Chapter 4

4.1

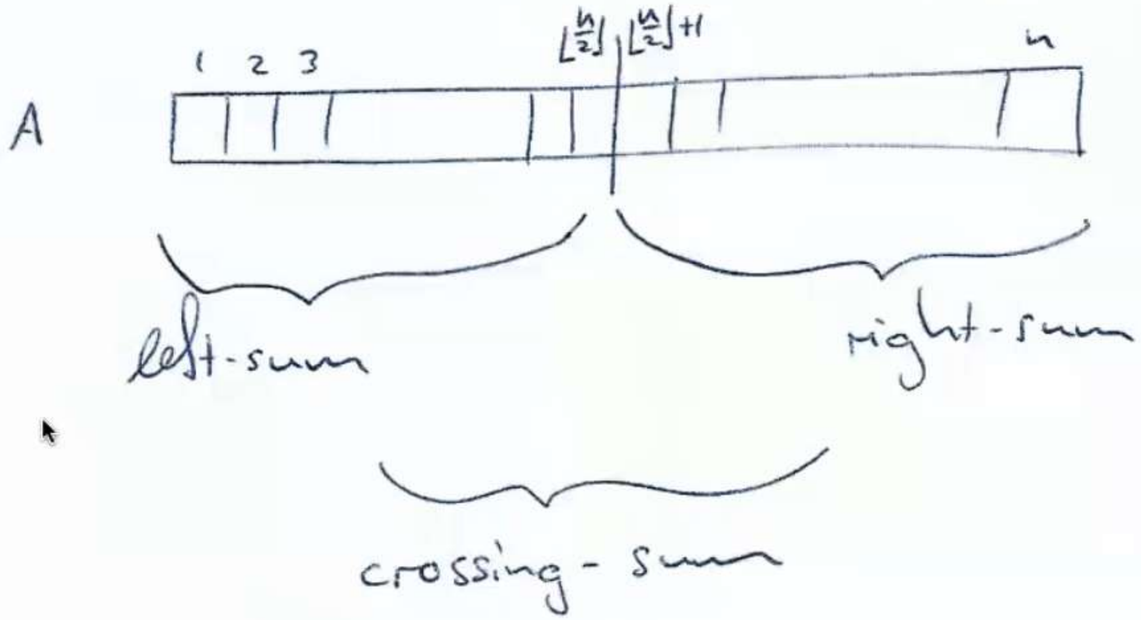
	1	2	3	4	5	6	7	8	9	10	11
A	5	-3	-4	7	-5	-1	3	2	-4	3	0

Problem: Find maximum contiguous subarray of an array.

Brute force approach:

for $i = 1$ to n
    for $j = i$ to n
        calculate $A[i, \ldots, j]$
        & update max
        if necessary

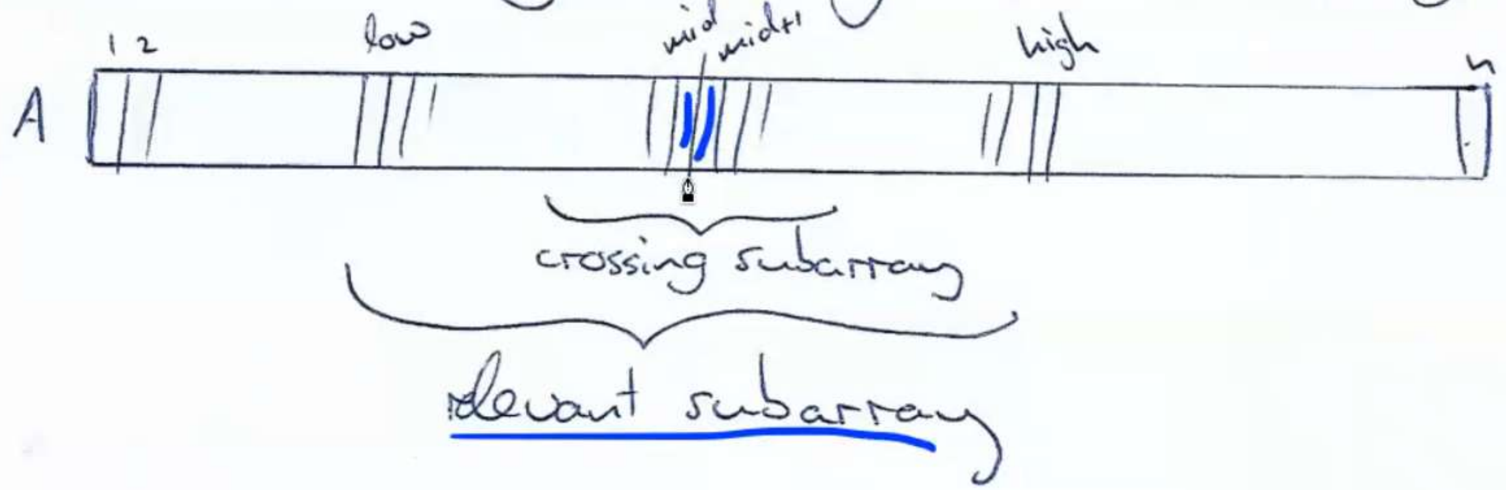
Divide-and-Conquer approach:



```
Find-Max-Subarray( A, low, high )
 if high = low    return A[low]
 else   mid = ⌊(low + high)/2⌋
          left-sum = Find-Max-Subarray(A, low, mid)
          right-sum = Find-Max-Subarray (A, mid+1, high)
          cross-sum = Find-Max-Crossing-Subarray
                              (A, low, mid, high)

          Return Max(left-sum, right-sum, cross-sum)
```

Find-Max-Crossing-Subarray (A, low, mid, high)



left-sum = $-\infty$
sum = 0
for i from mid downto low
    sum = sum + A[i]

```
left-sum = -∞
sum = 0
for i from mid downto low
        sum = sum + A[i]
        if sum > left-sum
                left-sum = sum
```

right-sum = $-\infty$
sum = 0
for j = mid+1 to high
    sum = sum + A[j]
    if sum > right-sum
        right-sum = sum

Return left-sum + right-sum

A recurrence for Find-Max-Subarray

Let $T(n)$ be the running time of Find-Max-Subarray on a subarray of size n (i.e, $1+\text{high}-\text{low} = n$)

Then $T(1) = \Theta(1)$ (constant time)

If $n > 1$ then

$$T(n) = T\left(\frac{n}{2}\right) + T\left(\frac{n}{2}\right) + \Theta(n) + \Theta(1)$$

$$= 2T\left(\frac{n}{2}\right) + cn + d$$

If $n > 1$ then

$$\begin{aligned} T(n) &= T\left(\frac{n}{2}\right) + T\left(\frac{n}{2}\right) + \Theta(n) + \Theta(1) \\ &= 2T\left(\frac{n}{2}\right) + cn + d \\ &= 2T\left(\frac{n}{2}\right) + cn \end{aligned}$$

$$T(n) = \begin{cases} \Theta(1) & \text{if } n = 1 \\ 2T\left(\frac{n}{2}\right) + cn & \text{if } n > 1 \end{cases}$$