	ASSIGNMENT-3							
1.	We the viccionather method to solve the siccionences							
	$T(n) = 2T(n/2) + n^2$			1				
->	T(n)_	level	node	Running time				
				V				
	T(n/2) $T(n/2)$ n	0	2 = 1	n ²				
	(1/2) (1/2)	1	2'=2	2 (~/2)				
	(n/4)° (n/4) (n/4) (n/4)°	2	2=4	2"("/2) = 4("/4)				
	(\(\si_2\)\)\ \(\si_2\)\\\\(\si_2\)\\\\\(\si_2\)\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\	0	2	2 '(n/2;)				
	Num of T(1) = 2 log = n log = 1 log = 1 log = 2 log = n log =							
	$\frac{1}{1} = \frac{1}{1} = \frac{1}$							
	= 2 n ² [1 - (1/2) (1/2)]							
	= 2 n° [1 - 1/2n]							
	$= 2n^2 - 2n^2 \times n$							
	= 2n²-n => 0	(nª)						
<u>Sundaram</u> ®	FOR EDUCATIONAL	L USE						
				ls.				

•

2. Use Marter theorem to solve (if marter theorem can not be applied, relite the vienson) ca. T(n)= 9T(1/3)+n → Hue, a=9, b=3, k=1, β=0 f(n) = 0 (n log n) Case 1:- log a >k => log 9 >1 => 2 >1 => TRUE : 0 (n log ba) = 0 (n log 9) = 0 (n2) b. T(n)= 9T(n/3) +1000n2 → Hue, a=9, b=3, k=2, p=0 f(n) = o(n2 log on) Care 1:- log a > k => log 9 > 2 => log 2 > 2 => False Case 2:- log 9 = k => log 9 = 2 => 2 = 2 => True i) \$>-1 => 0>-1 => True : 0 (n * log + 'n) = 0 (n * log + 'n) = 0 (n * log n) C. T(n)= 9T(n/3)+1000 n3 → Hue, a=9, b=3, k=3, f=0 0 f(n)= o(n3 log on) (ase 1 + log ba > k => log 9 > 3 => log 2 > 3 => False. (a) e 2 t log b a = k => log 9 = 3 => 2 = 3 => False Case 3 + log a < k => log 9 < 3 => 2 < 3 => Time i) \$ 70 => 070 => There

 $\therefore \Theta(n^{k} \log^{k} n) = \Theta(n^{3} \log^{n} n) = \Theta(n^{3})$

2. d.	$T(n) = 9T(n/3) + n^{2} \log n$
	Here, a = 9, b = 3, k = 2, p = 1
	f(n) = 0 (n 2 log n) Case 1:- log a > k => log 9 > 2 => 2 > 2 > 2 => False
	Case 1:- log a > te => log 9 > 2 => 2 > 2 => False
	Case 2 + log a = & => log 9 = 2 => 2 = 2 => Tane
	i) lo >-1 => 1 >-1 => Tane
	i) fo >-1 => 1 > -1 => True : 0 (n log f+' n) = 0 (n log 1+1) = 0 (n log n)
eð	T(n) = 0.5 T(n/2) + n
	f(n) = 0 (n' log °n)
	f(n) = 0 (n log n)
	Case 16- log a > k => log 0.5 velue a = 0.5 : a 15+ does not extresy the condition a > 1.
	Hence we cannot affely master's theorem here.
	regner, we carried apply warren med enginere.
f.	T(n) = 2 T(n/2) - N
0	H. n n = 2 h = 2 k = 1 b = 0
	But this recurrence form given donnot saliefy marter's theorem condition where the equation we
	marter's theorem condition where the equation we
	T(n) = a T(n/b) + f(n) We are the contract of all the contract the contract of the contract
	Hence, we cannot apply master's theorem here.
q .	T(n) = nT(n/a) + nlog n
<u></u>	Here, a = n, b = 2, le=1, f=1
	: a = n and it should be a constant where a > 1.
	as few the condition.
	Hence, we cannot apply master's theorem here.
Sundaram	FOR EDUCATIONAL USE

h. T(n)= T(n-2) +n?

→ Hue, T(n) should be un dividing form and according to the vie currence form given, it us in subtraction form. Hence, we cannot apply master's theorem here.

u. T(n) = T (7n/10) + n2

-> Flere, a = =1, b = 10/7, k = 1, p=0

f(n) = o(n'log'n)

Case 10-log a > k => log. 1 > 1 => 0 > 1 => False.

(are 25- log 6 a = & => log, y, 1=1=>0=1=> False.

Case 31. log 6 a < k => log 1/2 1 < 11 => 0 < 1 => True

i) \$ > 0 => 0 > 0 => Tume.

.: 0 (n k log fn) = 0 (n' log "n) = 0 (n)

j. T(n) = 4T(n/2) + n 20g n

-> Hue, a=4, b=2, k=1, f=1

f(n)=0 (n 2 log n)

Case 1 = log a > le => log 4 > 2 => 2 > 2 => Falie.

Case 2 + log 6 a = k => log 4 = 2 => 2 = 2 => Tiene

i) \$>-1 => 1>-1 => Tume.

: 0 (n * log f+'n) = 0 (n * log 'n) = 0 (n * log 'n)

3. Solve the leetcode question no 53 (Max Subarray)
Implement a solution that submission can be accepted.
Provide a screenshot of your submission.
(Check discussion for solution if you cannot figure it out yourself, a linear solution can be found in the file bentley-max-subarray.pdf in camino)

Solution:

```
class Solution {
   public int maxSubArray(int[] nums) {
      int sum = nums[0];
      int result = nums[0];
      for (int i=1; i<nums.length; i++) {
            sum = Math.max(sum + nums[i], nums[i]);
            result = Math.max(result, sum);
      }
      return result;
   }
}</pre>
```

Screenshot:

```
Runtime: 1 ms, faster than 100.00% of Java online submissions for Maximum Subarray.

Memory Usage: 51.4 MB, less than 54.41% of Java online submissions for Maximum Subarray.

Next challenges:

Degree of an Array

Longest Turbulent Subarray

Maximum Absolute Sum of Any Subarray

Maximum Subarray Sum After One Operation

Show off your acceptance:

f in
```

Time Submitted	Status	Runtime	Memory	Language
01/29/2022 15:55	Accepted	1 ms	51.4 MB	java
12/28/2021 13:32	Accepted	1 ms	49 MB	java

4. Solve the leetcode question no. 240 (Search a 2D Matrix II)
Implement a solution that can be accepted.
Provide a screenshot of your submission.
(You can check "discussion" if you cannot figure out an efficient solution).

Solution:

```
class Solution {
  public boolean searchMatrix(int[][] matrix, int target) {
     if(matrix == null || matrix.length < 1 || matrix[0].length < 1) {
        return false;
     }
     int col = matrix[0].length - 1;
     int row = 0;
     while(col >=0 && row<=matrix.length-1) {
        if(target == matrix[row][col]) {
           return true;
        } else if(target < matrix[row][col]) {</pre>
           col--;
        } else if(target > matrix[row][col]) {
           row++;
        }
      }
     return false;
}
```

Screenshot:

Success Details >

Runtime: 4 ms, faster than 100.00% of Java online submissions for Search a 2D Matrix II.

Memory Usage: 47.9~MB, less than 52.16% of Java online submissions for Search a 2D Matrix II.

Next challenges:

Search a 2D Matrix

Show off your acceptance:







Time Submitted	Status	Runtime	Memory	Language
01/29/2022 16:07	Accepted	4 ms	47.9 MB	java