Maximum sum subarray

The maximum sum subarray problem consists in finding the maximum sum of a contiguous subsequence in an array or list of integers:

```
max_sequence([-2, 1, -3, 4, -1, 2, 1, -5, 4])
# should be 6: [4, -1, 2, 1]
```

Easy case is when the list is made up of only positive numbers and the maximum sum is the sum of the whole array. If the list is made up of only negative numbers, return 0 instead.

Empty list is considered to have zero greatest sum. Note that the empty list or array is also a valid sublist/subarray.

ALGORITHMS

LISTS

DYNAMIC PROGRAMMING

FUNDAMENTALS

PERFORMANCE

```
def max_sequence(arr):
    if arr == []:
        return 0

    max_sum = float('-inf')
    current_sum = 0

    for num in arr:
        current_sum += num
        if max_sum < current_sum:
            max_sum = current_sum
            if current_sum < 0:
                current_sum = 0

    # Check if max_sum is still -inf (meaning all elements were
    if max_sum == float('-inf'):
        return 0

    return max_sum
```

KADANE Algorithm