思考题讲解1

人员

于潇涵、石宇嫿、胡赫轩、崔嘉睿、穆鹏宇、程晟泰、梁钰涵、周子航、刘子淇 到课

作业检查

于潇涵 未完成

王梓同 未完成

蔡云翔 未完成

石宇嫿 已完成

李佳声 未完成

胡赫轩 已完成

崔嘉睿 未完成

穆鹏宇 未完成

程晟泰 已完成

梁钰涵 已完成

周子航 已完成

刘子淇 已完成

作业

https://www.luogu.com.cn/contest/178768, 5道题目要求补完

https://www.luogu.com.cn/contest/176989, 课后思考题，要求同学们课下思考完成

课堂表现

这节课上讲的题目有些多，课上没有给大家特别多的时间写代码，要求同学们课下编程实现课上的题目

要不然同学们只是听懂了，自己觉得自己会了，不一定是真的会了

课堂内容

T464543 rescue

二分 + 并查集/bfs，判断能否把 左上角 和 右下角 连起来

```cpp
#include <bits/stdc++.h>
#define x first
#define y second

using namespace std;

typedef pair<int, int> PII;
const int maxn = 3000 + 5;
PII w[maxn];
int f[maxn];

int fFind(int x) {
    if (f[x] != x) f[x] = fFind(f[x]);
    return f[x];
}

void fMerge(int x, int y) {
    int p = fFind(x), q = fFind(y);
    if (p != q) f[p] = q;
}

bool check(double mid, int n, int row, int line) {
    for (int i = 1; i <= n + 2; ++i) f[i] = i;
    for (int i = 1; i <= n; ++i) {
        if (w[i].x - mid < 1 || w[i].y + mid > line) fMerge(i, n+1);
        if (w[i].x + mid > row || w[i].y - mid < 1) fMerge(i, n+2);
    }
    for (int i = 1; i <= n; ++i) {
        for (int j = i + 1; j <= n; ++j) {
            int a = w[i].x - w[j].x, b = w[i].y - w[j].y;
            double d = sqrt(a*a + b*b);
            if (d < 2*mid) fMerge(i, j);
        }
    }
    if (fFind(n + 1) == fFind(n + 2)) return false;
    return true;
}

int main()
{
    int n, row, line; cin >> n >> row >> line;
    for (int i = 1; i <= n; ++i) cin >> w[i].x >> w[i].y;

    double l = 0, r = row + line;
    while (r - l > 1e-4) {
        double mid = (l + r) / 2;
        if (check(mid, n, row, line)) l = mid;
        else r = mid;
    }
    printf("%.2f\n", l);
    return 0;
}
```

T464551 number

状压dp

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

using namespace std;

const int N = 225, M = 15;
const int mod = 10007;
char s[N][N], t[N][N];
int f[N][1<<M];
int n, m;

char cValue(int x) { return x+'0'; }

bool check(int i, int k) {
  for (int j = 0; j < m; ++j) {
    if (s[i][j] == '.') continue;
    if (s[i][j] != cValue((k>>j)&1)) return false;
  }
  return true;
}

bool isOk(int m, int k) {
  for (int i = 0; i < m-1; ++i) {
    if (((k>>i)&1) && ((k>>(i+1))&1)) return false;
  }
  return true;
}

int main()
{
  cin >> n >> m;
  for (int i = 0; i < n; ++i) cin >> s[i];

  if (n < m) {
    for (int i = 0; i < n; ++i) {
      for (int j = 0; j < m; ++j) {
        t[j][n-i-1] = s[i][j];
      }
    }
    swap(n, m);
    memcpy(s, t, sizeof(s));
  }

  vector<int> vec;
  for (int i = 0; i < (1<<m); ++i) {
    if (isOk(m, i)) vec.push_back(i);
  }
```

```
  for (int i = 0; i < n; ++i) {
    for (int j : vec) {
      if (!check(i, j)) continue;
      if (i == 0) f[i][j] = 1;
      else {
        for (int k : vec) {
          if (j&k) continue;
          f[i][j] = (f[i][j] + f[i-1][k]) % mod;
        }
      }
    }
  }

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

T460086 swap

bfs, 求 [1,n] 这种状态到 [n,1] 这种状态的最短路

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

using namespace std;

const int maxn = 2000 + 5;
int f[maxn][maxn], w[maxn];
vector<int> vec[maxn];

struct node {
  int x, y;
};

int bfs(int n, int u, int v) {
  queue<node> q; q.push({u, v}); f[u][v] = 0;
  while (!q.empty()) {
    node p = q.front(); q.pop();
    int x = p.x, y = p.y;
    if (x==n && y==1) return f[n][1];
    for (int i : vec[x]) {
      for (int j : vec[y]) {
        if (w[i]!=w[j] && f[i][j]==-1) {
          q.push({i, j}); f[i][j] = f[x][y]+1;
        }
      }
    }
  }
  return -1;
}
```

```
void solve() {
  memset(f, -1, sizeof(f));
  for (int i = 0; i < maxn; ++i) vec[i].clear();
  int n, m; cin >> n >> m;
  for (int i = 1; i <= n; ++i) cin >> w[i];
  while (m -- ) {
    int u, v; cin >> u >> v;
    vec[u].push_back(v), vec[v].push_back(u);
  }

  cout << bfs(n, 1, n) << endl;
}

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

P1908 逆序对

在归并排序的过程中，求逆序对的数量

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

using namespace std;

typedef long long LL;
const int maxn = 5e5 + 5;
int w[maxn];
LL res = 0;

void mergeSort(int l, int r) {
    if (l >= r) return ;
    int mid = (l + r) / 2;
    mergeSort(l, mid); mergeSort(mid+1, r);
    // w[l]~w[mid], w[mid+1]~w[r]

    vector<int> vec;
    int i = l, j = mid+1;
    while (i<=mid && j<=r) {
        if (w[i] <= w[j]) vec.push_back(w[i]), ++i;
        else vec.push_back(w[j]), res += mid-i+1, ++j;
    }

    while (i<=mid) vec.push_back(w[i]), ++i;
    while (j<=r) vec.push_back(w[j]), ++j;

    for (int i = 0, j = l; i < (int)vec.size(); ++i, ++j) w[j] = vec[i];
```

```
}

int main()
{
    int n; cin >> n;
    for (int i = 1; i <= n; i++) cin >> w[i];
    mergeSort(1, n);
    cout << res << endl;
    return 0;
}
```

CF1117C Magic Ship

前缀和 + 二分

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

using namespace std;

typedef long long LL;
const int maxn = 1e5 + 5;
char str[maxn];
int wx[maxn], wy[maxn];
int sx, sy, ex, ey, n;

int xDelta(char c) {
  if (c == 'L') return -1;
  if (c == 'R') return 1;
  return 0;
}
int yDelta(char c) {
  if (c == 'U') return 1;
  if (c == 'D') return -1;
  return 0;
}

bool check(LL mid) {
  LL k = mid / n;
  int t = mid % n;

  LL x = sx + k*wx[n] + wx[t], y = sy + k*wy[n] + wy[t];
  LL d = abs(ex - x) + abs(ey - y);
  return d <= mid;
}

int main()
{
  cin >> sx >> sy >> ex >> ey;
  cin >> n;
  cin >> (str+1);
  for (int i = 1; i <= n; ++i) {
```

```
    wx[i] = wx[i-1] + xDelta(str[i]);
    wy[i] = wy[i-1] + yDelta(str[i]);
  }

  LL l = 0, r = 1e14+1;
  while (l < r) {
    LL mid = (l + r) >> 1;
    if (check(mid)) r = mid;
    else l = mid+1;
  }
  if (l == 1e14+1) cout << -1 << endl;
  else cout << l << endl;
  return 0;
}
```