Manacher's Algorithm for numbers:

```cpp
struct Manacher{
    int n;
    vector<int> d1, d2 , v;

    Manacher(vector<int> v) : v(v)
    {
        n = v.size();
        d1.resize(n);
        d2.resize(n);
        // d1
        for (int i = 0, l = 0, r = -1; i < n; i++)
        {
            int k = (i > r) ? 1 : min(d1[l + r - i], r - i + 1);
            while (0 <= i - k && i + k < n && v[i - k] == v[i + k])
            {
                k++;
            }
            d1[i] = k--;
            if (i + k > r)
            {
                l = i - k;
                r = i + k;
            }
        }
        // d2
        for (int i = 0, l = 0, r = -1; i < n; i++)
        {
            int k = (i > r) ? 0 : min(d2[l + r - i + 1], r - i + 1);
            while (0 <= i - k - 1 && i + k < n && v[i - k - 1] == v[i + k])
            {
                k++;
            }
            d2[i] = k--;
            if (i + k > r)
            {
                l = i - k - 1;
                r = i + k;
            }
        }
    }

    //check if subArray is palindrome O(1)
    bool isPal(int l, int r)
    {
        int len = r - l + 1;
        int i = l + r >> 1;
        if (len % 2)
            return d1[i] > len / 2;
        else
            return d2[i + 1] >= len / 2;
    }

    //get the number of palindrome subArray in array O(n)
    ll numberOfPal(){
        ll even = accumulate(d1.begin() , d1.end() , 0LL);
        ll odd = accumulate(d2.begin() , d2.end() , 0LL);
        ll count = even + odd;
        return count;
    }
```