综合练习

人员

赵熙羽、杨瑾硕、谢亚锴、陈洛冉 到课, 于子珈 线上

上周作业检查

https://cppoj.kids123code.com/contest/133

2025-0629 周日10:30 (综合练习)

刷新

#	用户名	姓名	总分	选择分	编程分	时间	A	B	C	D
1	zhaoxiyu	赵熙羽	400	0	400	2104	100	100	100	100
2	yuzijia1	于子珈	310	0	310	2128	100	100	20	90
3	yangjinshuo	杨谨硕	200	0	200	2438	100	100		
4	liuchuangsu	刘闯速	200	0	200	2698	100	100		
5	qinxiansen	秦显森	190	0	190	2687	100		90	
6	xieyakai	谢亚锴	100	0	100	1533	100			
7	gaojianhuan	高健桓	100	0	100	2136	100			
8	sunjingke	孙靖轲	100	0	100	2145	100			

作业

https://cppoj.kids123code.com/contest/178 (课上讲了 A ~ B 题, 课后作业是 C 题)

课堂表现

今天课上的题相对复杂一些, A 题尤其是一种 正难则反 的思想, 争着做比较难所以反过来二分做, 同学们课下可以好好复习复习。

课堂内容

[蓝桥杯 2025 省 A/Python B 第二场] 消消乐

希望最终剩下的字符尽量多, 说明要尽量少的消掉原本字符串中的 A 和 B, 使得最终的字符串中不存在 前A后B 的情况

可以参考 ABAB 这个例子, 我们可以消掉第一个 A 和 最后一个 B, 就只剩 BA, 不需要再消除了

那么, 就是希望尽量消掉靠前的 A 和靠后的 B

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

using namespace std;

const int maxn = 1e6 + 5;
char s[maxn];
```

```
int main()
{
  cin >> (s+1);
  int n = strlen(s+1);
  int i = 1, j = n;
  int res = n;
  while (i < j) {
    while (i<j && s[i]=='B') ++i;
    while (i<j && s[j]=='A') --j;
    if (i < j) res -= 2, ++i, --j;
    else break;
  }
  cout << res << endl;
  return 0;
}
```

公园

二分 + 二维前缀和

二分, 判断是否存在有一个 K * K 的区间的中位数超过 mid

可以把整个数组中所有 >=mid 的值定为 1, 把 <mid 的值定为 0

然后就是判断是否有一个 K * K 的矩阵的矩阵和超过 (K * K) / 2 + 1 即可

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

using namespace std;

const int maxn = 800 + 5;
int w[maxn][maxn], f[maxn][maxn];

int get_sum(int x1, int y1, int x2, int y2) {
  return f[x2][y2] - f[x1-1][y2] - f[x2][y1-1] + f[x1-1][y1-1];
}

bool check(int n, int k, int mid) {
  memset(f, 0, sizeof(f));
  for (int i = 1; i <= n; ++i) {
    for (int j = 1; j <= n; ++j) {
      f[i][j] = (w[i][j]<=mid ? 1 : 0);
      f[i][j] += f[i-1][j] + f[i][j-1] - f[i-1][j-1];
      if (i>=k && j>=k && get_sum(i-k+1, j-k+1, i, j)>=(k*k+1)/2) return true;
    }
  }
  return false;
}

int main()
{
```

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

  int l = 0, r = 1e9;
  while (l <= r) {
    int mid = (l + r) / 2;
    if (check(n, k, mid)) r = mid-1;
    else l = mid+1;
  }
  cout << l << endl;
  return 0;
}
```

地标访问

贪心思路: 一定是先往左走一段, 然后直接往右走; 或者是先往右走一段, 然后直接往左走

开一个 -1e5 ~ 1e5 的桶数组, 哪些位置有对应的地标, 就在对应桶的位置 +1 即可

然后枚举往左走一段的终点, 考虑在这个情况下往右最多走到哪; 再枚举往右走一段的终点, 考虑这种情况下往左最多走打哪。

中间经过多少点可以用前缀和来快速求

最后, 因为数组不能开负数, 给数组加一个偏移量即可, 把 -1e5 ~ 1e5 映射到 0 ~ 2e5即可

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

using namespace std;

const int N = 1e6 + 5;
int w[2*N], p[2*N];

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

int main()
{
  int n, m; cin >> m >> n;
  while (n -- ) { int x; cin >> x; w[x+N]++; }

  for (int i = 1; i < 2*N; ++i) p[i] = p[i-1] + w[i];

  int res = 0;
  for (int i = 0; i <= N; ++i) {
    if (N - i > m) continue;
    int j = max(N, min(m-2*(N-i)+N, 2*N-1));
    res = max(res, get_sum(i, j));
  }
```

```
  for (int i = 2*N-1; i >= N; --i) {
    if (i - N > m) continue;
    int j = min(N, max(N-m-2*(N-i), 0));
    res = max(res, get_sum(j, i));
  }
  cout << res << endl;
  return 0;
}
```

飞机降落

找一个最佳排列方式问题

N <= 10, 所以可以暴力枚举全部的排列, 看是否能有一个排列符合要求即可

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

using namespace std;

const int maxn = 10 + 5;
int T[maxn], D[maxn], L[maxn];
int w[maxn];
int n;

bool check() {
  int last = -1;
  for (int i = 1; i <= n; ++i) {
    int id = w[i];
    int l = T[id], r = T[id] + D[id];
    if (last > r) return false;
    last = max(last, l) + L[id];
  }
  return true;
}

void solve() {
  cin >> n;
  for (int i = 1; i <= n; ++i) cin >> T[i] >> D[i] >> L[i];

  for (int i = 1; i <= n; ++i) w[i] = i;
  do {
    if (check()) { cout << "YES" << endl; return; }
  } while (next_permutation(w+1, w+n+1));
  cout << "NO" << endl;
}

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

```
    return 0;
}
```