→ Gridgame

→ Max Rec sum at mtk

→ Kadane

→ 2

```java
public int MaxSum(int[] a, int n) {
    // Write your code here
    int maxSumSubarray = Integer.MIN_VALUE;
    int minSumSubarray = Integer.MAX_VALUE;
    int total = 0;

    int csummax = 0;
    int csummin = 0;

    for(int i = 0;i<n;i++){
        //add a[i] to total
        total += a[i];

        //tasks for calculating maxSumSubarray
        if(csummax < 0) csummax = a[i];
        else csummax+=a[i];
        maxSumSubarray = Math.max(maxSumSubarray,csummax);

        //tasks for calculating minSumSubarray
        if(csummin>0) csummin = a[i];
        else csummin += a[i];
        minSumSubarray = Math.min(minSumSubarray,csummin);
    }

    //if all elements are +ve
    if(total == maxSumSubarray) return maxSumSubarray;
    ...
    //if all elements are -ve
    if(total == minSumSubarray) return maxSumSubarray;
    ...
    //normal condition vs wrap around
    int normalCase = maxSumSubarray;
    int wrapAround = total - minSumSubarray;
    return Math.max(normalCase,wrapAround);
}
```

$-\infty \Rightarrow 8$

$\infty \Rightarrow -10$

$\sum_{i} a[i] = -1$

-1	1	5	-10	2	6	-2

Kadane's algo

Max sub subarry

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
2	-10	-5	-20	6	2	-1	4	-2	3	-1	-7	2											
0	2	-8		5	-15		6	3	2	11	9	10	5	-3	6								

Max(0, Max)

```
int max1 = -∞
int train = 0
for(int i=0;i<n) i++){
    if(train<0){train = a[i]}
    else{train += a[i]}
    max1 = max(max, train)
}
return max
```

12

$\Rightarrow -1$

-2	-10	-1	-20	-15		
0	-2	-10	-1	-20	-15	

1	15	10		
2	13	9		
3	X	7		
4	X	5		
5	.	7		

```java
public long maxSum(int grid[][]) {
    // Your code here
    int n = grid[0].length;
    // prefix sum array in the 0th row
    for(int i = 0;i<n;i++){
        grid[0][i] = grid[0][i-1];
    }
    //prefix sum array on the 1th row
    for(int i = 1;i<n;i++){
        grid[1][i] = grid[1][i-1];
    }
    //start exploring all possible paths
    //int overallAns = Integer.MAX_VALUE;
    for(int i = n;i>=1;i--){
        //if i==n if equal i mirroring row on the 1th col
        //options for robot 1
        int firstOption = i-1 == n ? 0 : grid[1][i-1];
        int secondOption = i-1 == 0 ? 0 : grid[0][i-1];
        int robot1choice = Math.max(firstOption,secondOption);
        overallAns = Math.min(overallAns,robot1choice);
    }
    return overallAns;
}
```

$n = 5$

Mirror in mirror that is idx

	0	1	2	3	4
0	4	1	12	5	2
1	2	1	1	3	4

$\Rightarrow$ 9, 5

5

$-\infty$

$i = 0$

Subarry Sum $< k$

$k = 3$

3	1	4	-3	5	5	1	6
3	4	8	5	10	15	16	22

$n - 0 = 4$

$n - k$

$1 + 0$

Mirror

15	16	9		
4	5	3	10	

ceil(a[i] - k)

Max Sum Circular array

-1	1	5	-10	2	
-1	0	5	-10	-1	-5

thisMax

thisMin

total = 0

Max % = 6

Min % = -10

6 vs 10

MSS vs | Wrap around

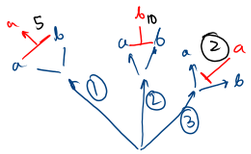
total sum − MSS

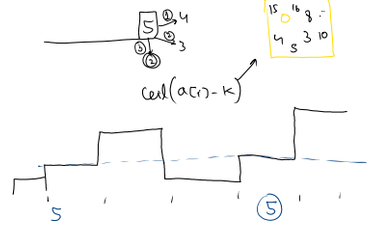
MSS

total − MinSum subarray

max(MSS, total − MinSS)

Min (0 ms, Max (grid[0][i+1], grid[1][i-1]))





k =

K = 3

k

1	2	3	4	-2	-2	10		
1	3	6	10	8	6			

0

8

1	6	
	3	10

k = 3

0

2

1

2

2 − 3 ⇒ -1

1	10	15	17	12	13	
		0				

15 → 14

12 13

15 − 3 ⇒ 12

ceil(15 − 3)

ceil(12)

12

3

5 6 10

5 10 6

1D arr

20

30

70

50

60

$O(n^2 \log n)$

n^2

$n \log$

Min Sum Sor = total

-5	-10	-2	→	-40
0	-5	-10	-15	
		-15		

② k = 5

5	2	3	4
1	1	0	3
8	6	2	1