মনে করো তোমাকে একটা অ্যারে দেয়া হয়েছে যেখানে nn টা সংখ্যা আছে। তোমাকে বলা হলো সেই অ্যারের m=৩ আকারের যতগুলো সাবঅ্যারে আছে সবগুলো থেকে সবথেকে ছোটো সংখ্যাটা বের করতে হবে।
যেমন অ্যারেটা যদি হয় ১০,২,৫,৯,৬,৪ তাহলে m=৩ সাইজের সবগুলো সাবঅ্যারে হলো:

১০,২,৫ , সর্বনিম্ন সংখ্যা ২
২,৫,৯ , সর্বনিম্ন সংখ্যা ২
৫,৯,৬, সর্বনিম্ন সংখ্যা ৫
৯,৬,৪ , সর্বনিম্ন সংখ্যা ৪

তাহলে তোমার আউটপুট হবে [২,৫,৫,৪]।

mm এর মান ৩ না হয়ে ১ থেকে nn পর্যন্ত যেকোনো সংখ্যা হতে পারে।
nn এর মান যদি ছোটো হয় তাহলে আমরা সহজেই প্রতিটা সাবঅ্যারের উপর লুপ চালিয়ে সমস্যাটা সমাধান করতে পারি। নিচের পাইথন কোডটি দেখো:

```
def brute_rmq(arr,m):
        res=[]
        for i in range(0,len(arr)-m+1):
                subarr=arr[i:i+m] #take subarray of size m, starting from index i
                res.append(min(subarr)) #append the minimum element in result
        return res
```

এই কোডের কমপ্লিক্সিটি $O(n^2)O(n2)$।
আমরা O(nlogn)O(nlogn) এ সমস্যাটা সমাধান করতে পারি সেগমেন্ট ট্রি ব্যবহার করে।
স্লাইডিং উইন্ডো এবং মনোটোনাস ডিকিউ ব্যবহার করে সমস্যাটা O(n)O(n) কমপ্লেক্সিটিতে সমাধান করা সম্ভব, সেটাই আজকে আমরা শিখবো। মনোটোনাস ডিকিউ বা ডাবল-এন্ডেড-কিউ হলো এমন একটা ডিকিউ যেখানে সংখ্যাগুলো সবসময় সর্টেড থাকে।
মনে করি অ্যারেতে সংখ্যাগুলো হলো [১০,৫০,১৫,১২,৪] এবং m=৩।

আমরা বাম থেকে ডানে একটা একটা সংখ্যা নিয়ে কাজ করতে থাকবো। আমরা সংখ্যাগুলোকে এমনভাবে ডিকিউতে ঢুকাবো যেন সবথেকে ছোটো সংখ্যাটা সবসময় সবার ডানে থাকে। i তম ইনডেক্সে যখন থাকবো তখন (i−m+1,i)(i−m+1,i) সাবঅ্যারের সর্বনিম্ন সংখ্যাটাকে ডিকিউর সবথেকে ডানে পাওয়া যাবে।
প্রথম সংখ্যাটা হলো ১০, এটাকে আমরা ডিকিউ তে বামদিক থেকে ঢুকাবো:

DQ=[১০]

পরের সংখ্যাটা হলো ৫০, এটাকেও বামদিক থেকে ঢুকাবো:

DQ=[৫০,১০]

পরের সংখ্যাটা হলো ১৫। এখন লক্ষ্য করো, এখন পর্যন্ত যতগুলো সংখ্যা পেয়েছি তাদের মধ্যে যারা ১৫ এর থেকে বড় তারা কখনোই সর্বনিম্ন সংখ্যা হতে পারবে না, কারণ তারা ১৫ এর বামে আছে এবং তারা যে সাবঅ্যারেতে আছে সেগুলোতে ১৫ ও অবশ্যই আছে। এটা বোঝাই অ্যালগোরিদমের সবথেকে গুরুত্বপূর্ণ অংশ। **কোনো একটা সংখ্যা ডিকিউতে ঢুকানোর আগে সেই সংখ্যাটার থেকে যতগুলো বড় সংখ্যা ডিকিউতে আছে সেগুলো বের করে দিতে হবে।**
DQ=[১৫,১০]

তাহলে প্রথম ৩ আকারের সাবঅ্যারে [১০,৫০,১৫] এ সর্বনিম্ন সংখ্যা হলো ডিকিউ এর সর্বডানের সংখ্যা ১০।

পরের সংখ্যাটা হলো ১২। তাহলে আমরা ১৫ কে ফেলে দিয়ে ১২ কে ঢুকাবো।

DQ=[১২,১০]

লক্ষ্য করো আমরা এখন i=3 নম্বর ইনডেক্সে আছি এবং i-m+1=৩-৩+১=১ নম্বর ইনডেক্সের বামের কোনো সংখ্যা আমাদের দরকার নেই কারণ সেগুলো রেঞ্জের বাইরে। কিউ এর সবার ডানের সংখ্যা ১০ মূল অ্যারের এর ০ তম ইনডেক্সে অবস্থিত, সেটাকে আমরা ফেলে দিতে পারি।

DQ=[১২]

তাহলে ২য় ৩ আকারের সাবঅ্যারে [৫০,১৫,১২] এ সর্বনিম্ন সংখ্যা হলো ডিকিউ এর সর্বডানের সংখ্যা ১২।

পরবর্তি সংখ্যাটা হলো ৪। আমরা ১২ ফেলে দিয়ে ৪ ঢুকাবো:

DQ=[৪]

তাহলে ৩য় ৩ আকারের সাবঅ্যারে [১৫,১২,৪] সর্বনিম্ন সংখ্যা হলো ডিকিউ এর সর্বডানের সংখ্যা ৪।

তাহলে O(n)O(n) কমপ্লিক্সিটিতে আমরা সবগুলো রেঞ্জের সর্বনিম্ন সংখ্যাগুলো বের করে ফেললাম।
নিচের পাইথন কোডে উপরের অ্যালগোরিদমটা ইমপ্লিমেন্ট করা হয়েছে। পাইথন না জানলেও বুঝতে সমস্যা হবে না:

```
def sliding_rmq(arr, m):

    DQ = deque()
    res=[]
    for i,val in enumerate(arr):
        while len(DQ) and DQ[0][0]>=val: #DQ[0][0] is the leftmost element of DQ
            DQ.popleft()্
        DQ.appendleft((val,i)) #pushing a pair containing the value and the index

        while len(DQ) and DQ[-1][1]<=i-m: #DQ[-1][1] is the index of the rightmost element of DQ
            DQ.pop() #popping the out-of-range elements

        if i>=m-1: #We got a m size range
            print DQ[-1][0] #print the rightmost element of DQ
            res.append(DQ[-1][0])

    return res
```

চিন্তা করার জন্য সমস্যা:

১. মনে করো তোমাকে nn টা সংখ্যা এবং qq টা রেঞ্জ দেয়া হয়েছে, রেঞ্জগুলো হলো $[a_1,b_1],[a_2,b_2]$.......$[a_q,b_q]$[a1,b1],[a2,b2].......[aq,bq] এবং প্রতিটা i<qi<q এর জন্য $a_i \leq a_{i-1}$ai≤ai−1 এবং $b_i \leq b_{i-1}$bi≤bi−1। প্রতিটা রেঞ্জের সর্বনিম্ন সংখ্যা বের করতে হবে। কিভাবে করবে?

২. http://www.spoj.com/problems/PARSUMS/