A 错误探测

```cpp
#include <iostream>
#include <queue>
#include <string.h>
#include <stdio.h>
#include <set>
using namespace std;
int dt[102][102];
int main(){
    int N;
    cin>>N;
    for(int i=0;i<N;++i){
        for(int j=0;j<N;++j)
            cin>>dt[i][j];
    }
    int tot=0, ax=0, ay=0;
    for(int i=0;i<N;++i){
        int l=0;
        for(int j=0;j<N;++j)
            l+=dt[i][j];
        if(l%2==1){tot++;ax=i;}
    }
    int t2=0;
    for(int i=0;i<N;++i){
        int l=0;
        for(int j=0;j<N;++j)
            l+=dt[j][i];
        if(l%2==1){t2++;ay=i;}
    }
    if(t2==0&&tot==0)
        printf("OK");
    else if(t2==1&&tot==1)
        printf("%d %d\n",ax+1,ay+1);
    else printf("Corrupt");
}
```

B 移动办公

```cpp
#include <iostream>
using namespace std;
int T,M;
const int mx = 110;
int p[mx],n[mx];
int dp[mx][2]; // dp[i][0] 第 i 个月在北京办公，从此及以前最大总营业额
               // dp[i][1]  第 i 个月在南京办公，从此及以前最大总营业额
int main()
{
  cin >> T >> M;
  for(int i = 0;i < T; ++i)
    cin >> p[i] >> n[i];
  dp[0][0] = p[0];
  dp[0][1] = n[0];
  for(int i = 1;i < T; ++i) {
    dp[i][0] = max(dp[i-1][0] + p[i],dp[i-1][1]-M + p[i]);
    dp[i][1] = max(dp[i-1][1] + n[i],dp[i-1][0]-M + n[i]);
  }
  cout << max(dp[T-1][0],dp[T-1][1]) << endl;
```

```

}
```

C 重启系统

```cpp
#include <iostream>
using namespace std;
int N;

const int mx = 1010;
int a[mx];
int dp[mx][2]; //dp[i][0] 表示以 ai 为终点的，没有重启过的最长不升子序列的长度
               // dp[i][1] 表示以 ai 为终点的，重启过的最长不升子序列的长度

int main()
{
  cin >> N;
  for(int i = 0;i < N; ++i) {
    cin >> a[i];
    dp[i][0] = dp[i][1] = 1;
  }
  for(int i = 1;i < N; ++i) {
    for(int j = 0; j < i; ++j) {
      if( a[j] >= a[i] ) {
        dp[i][0] = max(dp[i][0],dp[j][0] + 1);
        dp[i][1] = max(dp[i][1],dp[j][1] + 1);
      }
      dp[i][1] = max(dp[i][1],dp[j][0]+1) ; //在 i 之前一瞬间重启
    }
  }

  int ans = 0;
  for(int i = 0;i < N; ++i)
    ans = max(ans,dp[i][1]);
  cout << ans << endl;
}
```

D 苹果消消乐

```cpp
#include <bits/stdc++.h>
using namespace std;
int fruits[105];
int main(){
    int T, n, m;
    cin >> T;
    while(T--) {
        cin >> n >> m;
        for(int i = 1; i <= n; i++) cin >> fruits[i];
        int res = 0;
        if(n <= m) {
            cout << 100 << endl;
            continue;
        }
        res = max(res, fruits[m+1] - 1);
        res = max(res, 100 - fruits[n-m]);
        for(int i = 2; i <= n - m; i++)
            res = max(res, fruits[i+m] - fruits[i-1] - 1);
```

```
        cout << res << endl;
    }
    return 0;
}
```

E 课程小作业

```
#include<iostream>
#include<stdlib.h>
#include<stdio.h>
#include<set>
#include<queue>
#include<string.h>
#include<algorithm>
using namespace std;
struct item
{
    int m,t;
};
bool cmp(const item&a,const item&b)
{if(a.t==b.t)return a.m>b.m;return a.t<b.t;}
item it[10040];
int n;
int sell[10040];
int main()
{
    memset(sell,0,sizeof(sell));
    cin>>n;
    for(int i=0;i<n;i++)
    scanf("%d%d",&it[i].m,&it[i].t);
    sort(it,it+n,cmp);
    for(int i=0;i<n;i++)
    {
        int m3=10000000,mpo;
        for(int j=it[i].t;j>=1;j--)
        {
            if(sell[j]<m3){m3=sell[j];mpo=j;}
        }
        if(m3<it[i].m)
        {
            sell[mpo]=it[i].m;
        }
    }
    int pss=0;
    for(int i=1;i<=10000;i++)
        pss+=sell[i];
    cout<<pss<<endl;
}
```

F 宝藏

```
//19175   宝藏 滚动数组的背包动规
#include <iostream>
#include <cstring>
#include <algorithm>
using namespace std;
int N,W;
```

```

const int mx = 10010;

int dp[2][mx]; //dp[i][j] 表示从第 i 件物品到第 N-1 件物品中取物，总重量不超过 j，能拿到的最大价值。物品按价值从小到大排序了，且序号从 0 开始
//用滚动数组，所以只有 3 行

struct Good {
  int v;
  int w;
  bool operator< ( const Good & g) {
    return v < g.v;
  }
};
Good goods[mx];

int main()
{
  scanf("%d%d",&N,&W);
  for(int i = 0;i < N; ++i)
    scanf("%d%d",&goods[i].v,&goods[i].w);
  sort(goods,goods + N);
  memset(dp,0,sizeof(dp));
  for(int w = 0; w <= W; ++ w) {
    if( goods[N-1].w <= w)
      dp[0][w] = goods[N-1].v;  // dp[0] 对应于第 N -1 行，dp[1] 对应于第 N 行 。第 N 行全是 0
  }
  int first = 0;
  for(int i = N - 2; i >= 0; -- i) {
    for(int j = W; j >= 0; -- j) {
      int tmpAns = dp[first][j]; //dp[first][j] 相当于是 dp[i+1][j]
      if( goods[i].w <= j )
        tmpAns = max( tmpAns, goods[i].v + dp[1-first][j-goods[i].w]);  // dp[1-first] 相当于 dp[i+2]
      dp[1-first][j] = tmpAns;
    }
    first = 1- first;
  }
  printf("%d\n",dp[first][W]);
  return 0;
}
```

G 课程表

```
#include <iostream>
#include <cstring>
#include <vector>
using namespace std;
int n,m;
const int mx = 1010;
vector< vector <int> > G(mx);
bool visited[mx];
bool inPath[mx]; // inPath[i] 表示点 i 是否在当前搜索路径上

bool dfs(int v)
{ //返回值 true 代表发现环

  //cout <<"v:" <<  v << endl;
  bool result = false;
  visited[v] = true;
```

```
  inPath[v] = true;
  for(int i = 0;i < G[v].size(); ++i) {
    int u = G[v][i];
    if( !visited[u]) {
      //cout <<"u:" <<  v << endl;
      if( dfs(u) ) {
        result = true;
        break;
      }
    }
    else {
      if( inPath[u])
        result = true;
    }
  }
  inPath[v] = false;
  return result;

}
int main()
{
  while( scanf("%d%d",&n,&m) == 2) {
    bool hasLoop = false;
    for(int i = 0;i < n; ++i) {
      visited[i] = false;
      inPath[i] = false;
      G[i].clear();
    }
    for(int i = 0;i < m; ++i) {
      int s,e;
      scanf("%d%d",&s,&e);
      G[s].push_back(e);
    }
    for(int i = 0;i < n; ++i)
      if( ! visited[i]) {
        if( dfs(i)) {//有环
          hasLoop = true;
          break;
        }
      }
    if( hasLoop)
      printf("False\n");
    else
      printf("True\n");
  }
  return 0;
}

```

H 密室逃脱

```
#include <iostream>
#include <cstring>
#include <map>
#include <vector>
#include <functional>
#include <iterator>
#include <cstdlib>
```

```cpp
#include <cstdio>
#include <algorithm>
#include <queue>
#include <set>
using namespace std;

#define MAXR 101
#define MAXC 101

bool mem[MAXR][MAXC];

struct state {
  int x, y;
  int steps;
};

int bfs(vector<vector<int> > &room, int srcx, int srcy, int dstx, int dsty)
{
  int dx[] = {0,1,0,-1};
  int dy[] = {1,0,-1,0};

  int m = room.size();
  int n = room[0].size();
  queue<state> q;
  q.push({srcx, srcy, 0});
  for (int i = 0; i < m; ++i)
    for (int j = 0; j < n; ++j)
      mem[i][j] = false;
  mem[srcx][srcy] = true;
  while (!q.empty()) {
    state cur = q.front();
    q.pop();
    if (cur.x == dstx && cur.y == dsty)
      return cur.steps;
    for (int di = 0; di < 4; ++di) {
      int r = cur.x + dx[di];
      int c = cur.y + dy[di];
      if (0 <= r && r < m && 0 <= c && c < n &&
        !mem[r][c] && room[r][c] > 0) {
        mem[r][c] = true;
        q.push({r, c, cur.steps + 1});
      }
    }
  }
  return -1;
}


int solve(vector<vector<int> > &room)
{
  vector<vector<int> > points;
  for (int i = 0; i < room.size(); ++i)
    for (int j = 0; j < room[0].size(); ++j) {
      int v = room[i][j];
      if (v > 1) {
        vector<int> tmp;
        tmp.push_back(v);
        tmp.push_back(i);
```

```
66          tmp.push_back(j);
67          points.push_back(tmp);
68        }
69      }
70    sort(points.begin(), points.end());  // default vector compare function
71    int ret = 0, srcx = 0, srcy = 0;
72    for (int i = 0; i < points.size(); ++i) {
73      int d = bfs(room, srcx, srcy, points[i][1], points[i][2]);
74      if (d < 0)
75        return -1;
76      ret += d;
77      srcx = points[i][1];
78      srcy = points[i][2];
79    }
80    return ret;
81  }
82
83
84  int main()
85  {
86    int T;
87    // cin >> T;
88    scanf("%d", &T);
89    while (T--) {
90      int m, n;
91      vector<vector<int> > room;
92      // cin >> m >> n;
93      scanf("%d %d", &m, &n);
94      for (int i = 0; i < m; ++i) {
95        vector<int> row;
96        int tmp;
97        for (int j = 0; j < n; ++j) {
98          // cin >> tmp;
99          scanf("%d", &tmp);
100         row.push_back(tmp);
101       }
102       room.push_back(row);
103     }
104     cout << solve(room) << endl;
105   }
106   return 0;
107 }
```

I 控制公司

```
1   #include<bits/stdc++.h>
2   using namespace std;
3
4   const int Max = 10001;
5   int num, n;
6   int sum[Max];
7   bool vis[Max];
8   int matrix[101][101];
9
10  void dfs(int x) {
11      vis[x] = true;
12      for(int i = 1; i <= num; i++) {
13          sum[i] += matrix[x][i];
```

```
        if(sum[i] > 50 && !vis[i])
            dfs(i);
    }

}
int main() {
    vector<string> file_lis = {"ctrl0", "ctrl1", "ctrl2"};
        cin >> n;
        int x,y,z;

        memset(matrix, 0, sizeof matrix);
        for(int i=0; i<n; i++) {
            cin >> x >> y >> z;
            matrix[x][y] += z ;
            num = max(num, y);
            num = max(num, x);
        }
        for(int i = 1; i <= num; i++) {
            memset(vis, 0, sizeof vis);
            memset(sum, 0, sizeof sum);
            dfs(i);
            for(int j = 1; j <= num; j++)
                if(j != i && sum[j] > 50)
                    cout << i << ' ' << j << endl;
        }

    return 0;
}
```

J 冠军之路

```
#include <iostream>
#include <queue>
#include <string.h>

using namespace std;
struct st{
    int x, y,p;
    st(int _x, int _y, int _p):x(_x),y(_y),p(_p){}
};
char area[103][103];
int score[103][103];
int vis[103][103][256];
char dx[4]={1,-1,0,0};
char dy[4]={0,0,1,-1};
int main(){
    int N, M, K;
    cin>>N>>M;
    int sx,sy,ex,ey;
    memset(area, '#',sizeof(area));
    memset(vis,0,sizeof(vis));
    for(int i=1;i<=N;++i){
        for(int j=1;j<=M;++j){
            cin>>area[i][j];
            if(area[i][j]=='I'){
                sx=i,sy=j;
```

```
                area[i][j]='.';
            }
            if(area[i][j]=='O'){
                ex=i,ey=j;
                area[i][j]='.';
            }
        }
    }
    int tr=0,cursc=0;
    for(int i=1;i<=N;++i){
        for(int j=1;j<=M;++j){
            if(area[i][j]=='w'){
                cursc=1<<tr;tr++;
                for(int p=i-1;area[p][j]=='.';p--){
                    score[p][j]+=cursc;
                }
            }
            if(area[i][j]=='a'){
                cursc=1<<tr;tr++;
                for(int p=j-1;area[i][p]=='.';p--){
                    score[i][p]+=cursc;
                }
            }
            if(area[i][j]=='s'){
                cursc=1<<tr;tr++;
                for(int p=i+1;area[p][j]=='.';p++){
                    score[p][j]+=cursc;
                }
            }
            if(area[i][j]=='d'){
                cursc=1<<tr;tr++;
                for(int p=j+1;area[i][p]=='.';p++){
                    score[i][p]+=cursc;
                }
            }
        }
    }
    queue<st> q;
    q.push(st(sx,sy,score[sx][sy]));
    while(!q.empty()){
        st cur=q.front();
        q.pop();
        int cx=cur.x,cy=cur.y,p=cur.p;
        //printf("%d %d %d %d\n",cx,cy,p,vis[cx][cy][p]);
        for(int i=0;i<4;++i){
            int nx=cx+dx[i],ny=cy+dy[i];
            if(area[nx][ny]=='.'){
                int newp=cur.p|score[nx][ny];
                if(vis[nx][ny][newp]==0||vis[nx][ny][newp]>vis[cx][cy][p]+1){
                    q.push(st(nx,ny,newp));
                    vis[nx][ny][newp]=vis[cx][cy][p]+1;
                    if(nx==ex&&ny==ey&&newp==(1<<tr)-1){
                        cout<<vis[ex][ey][(1<<tr)-1]<<endl;
                        return 0;
                    }
                }
            }
        }
```

```
    }
    cout<<-1<<endl;

}
```

K 他和他的猫

```
#include "stdio.h"
#include "cstring"
#include "algorithm"
using namespace std;

#define MAX 100005

struct point
{
    long long x, y;
}monoq[MAX] = {};

int N = 0, M = 0, P = 0, num = 0, le = 0, ri = 0;
long long dis[MAX] = {}, dest[MAX] = {};
long long preff[MAX] = {}, dp[2][MAX] = {};

double slape(int ptr)
{
    return double(monoq[ptr + 1].y - monoq[ptr].y) / double(monoq[ptr + 1].x - monoq[ptr].x);
}

int main()
{
    scanf("%d", &num);
    while(num--)
    {
        memset(dis, 0, sizeof(dis));
        memset(dest, 0, sizeof(dest));
        memset(preff, 0, sizeof(preff));
        memset(dp, 0, sizeof(dp));
        memset(monoq, 0, sizeof(monoq));

        long long H = 0, T = 0;
        scanf("%d%d%d", &N, &M, &P);
        for(int i = 2; i <= N; i++)
        {
            scanf("%lld", dis + i);
            dis[i] += dis[i - 1];
        }
        for(int i = 1; i <= M; i++)
        {
            scanf("%lld%lld", &H, &T);
            dest[i] = T - dis[H];
        }

        sort(dest + 1, dest + M + 1);
        for(int i = 1; i <= M; i++) preff[i] = preff[i - 1] + dest[i];
        for(int i = 1; i <= M; i++) dp[0][i] = i * dest[i] - preff[i];

        int ptr = 0;
        for(int j = 1; j < P; j++)
```

```
        {
            ptr ^= 1;
            le = 0, ri = 1;
            monoq[0].x = monoq[0].y = 0;
            for(int i = 1; i <= M; i++)
            {
                while(ri > le + 1 && slape(le) < dest[i]) ++le;

                dp[ptr][i] = monoq[le].y - dest[i] * monoq[le].x + i * dest[i] - preff[i];
                monoq[ri].x = i;
                monoq[ri++].y = dp[ptr ^ 1][i] + preff[i];

                while(ri > le + 2 && slape(ri - 3) >= slape(ri - 2))
                    monoq[ri - 2] = monoq[ri - 1], --ri;
            }
        }
        printf("%lld\n", dp[ptr][M]);
    }
    return 0;
}
```

L 多连块拼图

```
#include <set>
#include <map>
#include <list>
#include <cmath>
#include <ctime>
#include <deque>
#include <queue>
#include <stack>
#include <cctype>
#include <cstdio>
#include <string>
#include <vector>
#include <cassert>
#include <cstdlib>
#include <cstring>
#include <sstream>
#include <iostream>
#include <algorithm>

using namespace std;

int m, n;
char bigp[15][15], smallp[15][15];

int countStar( char a[15][15], int n ) {
  int res = 0;
  for( int i = 0; i < n; i++ ) for( int j = 0; j < n; j++ ) res += ( a[i][j] == '*' );
  return res;
}

bool isValid( int x, int y ) {
  return x >= 0 && x < n && y >= 0 && y < n;
}

bool isSame( int i, int j ) {
```

```
36    for( int x = 0; x < m; x++ ) for( int y = 0; y < m; y++ ) if( smallp[x][y] == '*' )
37      if( !isValid( x + i, y + j ) || bigp[x+i][y+j] != '*' ) return false;
38    return true;
39  }
40
41  bool canMatch( int k, int i, int j ) {
42    if( !k ) return true;
43    if( i == n ) return false;
44    if( j == n ) return canMatch( k, i + 1, -10 );
45
46    if( isSame( i, j ) ) {
47      for( int x = 0; x < m; x++ ) for( int y = 0; y < m; y++ ) if( smallp[x][y] == '*' ) bigp[x+i][y+j] = '.';
48      if( canMatch( k - 1, i, j ) ) return true;
49      for( int x = 0; x < m; x++ ) for( int y = 0; y < m; y++ ) if( smallp[x][y] == '*' ) bigp[x+i][y+j] = '*';
50    }
51    return canMatch( k, i, j + 1 );
52  }
53
54  int main() {
55    double cl = clock();
56
57    while( scanf("%d %d", &n, &m) == 2 && m + n ) {
58      for( int i = 0; i < n; i++ ) scanf("%s", bigp[i]);
59      for( int i = 0; i < m; i++ ) scanf("%s", smallp[i]);
60
61      int cnt1 = countStar( bigp, n );
62      int cnt2 = countStar( smallp, m );
63
64      if( cnt1 != cnt2 * 2 || !canMatch(2, -10, -10) ) puts("0");
65      else puts("1");
66    }
67
68    cl = clock() - cl;
69    //fprintf(stderr, "Total Execution Time = %lf seconds\n", cl / CLOCKS_PER_SEC);
70
71    return 0;
72  }
```