```
FIND-MAXIMUM-SUBARRAY(A, low, high)
  if high == low
      return (low, high, A[low])          // base case: only one element
  else mid = ⌊(low + high)/2⌋
      (left-low, left-high, left-sum) =
          FIND-MAXIMUM-SUBARRAY(A, low, mid)
      (right-low, right-high, right-sum) =
          FIND-MAXIMUM-SUBARRAY(A, mid + 1, high)
      (cross-low, cross-high, cross-sum) =
          FIND-MAX-CROSSING-SUBARRAY(A, low, mid, high)
      if left-sum ≥ right-sum and left-sum ≥ cross-sum
          return (left-low, left-high, left-sum)
      elseif right-sum ≥ left-sum and right-sum ≥ cross-sum
          return (right-low, right-high, right-sum)
      else return (cross-low, cross-high, cross-sum)
```

```
FIND-MAX-CROSSING-SUBARRAY(A, low, mid, high)
  // Find a maximum subarray of the form A[i .. mid].
  left-sum = −∞
  sum = 0
  for i = mid downto low
      sum = sum + A[i]
      if sum > left-sum
          left-sum = sum
          max-left = i
  // Find a maximum subarray of the form A[mid + 1 .. j].
  right-sum = −∞
  sum = 0
  for j = mid + 1 to high
      sum = sum + A[j]
      if sum > right-sum
          right-sum = sum
          max-right = j
  // Return the indices and the sum of the two subarrays.
  return (max-left, max-right, left-sum + right-sum)
```