		
det trag ( N: Int) : (Array)	[ Int I all I Jut, I	m+ ) = 1
war are new treay EInt		
var j=0, var k=0; va		
var bittlap = new likay		
// Inv: a (i) = 10 12. a(i)		
1/ remainders we have		
bithap(1) = true! 1 100	_	
mulle (numerator e N)		
( while ( numerator % 0		, ,
alil = numerator / N		
WAS A STOCK		

numerator = numerator % N

\$ bitMap(numerator) = true

\$ wwwle(numerator c N) numerator x = 10

while(alj! = (numerator / N)) j + = 1

\$ (a,j,k)

-

4

9 9

4

-

4

1

-

-

c) Every time the numerator is less than W, multiply by 10 and put a zero into the array and increment the length of the "real" array. The code correctly identifies the recurring segment, because it creates a lit-map set, which holds all the remainders we have seen. Using lit-map set allows us to look-up whether & we have seen a remainder in O(1) = 1 the program has complexity O(N). The program correctly terminates because it looks at the two different cases: 1) numerator % N = 0, 2, we have already exhauster seen the remainder value.