Given an array and find the maximum continuous sum in linear time

This problem is highly non trivial

1. Let's try the naiver solution which is O(n^3)
   Array = [-3, 4, 5, -8]

   -3
   -3, 4 4
   -3, 4, 5 4, 5 4
   -3, 4, 5, 8 4, 5, 8 4, 5 5

   Sum all the rows and find the maximum

```
int max = Int.min;
int len = Array.length;
for(int b = 0; b < len; b++)
{
   for(int i=0; i<len; i++)
   {
      sum = 0;
      for(int k = i; k < i+1; k++)
      {
         sum += Array[k+b];
      }
      if(max < sum)
         max = sum;
   }
}
```

   The algorithm has three loops and it is easy to analysis and show
   the run time is O(n^3)

2. Let's find a more efficient algorithm.
   From the left to the right of picture bellow

   (1) If the Array has one element: Array = [-1]
   then it is trivial case, max = -1

   (2) If the Array has two elements: Array = [2, -1]
   then max = max(previous_max + 2, 2)

   (3) If the Array has three elements: Array = [-3, 2, -1]
   then max = max(previous_max + (-3), -3)

   (4) If the Array has four elements: Array = [4, -3, 2, -1]
   then max = max(previous_max + (4), 4)

   (5) If the Array has five elements: Array = [-2, 4, -3, 2, -1]
   then max = max(previous_max + (-2), -2)

   Full Algorithm 1: The algorithm is based on the staircase box bellow

```
len = Array.length;
if( len == 1)
  return max = Array[0];
else if(len > 1)
{
  int[ ] box = new int[len];
  box[0] = Array[0];
  for(int k = 1; k < len; k++)
     box[k] = box[k-1] + Array[k] > Array[k] ? box[k-1] + Array[k] : Array[k];

     max = box[0];
        for(int k = 1; k < len; k++)
          if(max < box[k]) max = box[k];
}
return max
```

   Full Algorithm 2: The algorithm will work only there are negative and positive/zero integer in the Array

```
max = Array[0]
sum = max;
for(int k=1; k<len; k++)
{
  if(sum < 0)                  (a)
     sum = 0;                  (b)
  sum += Array[k];
  if(max < sum)
    max = sum;
}
 return max
```

   Why the algorithm work?
   From left to right of staircase box:
      if the max value of a box is negative,
         then the max value of next staircase box(let the box is box[x]) (from left to right)
               array[x-1] + box[k] < array[x-1] (since box[x] is negative)
               so array[x-1] + box[k] will never be the max value of continuous sum
               and the next possible max value will be start from array[x-1] inclusively
   This is where the code (a) and (b) does