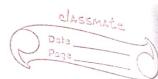


Name - Aastha Rajani Enby No. - 2021CS10093 To prove - The lang division method gives integer square scoot of any given number and the smallest number which should be subtracted from that given number to obtain a Perfect square. Paroof - By logical deduction. het us say we need to find the integer square ovort of any number eay a. het b be ût's integer square root. Let h be the least significant digit of b and x be the number formed by sumoving h from b. At every iteration of long division method, the value of "a" gets updated, initially being the mos' fost one or two digits of given number. At each iteration use asse suggested to find x and h $(10x+h)^2 \leq a \quad and \quad (10x+h+1)^2 \gamma a.$ Both the conditions together grewanters the uniqueness of x and ho Simplyifying the first inequality (2.10x+h)h < a- (10x)2 The subtraction in the oright hand side of this inequality is done by placing digits from the left most side. Analyzing the left hand side of the inequality 2.10 x involves doubling the number or and multiplying further by 10 to make space for next digit h. we have to choose next digit h so that the new number 20x +h when multiplied by h equals the now or is less than the updated a Coblained after multiplying the difference of the number obtained after subtraction on night hand side. The value of a then gets updated by. adding the next two digits of given number with 100 times the difference obtained above. This is exactly what we do un the long division process. Therefore every iteration of long division outputs the next digit of integer The percere terminates, when we are unable to find any such a and b for which the inequality above is satisfied. flence, et is justified that this method rectuerne the integer square scoot, and the lastly updated a is the smallest number which on subtracting with the given number gives a perfect square.



fun multi-adder (. court, arr 2, caux, index, sum). # ava 1, asia 2 = two strongs of untegers of same length A case = current carry # index = current index # sum = current sum. if index =-1 return sum Lo Fund the sum of digite of ara, are 2 at the current inder with care La concatenates the units digit to "current sum and Stores the rest in casa. La Desiaturely starling ferom the last digit, perform digit by digit addition and update care & sum until all digite acre exhausted. fun complement (aver, inden, ans): # are -> string of integers # are -> complement of # inden > current inden the are upto unden. Lo Recursively subtract every digit by 9 and concatenate Lo Once all the digits are subtracted once by 5, we add 1 to get 10's complement. fun make_equal (avorz, c, p)

orginal

orginal

ething of integers # p > current length. # arr2 -> strong of integers # i -> Required longth L> If i=p make_equal ("0"100 12, i-1, p)



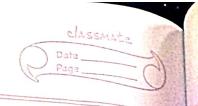
fun sub (aver 1, aver 2)
Mark 1 = Subtractor # ark 2 = Subtrahend
Is find 10's complement of aga 2 using complement function
L> Add 10's complement of are 2 to are 1 using multi-adder
fun compare -ge (num1, mem2, i) + num1, num2 → string of integers
If $\ell = Size (num 1)$ integers
return true
else.
Company the ith digits of num 1., num 2.
If (num: 1): > (num2):
return frue
else if (num 2) i < (num 2) i
return falle
else compare-ge (num1, num2, ê+1)
the same and the s
fun compace gt (num, num2, ?)
Works smilar to compare - ge, except for i = size(rum)
it returns false
ž ura
fun mul_10 (s).
Adds "O" to end of s.
fun mul_100 (S)
Adds "00" to end of s.
0.00 1 / 1 01100 2)
1 / The trope
By make_equal (m), marke und num2;). equal and call campage ge (num1, num2;). Size(num1)-1
fun multi-adder final (num 1, num 2)
fun multi-adder final (num!, num!
By make equal, make the lengths of Size (num1)-1, equal and call multi-adder (num1, num2, 0, Size (num1)-1,
equal come and

```
fun hinavy - seawich - helper (a1, a2, low, high, and)
# Takes tugo change of untegers and finds he [0,9] such that
 (10a_1 + h) h \leq a_2
If high-low = 1 or 2.
 P = (10a, + ara [mid]) x ara [mid]
  [anol] see X ([wal] see + 1001) = p
     ~= (10a) + and [high]) x and [high]
   if q < a2 and a2 <p.
    ( aser [low], 10a, + are [low])
   else if pear and arer
     (ass [ mid], 10a, + ass (mid])
    else
           ( are [high], 10a, + are [high])
 else.
    if (109, + ana [mid]) x ana [mid] < a2.

binaoy search - helper (a1, a2, mid, high, and)
     elle
         binasy _ search _ helper (a1, a2, low, mid, org)
```



1												
	ferm	binasy -	search (a1,a2)								-1
1		binasy_	- search _	- helperl	al,	a2	101	9,	LO,1,2,	3,4,5	8,5,0,0	(le,
			<i>.</i> }			. 1			7.4			
			L							1		
						4	<u> </u>		1			
		Ĭ.	-			+ (
			-9	· · · · · · · · · · · · · · · · · · ·					1			
				ş					<u> </u>			
			5 5					1	1 - 4 11 v			
								1 1 1				
									t _o .			
		- 1 9			*				Li nai	1 - 1		
			- 1	(C)	s	. ,.			· · · · · · · · · · · · · · · · · · ·			
									<u> </u>			
						~	_	1,1	1 N . 2			
				, , ,		l.		1 .				
			1		1 P							
_									-			
								l -				
			1		N. J			1 .				
-)			-			
				a 1		-	-		1			
_												
_					7							
_				1								



Pseudo Code

fun int_sqrt (s):

Takes a string of Entegers and returns the Enteger sq. root and difference of int(s) & reasest perfect equare,

92= make a list storing every character of storing as integer in each index; and making the length of the let even if it is not by adding 0 at the start of the list

call int_sqrt_helper.

fun int_sqst_helper (.or, q, num, length, l, stor_118+7: # r > curevent value of a (as decertibed in persof above) # 9 -> the value of sq. st found so face. # num > the current value of divisor.

length -> the number of digits we have operated # 1 -> length of the input slong # str_list \Rightarrow s2 made above.

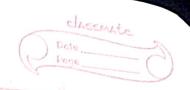
if all the digite are operated return (r, q)

Taking num as & and or as a (in the previous persof) find h and the updated value of num. Say (h, updated _ num). Using timouy search (mem, r)

if you are dealing with last two digits call Ent_egot_helper (updated or, 109+h, updated_

length +2, 6, str_118+). where updated or is updated a described in persof.

> call int_ sqrt _ helper (updated r, 109+h, updated_num length +2, 1, str_list)



where updated or 32 feet the difference of current of and (109+h).h.

& My algorithm takee integer in the foom of string as input and firstly charks if the length of this string is even or odd. In case it is odd, it adds a O in the beginning of the list formed by every character of input converted to Integer. It then starts with first two elements of the list (as integer) and finds the nearest equare less than or equal to that number, and this digit is the most significant digit of our integer square root. The difference of first two digite and this equare is added alongwith next two digite of input forme a in next recuersion and tuoice of current integer equare not multiplied to to forme next divisor. In this way occurringos on until we have completed considering all the digite of input. This is what is done in the long division method which we have peroued gives the integer equare evoot. Hence, my algorithm also reliance the integer square and the reacest smalleet number which when subbrated with the guer number ques a perfect square.

Since, there is a maximum limbt upto which integer can take values in eml, I have defined functions performing anithmetic operations with the help of slongs in order to deal with larger inputs.