单调队列优化dp

人员

王梓同、于潇涵、石宇赫、胡赫轩、崔嘉睿、穆鹏宇、程晟泰、梁钰涵、周子航、蔡云翔 到课

作业检查

王梓同 未完成

于潇涵 未完成

石宇赫 已完成

李佳声 未完成

窦浩轩 未完成

胡赫轩 已完成

崔嘉睿 已完成

穆鹏宇 已完成

程晟泰 未完成

梁钰涵 已完成

周子航 未完成

蔡云翔 未完成

作业

https://www.luogu.com.cn/contest/175939

D、G 题目

课堂表现

课堂默写中，只有 石宇赫 和 崔嘉睿 两位同学在限时时间内默写出来，提出表扬！！

其他同学要下不为例，课下一定要好好复习课上内容。

课堂内容

P4999 烦人的数学作业

```
#include <bits/stdc++.h>

using namespace std;
```

```cpp
typedef long long LL;
const int maxn = 20;
const int mod = 1e9+7;
int f[maxn][maxn];  // f[i][j]: 第 i 位填 j 时，总数字和是多少

int wPow(int a, int k) {
  int res = 1;
  while (k -- ) res = (LL)res*a % mod;
  return res;
}

void init() {
  for (int i = 0; i <= 9; ++i) f[1][i] = i;
  for (int i = 2; i < maxn; ++i) {
    for (int j = 0; j <= 9; ++j) {
      for (int k = 0; k <= 9; ++k) f[i][j] = (f[i][j] + f[i-1][k]) % mod;
      f[i][j] = (f[i][j] + (LL)j*wPow(10, i-1)) % mod;
    }
  }
}

int calc(vector<int>& nums, int pos) {
  int res = 0;
  for (int i = pos; i >= 0; --i) res = ((LL)res*10 + nums[i]) % mod;
  return res;
}

int dp(LL n) {
  vector<int> nums;
  while (n) nums.push_back(n%10), n /= 10;

  int res = 0;
  for (int i = (int)nums.size()-1; i >= 0; --i) {
    int x = nums[i];
    for (int j = 0; j < x; ++j) res = (res + f[i+1][j]) % mod;
    int t = calc(nums, i-1) + 1;
    res = (res + (LL)x*t) % mod;
  }
  return res;
}

int main()
{
  init();
  int T; cin >> T;
  while (T -- ) {
    LL l, r; cin >> l >> r;
    cout << (dp(r) - dp(l-1) + mod) % mod << endl;
  }
  return 0;
}
```

P2034 选择数字

对于形如 f[i] = max{f[j]+w[j]} + w[i] (i-k<=j<=i) 样子的式子，我们都可以采用单调队列进行优化

```
#include <bits/stdc++.h>

using namespace std;

typedef long long LL;
const int maxn = 1e5 + 5;
int w[maxn];
LL pSum[maxn], f[maxn][2];

LL calc(int pos) { return f[pos][0] - pSum[pos]; }

int main()
{
  int n, k; cin >> n >> k;
  for (int i = 1; i <= n; ++i) {
    cin >> w[i]; pSum[i] = pSum[i-1] + w[i];
  }

  deque<int> q; q.push_back(0);
  for (int i = 1; i <= n; ++i) {
    while (!q.empty() && q.front()<i-k) q.pop_front();
    f[i][0] = max(f[i-1][0], f[i-1][1]);
    f[i][1] = calc(q.front()) + pSum[i];
    while (!q.empty() && calc(i)>=calc(q.back())) q.pop_back();
    q.push_back(i);
  }
  cout << max(f[n][0], f[n][1]) << endl;
  return 0;
}
```

P1886 滑动窗口 /【模板】单调队列

```
#include <bits/stdc++.h>
using namespace std;

const int maxn = 1e6 + 6;
int w[maxn];

void solve_min(int n, int m) {
    deque<int> q;
    for (int i = 1; i <= n; ++i) {
        while (!q.empty() && i-q.front()>=m) q.pop_front();
        while (!q.empty() && w[i]<=w[q.back()]) q.pop_back();
        q.push_back(i);
        if (i >= m) cout << w[q.front()] << " ";
    }
```

```
    cout << endl;
}

void solve_max(int n, int m) {
    deque<int> q;
    for (int i = 1; i <= n; ++i) {
        while (!q.empty() && i-q.front()>=m) q.pop_front();
        while (!q.empty() && w[i]>=w[q.back()]) q.pop_back();
        q.push_back(i);
        if (i >= m) cout << w[q.front()] << " ";
    }
    cout << endl;
}

int main() {
    ios::sync_with_stdio(false);
    cin.tie(0);
    int n, m; cin >> n >> m;
    for (int i = 1; i <= n; ++i) cin >> w[i];
    solve_min(n, m); solve_max(n, m);
    return 0;
}
```

全排列

```
#include <bits/stdc++.h>

using namespace std;

int a[] = {0, 3, 4, 2, 5, 1};

int main()
{
    sort(a+1, a+6);
    do {
        for (int i = 1; i <= 5; ++i) cout << a[i] << " ";
        cout << endl;
    } while (next_permutation(a+1, a+6));
    return 0;
}
```

求组合数

```
#include <bits/stdc++.h>

using namespace std;

const int maxn = 1000 + 5;
```

```
const int mod = 998244353;
int c[maxn][maxn];

int main()
{
    for (int i = 0; i < maxn; ++i) {
        for (int j = 0; j <= i; ++j) {
            if (!j) c[i][j] = 1;
            else c[i][j] = (c[i-1][j-1] + c[i-1][j]) % mod;
        }
    }
    return 0;
}
```

二分模板

要学会区分 左0右1 和 左1右0 两种情况

```
while (l <= r) {
    int mid = (l + r) / 2;
    if (check(mid)) {
        l = mid+1;
        // r = mid-1;
    }
    else {
        r = mid-1;
        // l = mid+1;
    }
}
cout << r << endl;
// cout << l << endl;
```