综合混练

人员

刘锦轩 到课

上周作业检查

上上周作业链接: https://vjudge.net/contest/722784

Begin: 2025-06-14 08:30 CST ☆ 2025-0614 五队上课 (综合混练) **End:** 2026-01-08 16:30 CST

Elapsed: 14:00:03:50 Running **Remaining:** 194:07:56:09

Overview | Problem | Status | Rank (14:00:03:42) | Discuss | Setting | Clone | Update | Delete

Rank	Team	Score	Penalty	A 12 / 25	B 4 / 4	C 1 / 1	D 2 / 2	E 0 / 7
1	☆ sty0948 (隋...)	4	54828	6:12:14:55	10:12:24:02	10:12:26:26	10:12:43:06	(-2)
2	☆ two_tiger (卢炫佑)	3	56361	12:11:43:50 (-7)	13:02:19:07		13:10:58:54	
3	☆ misaka16384 (黄诗琦)	2	25684	7:08:15:33 (-1)	10:11:28:55			
4	☆ FeatherCrow (许岩)	2	39149	13:14:08:49	13:14:21:01			
5	☆ fbl123bc (付丙霖)	1	10430	7:05:30:24 (-1)				(-1)
6	☆ zhn123bc (张皓宁)	1	10532	7:07:12:05 (-1)				
7	☆ exLucas (范家畅)	1	10536	7:07:36:02				(-2)
8	☆ ccx123bc (曹承贤)	1	10553	7:07:33:02 (-1)				
9	☆ Hanhj (韩鸿钜)	1	10573	7:08:13:21				
10	☆ lzy123bc (刘智予)	1	10580	7:08:00:46 (-1)				
11	☆ ikunTLE (方冠霖)	1	10895	7:13:35:48				
12	☆ lxr123bc (刘新睿)	1	19443	13:11:43:03 (-1)				
13	☆ longlong_int (刘锦轩)	0	0					(-2)

上周作业链接: https://vjudge.net/contest/724229

作业

https://vjudge.net/contest/725693 (课上讲了上周比赛的 A B C D E, 课后作业是本周比赛的 A B C D E 题)

课堂表现

今天的 5 道题整体难度会比较大一些, 同学们做起来应该会比较吃力, 课下一定要沉住气好好做一做这几道题。

课堂内容

CF1921F Sum of Progression

按照 项数和sqrt(n)大小关系 进行分组

项数<=sqrt(n), 直接循环求答案; 项数>sqrt(n), 预处理前缀和数组求答案

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

using namespace std;

typedef long long LL;
const int maxn = 1e5 + 5;
int w[maxn];
LL p[505][maxn], pV[505][maxn];

int get_up(int a, int b) { return (a+b-1)/b; }

LL get_sum(int l, int r, LL c[]) {
  if (l <= 0) return c[r];
  return (l<=r ? c[r]-c[l] : 0);
}

void solve() {
  int n, m; cin >> n >> m;
  for (int i = 1; i <= n; ++i) cin >> w[i];

  int limit = sqrt(n);
  for (int i = 1; i <= limit; ++i) {
    for (int j = 1; j <= n; ++j) {
      if (j-i < 1) pV[i][j] = p[i][j] = w[j];
      else {
        pV[i][j] = pV[i][j-i] + (LL)w[j]*get_up(j,i);
        p[i][j] = p[i][j-i] + w[j];
      }
    }
  }

  vector<LL> vec;
  while (m -- ) {
    int s, d, k; cin >> s >> d >> k;
    if (d > limit) {
      LL res = 0;
      for (int i = 1; i <= k; ++i) res += (LL)w[s+(i-1)*d]*i;
      vec.push_back(res);
    } else {
      int l = s, r = s + d * (k-1);
      LL res = get_sum(l - d, r, pV[d]) - get_sum(l - d, r, p[d])*(get_up(l,d)-1);
      vec.push_back(res);
    }
  }
```

```
//  cout << "----------------------- ";
  for (LL i : vec) cout << i << " "; cout << endl;
}

int main()
{
  int T; cin >> T;
  while (T -- ) solve();
  return 0;
}
```

CF1606E Arena

dp, 设 f[i][j]: 还剩 i 个人, 最大血量为 j 时, 一共有多少方案

转移方法: j < i 时: f[i][j] <- qmod(j,i)-qmod(j-1,i) j >= i 时: f[i][j] <- C[i][k] * f[k][j-i+1] * qmod(i-1, i-k), (1<=k<=i)

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

using namespace std;

typedef long long LL;
const int maxn = 500 + 5;
const int mod = 998244353;
int f[maxn][maxn], C[maxn][maxn];

int qmod(int a, int k) {
  int res = 1;
  while (k) {
    if (k&1) res = (LL)res*a%mod;
    a = (LL)a*a%mod;
    k >>= 1;
  }
  return res;
}

int main()
{
  for (int i = 0; i < maxn; ++i) {
    C[i][0] = 1;
    for (int j = 1; j <= i; ++j) C[i][j] = (C[i-1][j-1] + C[i-1][j]) % mod;
  }

  int n, x; cin >> n >> x;

  for (int i = 2; i <= n; ++i) {
    for (int j = 1; j <= x; ++j) {
      if (j < i) f[i][j] = (qmod(j,i)-qmod(j-1,i)+mod)%mod;
      else {
```

```
        for (int k = 1; k <= i; ++k) {
          f[i][j] = (f[i][j] + (LL)C[i][k]*f[k][j-i+1]%mod*qmod(i-1,i-k)%mod) %
mod;
        }
      }
    }
  }

  int res = 0;
  for (int i = 1; i <= x; ++i) res = (res + f[n][i]) % mod;
  cout << res << endl;
  return 0;
}
```

CF1353E K-periodic Garland

dp

f[i][0]: 以第 i 个数是 1 结尾, 同时第 i 个数是开头, 最少需要改多少个

f[i][1]: 以第 i 个数是 1 结尾, 同时第 i 个数不是开头, 最少需要改多少个

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

using namespace std;

const int maxn = 1e6 + 5;
const int inf = 0x3f3f3f3f;
char s[maxn];
int p[maxn], f[maxn][2];

int get_sum(int l, int r) { return p[r] - p[l-1]; }

void solve() {
  int n, k; cin >> n >> k;
  for (int i = 0; i <= n+2; ++i) f[i][0] = f[i][1] = inf;

  cin >> (s+1);
  for (int i = 1; i <= n; ++i) p[i] = p[i-1] + (s[i]=='1');

  int res = get_sum(1,n);
  for (int i = 1; i <= n; ++i) {
    f[i][0] = get_sum(1,i-1) + (s[i]!='1');
    if (i > k) f[i][1] = min(f[i-k][0],f[i-k][1]) + get_sum(i-k+1,i-1) +
(s[i]!='1');
    res = min(res, min(f[i][0],f[i][1])+get_sum(i+1,n));
  }

//  cout << "-------------------- ";
  cout << res << endl;
}
```

```
int main()
{
  int T; cin >> T;
  while (T -- ) solve();
  return 0;
}
```

CF1437E Make It Increasing

设 ci = ai - i, 原本要求 a 数组单调递增, 现在只需要 c 数组单调不降即可

针对 c 数组的每一段, 求包含左右端点情况下的最长不下降子序列

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

using namespace std;

const int maxn = 5e5 + 5;
const int inf = 0x3f3f3f3f;
int a[maxn], b[maxn];
int n;

int LIS(int lpos, int rpos) {
  int lvalue = a[lpos], rvalue = a[rpos];
  vector<int> vec;
  for (int i = lpos; i <= rpos; ++i) {
    if (a[i]<lvalue || a[i]>rvalue) continue;
    if (vec.empty() || a[i]>=vec.back()) vec.push_back(a[i]);
    else *upper_bound(vec.begin(), vec.end(), a[i]) = a[i];
  }
  return vec.size();
}

int main()
{
  int k; cin >> n >> k;
  for (int i = 1; i <= n; ++i) cin >> a[i], a[i] -= i;
  for (int i = 1; i <= k; ++i) cin >> b[i];
  for (int i = 2; i <= k; ++i) {
    if (a[b[i-1]] > a[b[i]]) { cout << -1 << endl; return 0; }
  }

  b[0] = 0, b[k+1] = n+1;
  a[0] = -inf, a[n+1] = inf;

  int res = 0;
  for (int i = 0; i <= k; ++i) res += (b[i+1]-b[i]+1) - LIS(b[i],b[i+1]);
  cout << res << endl;
```

```
  return 0;
}
```

U573082 交换球

设 f[i][0] 为 i 轮之后黑球在第一个位置的概率, f[i][1] 为 i 轮之后黑球在后续其它位置的概率

做 k 轮转移即可

```
#include <bits/stdc++.h>
#define int long long

using namespace std;

const int maxn = 1e5 + 5;
const int mod = 998244353;
int f[maxn][2];

int qmod(int a, int k) {
  int res = 1;
  while (k) {
    if (k&1) res = res*a % mod;
    a = a * a % mod;
    k >>= 1;
  }
  return res;
}

int inv(int x) { return qmod(x, mod-2); }

signed main()
{
  int n, k; cin >> n >> k;
  int val = inv(n*n%mod);

  f[0][0] = 1, f[0][1] = 0;
  for (int i = 1; i <= k; ++i) {
    int a = ((n-1)*(n-1)+1)%mod*val%mod;
    int b = (n-1)*2*val%mod;
    f[i][0] = (f[i-1][0]*a + f[i-1][1]*b) % mod;

    int c = 2*val%mod, d = (n-2)*2*val%mod, e = a;
    f[i][1] = (f[i-1][0]*c+f[i-1][1]*d+f[i-1][1]*e) % mod;
  }

//  cout << f[k][0] << " ---------------------- " << f[k][1] << endl;
  int res = (f[k][0] + f[k][1]*(n+2)%mod*(n-1)%mod*inv(2)%mod) % mod;
  cout << res << endl;
  return 0;
}
```