The **Kadane's algorithm** is a well-known algorithm used to find the **maximum subarray sum** in an array of numbers. It is efficient, with a time complexity of **O(n)**. However, there are several variations of this problem that can be solved using similar techniques. Below are some common variations of the Kadane's algorithm problem:

---

1. Maximum Subarray Sum

- **Problem**: Given an array of integers, find the contiguous subarray with the maximum sum.
- **Example**:
  - Input: `[-2, 1, -3, 4, -1, 2, 1, -5, 4]`
  - Output: `6` (from subarray `[4, -1, 2, 1]`)
- **Solution**: Use Kadane's algorithm.

---

2. Maximum Subarray Sum with At Least K Elements

- **Problem**: Find the maximum subarray sum where the subarray must contain at least `k` elements.
- **Example**:
  - Input: `[1, 2, 3, -4, 5]`, `k = 3`
  - Output: `6` (from subarray `[1, 2, 3]`)
- **Solution**: Modify Kadane's algorithm to track the sum of the first `k` elements and then iterate through the array to find the maximum sum.

---

3. Maximum Subarray Sum with Circular Array

- **Problem**: Given a circular array, find the maximum subarray sum. A circular array means the end of the array wraps around to the beginning.
- **Example**:
  - Input: `[5, -3, 5]`
  - Output: `10` (from subarray `[5, -3, 5]` wrapping around)
- **Solution**: Use Kadane's algorithm twice:
  1. Find the maximum subarray sum in the linear array.
  2. Find the minimum subarray sum and subtract it from the total sum of the array to handle the circular case.

---

4. Maximum Subarray Product

- **Problem**: Given an array of integers, find the contiguous subarray with the maximum product.
- **Example**:
  - Input: $[2, 3, -2, 4]$
  - Output: 6 (from subarray $[2, 3]$)
- **Solution**: Modify Kadane's algorithm to track both the maximum and minimum product at each step (since multiplying two negative numbers can yield a positive result).

---

5. Maximum Subarray Sum with No Adjacent Elements

- **Problem**: Find the maximum sum of a subarray where no two elements are adjacent.
- **Example**:
  - Input: $[3, 2, 7, 10]$
  - Output: 13 (from subarray $[3, 10]$)
- **Solution**: Use dynamic programming instead of Kadane's algorithm.

---

6. Maximum Subarray Sum with Size Constraint

- **Problem**: Find the maximum subarray sum where the subarray size must be exactly k.
- **Example**:
  - Input: $[1, 4, 2, 10, 2, 3, 1, 0, 20]$, $k = 4$
  - Output: 17 (from subarray $[4, 2, 10, 1]$)
- **Solution**: Use a sliding window approach.

---

7. Maximum Subarray Sum with Alternating Elements

- **Problem**: Find the maximum subarray sum where the elements alternate between positive and negative.
- **Example**:
  - Input: $[1, -2, 3, -4, 5]$
  - Output: 5 (from subarray $[5]$)
- **Solution**: Modify Kadane's algorithm to track alternating sequences.

---

8. Maximum Subarray Sum with Non-Negative Constraint

- **Problem**: Find the maximum subarray sum where all elements in the subarray are non-negative.
- **Example**:
  - Input: $[1, -2, 3, 4, -1, 2, 1, -5, 4]$
  - Output: 7 (from subarray $[3, 4]$)
- **Solution**: Modify Kadane's algorithm to reset the sum when a negative number is encountered.

---

9. Maximum Subarray Sum with Even or Odd Length

- **Problem**: Find the maximum subarray sum where the subarray length is even or odd.
- **Example**:
  - Input: $[1, 2, 3, 4, 5]$, even length
  - Output: 7 (from subarray $[3, 4]$)
- **Solution**: Modify Kadane's algorithm to track sums for even and odd lengths separately.

---

10. Maximum Subarray Sum with Minimum Length

- **Problem**: Find the maximum subarray sum where the subarray length is at least L.
- **Example**:
  - Input: $[1, 2, 3, -4, 5]$, $L = 2$
  - Output: 6 (from subarray $[1, 2, 3]$)
- **Solution**: Use a sliding window approach combined with Kadane's algorithm.

---

11. Maximum Subarray Sum with Maximum Length

- **Problem**: Find the maximum subarray sum where the subarray length is at most L.
- **Example**:
  - Input: $[1, 2, 3, -4, 5]$, $L = 2$
  - Output: 5 (from subarray $[3, -4, 5]$)
- **Solution**: Use a sliding window approach.

---

12. Maximum Subarray Sum with Exactly K Elements

- **Problem**: Find the maximum subarray sum where the subarray length is exactly k.
- **Example**:

    - Input: $[1, 4, 2, 10, 2, 3, 1, 0, 20]$, $k = 3$
    - Output: 16 (from subarray $[2, 10, 4]$)
- **Solution**: Use a sliding window approach.

These variations demonstrate the flexibility of Kadane's algorithm and its adaptability to different constraints. Let me know if you'd like the implementation for any of these variations!