大连理工大学

ACM 代码册

Edited by

PuFanyi

目录

第一章　搜索

1.1　Dancing Links

```
const int maxn = 505;
const int maxm = 6005;

struct Dancing_Links {
    int n, m, total, ans;

    struct Node {
        int up, down, left, right, row, column;
    } no[maxm];

    int siz[maxn];
    int first[maxn];
    int stk[maxn];

    void init(int n, int m) {
        ans = 0;
        this->n = n, this->m = m;
        memset(first, 0, sizeof(first));
        memset(siz, 0, sizeof(siz));
        for (int i = 0; i <= m; ++i) {
            no[i].left = i - 1, no[i].right = i + 1;
            no[i].up = no[i].down = i;
        }
        no[0].left = m, no[m].right = 0, total = m;
    }

    void insert(int row, int col) {
        total++, siz[col]++;
        no[total].row = row, no[total].column = col;
        no[total].down = col, no[total].up = no[col].up;
        no[col].up = total, no[no[total].up].down = total;
        if (!first[row]) {
            first[row] = no[total].left = no[total].right = total;
        } else {
            no[total].right = first[row], no[total].left = no[first[row]].left;
```

```
36              no[no[total].left].right = no[first[row]].left = total;
37          }
38      }
39
40      void remove(int col) {
41          no[no[col].left].right = no[col].right;
42          no[no[col].right].left = no[col].left;
43          for (int i = no[col].down; i != col; i = no[i].down) {
44              for (int j = no[i].right; j != i; j = no[j].right) {
45                  no[no[j].up].down = no[j].down;
46                  no[no[j].down].up = no[j].up;
47                  siz[no[j].column]--;
48              }
49          }
50      }
51
52      void recover(int col) {
53          for (int i = no[col].up; i != col; i = no[i].up) {
54              for (int j = no[i].left; j != i; j = no[j].left) {
55                  no[no[j].up].down = no[no[j].down].up = j;
56                  siz[no[j].column]++;
57              }
58          }
59          no[no[col].left].right = no[no[col].right].left = col;
60      }
61
62      bool dance(int dep) {
63          if (!no[0].right) {
64              ans = dep - 1;
65              return true;
66          }
67          int col = no[0].right;
68          for (int i = no[0].right; i; i = no[i].right) {
69              if (siz[i] < siz[col]) {
70                  col = i;
71              }
72          }
73          remove(col);
74          for (int i = no[col].down; i != col; i = no[i].down) {
75              stk[dep] = no[i].row;
76              for (int j = no[i].right; j != i; j = no[j].right) {
77                  remove(no[j].column);
78              }
79              if (dance(dep + 1)) {
80                  return true;
81              }
82              for (int j = no[i].left; j != i; j = no[j].left) {
```

```
83                  recover(no[j].column);
84              }
85          }
86          recover(col);
87          return false;
88      }
89  } dlx;
90
91  int main() {
92      int n, m, x;
93      read(n), read(m);
94      dlx.init(n, m);
95      for (int i = 1; i <= n; ++i) {
96          for (int j = 1; j <= m; ++j) {
97              if (read(x) && x) {
98                  dlx.insert(i, j);
99              }
100         }
101     }
102     if (dlx.dance(1)) {
103         for (int i = 1; i <= dlx.ans; ++i) {
104             writesp(dlx.stk[i]);
105         }
106         puts("");
107     } else {
108         puts("No Solution!");
109     }
110     return 0;
111 }
```

1.2 $\alpha-\beta$ 剪枝

第二章　动态规划

第三章　字符串

第四章　数学

4.1　快速幂

```
template <class T>
T ksm(T a, T b, T mod) {
    T ans = 1;
    for (; b; b >>= 1, a = (LL) a * a % mod) {
        if (b & 1) {
            ans = (LL) ans * a % mod;
        }
    }
    return ans;
}
```

4.2　位运算

4.2.1　Gray 码

```
int g(int n) {
    return n ^ (n >> 1);
}

int rev_g(int g) {
    int n = 0;
    for (; g; g >>= 1) {
        n ^= g;
    }
    return n;
}
```

4.3　数论

4.3.1　最大公约数

4.3.2　欧几里得算法

```
template <class T>
T gcd(T a, T b) {
    while (b) {
        int t = a % b;
        a = b;
        b = t;
    }
    return a;
}
```

4.3.3 筛法

Eratosthenes 筛法

Euler 筛法

```
void oula(const int n = 100000) {
    np[1] = true;
    int cnt = 0;
    for (int i = 2; i <= n; ++i) {
        if (!np[i]) {
            prime[++cnt] = i;
        }
        for (int j = 1; j <= cnt && (LL) i * prime[j] <= n; ++j) {
            np[i * prime[j]] = true;
            if (!(i % prime[j])) {
                break;
            }
        }
    }
}
```

第五章　数据结构

5.1　动态树

5.1.1　Link-Cut Tree

```cpp
#include <cstdio>
#include <iostream>
#include <algorithm>

using namespace std;

const int maxn = 300005;

class LCT {
    // node

public:
    int sum[maxn], val[maxn];
    int s[maxn][2], fa[maxn];

private:
    bool lzy_fan[maxn];

    void push_up(int x) {
        sum[x] = val[x] ^ sum[s[x][0]] ^ sum[s[x][1]];
    }

    bool nrt(int x) {
        return s[fa[x]][0] == x || s[fa[x]][1] == x;
    }

    void fan(int x) {
        swap(s[x][0], s[x][1]);
        lzy_fan[x] ^= 1;
    }

    void push_down(int x) {
        if (lzy_fan[x]) {
```

```
34              if (s[x][0]) {
35                  fan(s[x][0]);
36              }
37              if (s[x][1]) {
38                  fan(s[x][1]);
39              }
40              lzy_fan[x] = 0;
41          }
42      }
43
44      // splay
45
46  private:
47      void rotate(int x) {
48          int y = fa[x], z = fa[y];
49          int k = (s[y][1] == x), ss = s[x][!k];
50          if (nrt(y)) {
51              s[z][s[z][1] == y] = x;
52          }
53          fa[x] = z;
54          s[x][!k] = y;
55          fa[y] = x;
56          s[y][k] = ss;
57          if (ss) {
58              fa[ss] = y;
59          }
60          push_up(y);
61          push_up(x);
62      }
63
64      int sta[maxn];
65      void splay(int x) {
66          int K = x, top = 0;
67          sta[++top] = K;
68          while (nrt(K)) {
69              sta[++top] = K = fa[K];
70          }
71          while (top) {
72              push_down(sta[top--]);
73          }
74          while (nrt(x)) {
75              int y = fa[x], z = fa[y];
76              if (nrt(y)) {
77                  rotate(((s[y][0] == x) ^ (s[z][0] == y)) ? x : y);
78              }
79              rotate(x);
80          }
```

```
81      }
82
83      // LCT
84  private:
85      void access(int x) {
86          for (int y = 0; x; x = fa[y = x]) {
87              splay(x);
88              s[x][1] = y;
89              push_up(x);
90          }
91      }
92
93      void make_root(int x) {
94          access(x);
95          splay(x);
96          fan(x);
97      }
98
99      int find_root(int x) {
100         access(x);
101         splay(x);
102         while (s[x][0]) {
103             push_down(x);
104             x = s[x][0];
105         }
106         splay(x);
107         return x;
108     }
109
110     void split(int x, int y) {
111         make_root(x);
112         access(y);
113         splay(y);
114     }
115
116 public:
117     void link(int x, int y) {
118         make_root(x);
119         if (find_root(y) != x) {
120             fa[x] = y;
121         }
122     }
123
124     void cut(int x, int y) {
125         make_root(x);
126         if (find_root(y) == x && fa[y] == x && !s[y][0]) {
127             fa[y] = s[x][1] = 0;
```

```
128             push_up(x);
129         }
130     }
131
132     void change(int x, int y) {
133         splay(x);
134         val[x] = y;
135         push_up(x);
136     }
137
138     int ask(int x, int y) {
139         split(x, y);
140         return sum[y];
141     }
142 } tr;
143
144 int main() {
145     int n, m;
146     scanf("%d%d", &n, &m);
147     for (int i = 1; i <= n; ++i) {
148         scanf("%d", &tr.val[i]);
149         tr.sum[i] = tr.val[i];
150     }
151     while (m--) {
152         int cmd, x, y;
153         scanf("%d%d%d", &cmd, &x, &y);
154         switch (cmd) {
155             case 0:
156             printf("%d\n", tr.ask(x, y));
157             break;
158             case 1:
159             tr.link(x, y);
160             break;
161             case 2:
162             tr.cut(x, y);
163             break;
164             case 3:
165             tr.change(x, y);
166         }
167     }
168     return 0;
169 }
```

第六章　图论

第七章　计算几何

第八章　其他

8.1　读入输出优化

```
inline char gc() {
    static const int L = 23333;
    static char sxd[L], *sss = sxd, *ttt = sxd;
    if (sss == ttt) {
        ttt = (sss = sxd) + fread(sxd, 1, L, stdin);
        if (sss == ttt) {
            return EOF;
        }
    }
    return *sss++;
}

#ifdef Debug
#define dd c = getchar()
#else
#define dd c = gc()
#endif
template <class T>
inline bool read(T& x) {
    x = 0;
    char dd;
    bool flg = false;
    for (; !isdigit(c); dd) {
        if (c == '-') {
            flg = true;
        } else if (c == EOF) {
            return false;
        }
    }
    for (; isdigit(c); dd) {
        x = (x * 10) + (c ^ 48);
    }
    if (flg) {
        x = -x;
    }
```

```
36      return true;
37  }
38  #undef dd
39
40  template <class T>
41  inline void write(T x) {
42      if (x < 0) {
43          x = -x;
44          putchar('-');
45      }
46      if (x > 9) {
47          write(x / 10);
48          x %= 10;
49      }
50      putchar(x | 48);
51  }
52
53  template <class T>
54  inline void writeln(T x) {
55      write(x);
56      puts("");
57  }
58
59  template <class T>
60  inline void writesp(T x) {
61      write(x);
62      putchar(' ');
63  }
```