

```

81 Manacher's Algorithm for numbers:
82 struct Manacher{
83     int n;
84     vector<int> d1, d2 , v;
85
86     Manacher(vector<int> v) : v(v)
87     {
88         n = v.size();
89         d1.resize(n);
90         d2.resize(n);
91         // d1
92         for (int i = 0, l = 0, r = -1; i < n; i++)
93         {
94             int k = (i > r) ? 1 : min(d1[l + r - i], r - i + 1);
95             while (0 ≤ i - k && i + k < n && v[i - k] == v[i + k])
96             {
97                 k++;
98             }
99             d1[i] = k--;
100             if (i + k > r)
101             {
102                 l = i - k;
103                 r = i + k;
104             }
105         }
106         // d2
107         for (int i = 0, l = 0, r = -1; i < n; i++)
108         {
109             int k = (i > r) ? 0 : min(d2[l + r - i + 1], r - i + 1);
110             while (0 ≤ i - k - 1 && i + k < n && v[i - k - 1] == v[i + k])
111             {
112                 k++;
113             }
114             d2[i] = k--;
115             if (i + k > r)
116             {
117                 l = i - k - 1;
118                 r = i + k;
119             }
120         }
121     }
122
123     //check if subArray is palindrome O(1)
124     bool isPal(int l, int r)
125     {
126         int len = r - l + 1;
127         int i = l + r >> 1;
128         if (len % 2)
129             return d1[i] > len / 2;
130         else
131             return d2[i + 1] ≥ len / 2;
132     }
133
134     //get the number of palindrome subArray in array O(n)
135     ll numberOfPal(){
136         ll even = accumulate(d1.begin() , d1.end() , 0LL);
137         ll odd = accumulate(d2.begin() , d2.end() , 0LL);
138         ll count = even + odd;
139         return count;
140     }

```