深搜和广搜

普通

BFS

走迷宫
注意状态框架

```
struct Coord {
    uint8_t x, y;
    Coord() {}
    Coord(int x, int y):x(x), y(y) {}
};

char g[N][N];
int d[N][N];
int r, c;

Coord q[N * N];

int dir[4][2] = {
    {-1, 0}, {1, 0}, {0, -1}, {0, 1}
};

inline bool inbound(int x, int y) { return 0 <= x && x < r && 0 <= y && y <
c; }

int bfs() {
    int head = 0, tail = 0;
    q[0] = {0, 0};

    memset(d, -1, sizeof d);
    d[0][0] = 1;  // contain start point

    int x, y;
    while (head <= tail) {
        auto& coord = q[head++];
        for (int i = 0; i < 4; ++i) {
            x = coord.x + dir[i][0];
            y = coord.y + dir[i][1];
            if (inbound(x, y) && g[x][y] == '.' && d[x][y] == -1) {
                d[x][y] = d[coord.x][coord.y] + 1;
                q[++tail] = {x, y};
            }
        }
```

```
    }

    return d[r-1][c-1];
}

int main() {
    cin.tie(0);
    cin >> r >> c;
    for (int i = 0; i < r; ++i)
        for (int j = 0; j < c; ++j)
            cin >> g[i][j];
    cout << bfs() << endl;
    return 0;
}
```

DFS

八皇后问题

```
int n;
char board[N][N];
bool cols[N], dig[N<<1], adig[N<<1];

/*  n = 3
dig[]: col - row = [-2, 2], so add n=3 would be [1, 5]   <=>   y = x + b,
that is b = y - x

    ----------   ----------   ----------   ----------   ----------
    |*|__|__|   |__|*|__|   |__|__|*|   |__|__|__|   |__|__|__|
    |__|*|__|   |__|__|*|   |__|__|__|   |__|__|__|   |*|__|__|
    |__|__|*|   |__|__|__|   |__|__|__|   |*|__|__|   |__|*|__|
      [0+3=3]      [1+3=4]      [2+3=5]     [-2+3=1]     [-1+3=2]

adig[]: col + row = [0, 4]  <=>  y = -x + b, that is b = y + x

    ----------   ----------   ----------   ----------   ----------
    |*|__|__|   |__|*|__|   |__|__|*|   |__|__|__|   |__|__|__|
    |__|__|__|   |*|__|__|   |__|*|__|   |__|__|*|   |__|__|__|
    |__|__|__|   |__|__|__|   |*|__|__|   |__|*|__|   |__|__|*|
        [0]          [1]          [2]          [3]          [4]
*/

void dfs(int row) {
    if (row == n) {
        for (int i = 0; i < n; ++i) puts(board[i]);
        puts("");
        return;
    }
    for (int col = 0; col < n; ++col) {
        if (cols[col] || dig[col - row + n] || adig[col + row]) continue;
        board[row][col] = 'Q';
```

```
        cols[col] = dig[col - row + n] = adig[col + row] = 1;
        dfs(row+1);
        cols[col] = dig[col - row + n] = adig[col + row] = 0;
        board[row][col] = '.';
    }
}

int main() {
    cin >> n;
    for (int i = 0; i < n; ++i)
        for (int j = 0; j < n; ++j)
            board[i][j] = '.';
    dfs(0);
    return 0;
}
```

图中

【dfs】

- 树重心的定义：删除重心后，剩余各连通块节点数的最大值最小（即尽可能拆得均匀）
  - dfs遍历可以求每个子树的节点个数
  - dfs遍历：知左子树节点数，知右子树节点数，则也可知将自己这颗子树删去后剩余全树的节点数
- 所以可知，一次树的dfs遍历即可求出各节点删除后各连通块中点数最大的值

```
int n;
int h[N], e[N<<1], ne[N<<1], idx;
bool flags[N];

int dfs(int u) {
    flags[u] = true;

    int sum = 1, res = 0;
    for (int i = h[u]; i != -1; i = ne[i]) {
        int j = e[i];   // for each subtree
        if (!flags[j]) {
            int s = dfs(j);
            res = max(res, s);
            sum += s;
        }
    }
    res = max(res, n - sum);
    ans = min(ans, res);
    return sum;
}
```

【bfs】有重边和自环的图，求从1号点走到n号点的最短距离

```
int n, m; // vertex & edge
int h[N], e[N], ne[N], idx;
int q[N]; // for bfs
int d[N]; // distance

int bfs() {
    int head = 0, tail = 0;
    q[0] = 1;

    memset(d, -1, sizeof d);
    d[1] = 0;
    while (head <= tail) {
        int t = q[head++];
        for (int i = h[t]; i != -1; i = ne[i]) {
            int j = e[i];
            if (d[j] == -1) { // not detect yet
                d[j] = d[t] + 1;
                q[++tail] = j;
            }
        }
    }
    return d[n];
}
```