

3.13 课堂笔记 3

课后作业：

- 1、从 $1 \sim n$ 中($n < 10$)个整数排成一行后随机打乱顺序，输出所有可能的选择方案。
- 2、从 $1 \sim n$ 中($n < 20$)个整数中随机选取 m 个，输出所有可能的选择方案。
- 3、从 $1 \sim n$ 中($n < 10$)个整数排成一行后随机打乱顺序，输出所有可能的选择方案。
- 4、课堂上没有写完的课堂练习

上次课后作业答案：

1.

1、互质组

给定 n 个正整数，将他们分组，使得每组中任意两个数互质（两个数的公约数只有 1）。至少要分多少个组？

输入：

第一行是一个正整数 n ($1 \leq n \leq 10$)；

第二行是 n 个不大于 10000 的正整数。

输出：

一个正整数，即最少需要的组数。

样例输入：

6

14 20 33 117 143 175

样例输出：

3

代码：

```

#define maxn 10000
using namespace std;

int gcd(int a, int b){
    if(b == 0) return a;
    return gcd(b, a%b);
}

int num[12];
int group[12][12];
int groupsize[12];
int vis[12];
int ans, n;
bool pd(int i, int x){///判断是否全部互质
    for(int j = 0; j < groupsize[i]; j++){
        if(gcd(group[i][j], x) != 1){
            return false;
        }
    }
    return true;
}

void dfs(int x, int cnt){
    if(x >= n){
        ans = min(ans, cnt);
        return ;
    }
    for(int i = 0; i < n; i++){
        if(vis[i] == 1) continue;
        for(int j = 0; j < cnt; j++){
            if(pd(j, num[i])){///如果第j组可以放入num[i]
                group[j][groupsize[j]++] = num[i];
                vis[i] = 1;
                dfs(x + 1, cnt);
                vis[i] = 0;
                groupsize[j]--;
            }
        }

        ///自成一组
        group[cnt][groupsize[cnt]++] = num[i];
        vis[i] = 1;
        dfs(x + 1, cnt + 1);
        vis[i] = 0;
        groupsize[cnt]--;
    }
    return ;
}

```

```

int main() {
    cin >> n;
    ans = n;
    for(int i = 0; i < n; i++){
        cin >> num[i];
    }

    for(int i = 0; i < n; i++){
        group[0][groupsize[0]++] = num[i];
        vis[i] = 1;
        dfs(1,1);///分组进度 组数
        vis[i] = 0;
        groupsize[0]--;
    }

    cout << ans << endl;
    return 0;
}

```

2. 洛谷 P2036

```

#define maxn 5000
using namespace std;

int dp[maxn + 5];
int ans = INT_MAX;
int n;
int s[12];
int b[12];
int solve(int x){
    int s_sum = 1;
    int b_sum = 0;
    for(int i = 0; i < n; i++){
        if(((x >> i) & 1) == 1){
            s_sum *= s[i];
            b_sum += b[i];
            ///cout << ">" << endl;
        }
    }
    if(s_sum > b_sum)
        return s_sum - b_sum;
    else
        return b_sum - s_sum;
}

int main() {
    ///1024
    cin >> n;
    for(int i = 0; i < n; i++){
        cin >> s[i] >> b[i];
    }

    int ans = INT_MAX;
    for(int i = 1; i < (1<<n); i++){
        ans = min(ans, solve(i));
    }
    cout << ans << endl;
    return 0;
}

```

3. 题目：

1. 最短路径 (熟练使用广搜算法模板)

一个迷宫由R行C列格子组成, 有的格子里有障碍物, 不能走; 有的格子是空地, 可以走。

给定一个迷宫, 求从左上角走到右下角最少需要走多少步(数据保证一定能走到)。只能在水平方向或垂直方向走, 不能斜着走。

输入

第一行是两个整数, R和C, 代表迷宫的长和宽。 ($1 \leq R, C \leq 40$)

接下来是R行, 每行C个字符, 代表整个迷宫。空地格子用'.'表示, 有障碍物的格子用'#'表示。迷宫左上角和右下角都是'.'。

输出

输出从左上角走到右下角至少要经过多少步 (即至少要经过多少个空地格子)。计算步数要包括起点和终点

样例输入

```
5 5
..###
.....
.....
..###
.....
```

样例输出

```
9
```

代码:

```
#define maxn 50
using namespace std;
string maze[maxn + 5];
int dp[maxn + 5][maxn + 5];
int n, m;
int dx[] = {0, 0, 1, -1};
int dy[] = {1, -1, 0, 0};
bool pd(int x, int y){
    if(x >= 0 && y >= 0 && x < n && y < m)
        return true;
    return false;
}

void dfs(int x, int y, int cnt){
    for(int i = 0; i < 4; i++){
        int tox = x + dx[i];
        int toy = y + dy[i];
        if(!pd(tox, toy))continue;///走越界
        if(maze[tox][toy] == '#')continue;///不能走
        if(cnt + 1 < dp[tox][toy]){///更近
            dp[tox][toy] = cnt + 1;
            dfs(tox, toy, cnt + 1);
        }
    }
}

int main() {
    cin >> n >> m;
    for(int i = 0; i < n; i++){
        cin >> maze[i];
    }
    for(int i = 0; i < n; i++){
        for(int j = 0; j < m; j++){
            dp[i][j] = INT_MAX;
        }
    }
    dp[0][0] = 1;
    dfs(0, 0, 1);
    cout << dp[n - 1][m - 1] << endl;
    return 0;
}
```

4. 洛谷 P1649

```

#define maxn 110
using namespace std;

string maze[maxn + 5];
int dp[maxn + 5][maxn + 5][4];
int n;
int sx, sy, ex, ey;
int dx[] = {0, 0, -1, 1};
int dy[] = {-1, 1, 0, 0};
///上下左右
bool pd(int x, int y){
    if(x >= 0 && y >= 0 && x < n && y < n)
        return true;
    return false;
}

void dfs(int x, int y, int last, int cnt){
    for(int i = 0; i < 4; i++){
        int tox = x + dx[i];
        int toy = y + dy[i];
        int temp = 0; ///转的次数
        if(dx[i] * dx[last] + dy[i] * dy[last] == 0) temp = 1; ///90度
        if(dx[i] * dx[last] + dy[i] * dy[last] == 1) temp = 0; ///0度
        if(dx[i] * dx[last] + dy[i] * dy[last] == -1) temp = 2; ///180度
        if(!pd(tox, toy)) continue; ///走越界
        if(maze[tox][toy] == 'x') continue; ///不能走
        if(cnt + temp < dp[tox][toy][i]) { ///更新
            dp[tox][toy][i] = cnt + temp;
            dfs(tox, toy, i, cnt + temp);
        }
    }
}

int main() {
    cin >> n;
    for(int i = 0; i < n; i++){
        string pstr;
        for(int j = 0; j < n; j++){
            cin >> pstr;
            maze[i].push_back(pstr[j]);
        }
    }
    for(int i = 0; i < n; i++){
        for(int j = 0; j < n; j++){
            for(int k = 0; k < 4; k++){
                if(maze[i][j] == 'A'){
                    dp[i][j][k] = 0;
                } else {
                    dp[i][j][k] = INT_MAX;
                }
            }
            if(maze[i][j] == 'A'){
                sx = i;
                sy = j;
            }
            if(maze[i][j] == 'B'){
                ex = i;
                ey = j;
            }
        }
    }
    for(int i = 0; i < 4; i++){
        dfs(sx, sy, i, 0);
    }
    int ans = INT_MAX;
    for(int i = 0; i < 4; i++){
        ans = min(ans, dp[ex][ey][i]);
    }
    if(ans == INT_MAX)
        cout << -1 << endl;
    else
        cout << ans << endl;
    return 0;
}

```

5. 洛谷 P1451

```

#define LL long long int
using namespace std;
int vis[maxn + 5][maxn + 5];
string str[maxn + 5];
int ans = 0;
int n, m;
int px[4] = {0, 0, 1, -1};
int py[4] = {1, -1, 0, 0};
bool inrange(int x, int y){
    if(x >= 0 && y >= 0 && x < n && y < m)
        return true;
    else
        return false;
}
void dfs(int x, int y){
    for(int k = 0; k < 4; k++){
        int tox = x + px[k];
        int toy = y + py[k];
        if(inrange(tox, toy) && !vis[tox][toy] && str[tox][toy] != '0'){
            vis[tox][toy] = 1;
            dfs(tox, toy);
        }
    }
}
int main(){
    cin >> n >> m;
    for(int i = 0; i < n; i++)
        cin >> str[i];
    for(int i = 0; i < n; i++){
        for(int j = 0; j < m; j++){
            if(!vis[i][j] && str[i][j] != '0'){
                vis[i][j] = 1;
                ans++;
                dfs(i, j);
            }
        }
    }
    cout << ans << endl;
}

```

课堂练习答案

1. 洛谷 P1331

```

#define maxn 1100
using namespace std;

bool flag = true;
string maze[maxn + 5];
int vis[maxn + 5][maxn + 5];
int n, m;
bool pd(int x, int y){
    if(x >= 0 && y >= 0 && x < n && y < m)
        return true;
    return false;
}
int h, w;
int dfs_x(int x, int y){
    if(pd(x + 1, y) && maze[x + 1][y] == '#'){
        return dfs_x(x + 1, y) + 1;
    }
    return 1;
}
int dfs_y(int x, int y){
    if(pd(x, y + 1) && maze[x][y + 1] == '#'){
        return dfs_y(x, y + 1) + 1;
    }
    return 1;
}

```

```

bool solve(int x, int y, int nowh, int noww){
    for(int i = x; i < x + nowh; i++){
        for(int j = y; j < y + noww; j++){
            vis[i][j] = 1;
            if(maze[i][j] != '#'){
                return false;
            }
        }
    }

    for(int i = 0; i < nowh; i++){
        if(pd(x + i, y - 1) && maze[x + i][y - 1] == '#')
            return false;
    }///左

    for(int i = 0; i < nowh; i++){
        if(pd(x + i, y + noww) && maze[x + i][y + noww] == '#')
            return false;
    }///右

    for(int i = 0; i < noww; i++){
        if(pd(x + nowh, y + i) && maze[x + nowh][y + i] == '#')
            return false;
    }///下

    for(int i = 0; i < noww; i++){
        if(pd(x - 1, y + i) && maze[x - 1][y + i] == '#')
            return false;
    }///上

    return true;
}

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

    int ans = 0;
    for(int i = 0; i < n; i++){
        for(int j = 0; j < m; j++){
            if(maze[i][j] == '#' && !vis[i][j]){
                h = dfs_x(i, j);
                w = dfs_y(i, j);
                if(!solve(i, j, h, w)){
                    flag = false;
                    break;
                }else{
                    ans++;
                }
            }
        }
        if(!flag)break;
    }

    if(flag){
        cout << "There are " << ans << " ships."<< endl;
    }else{
        cout << "Bad placement." << endl;
    }
    return 0;
}

```


2. 洛谷 P3956

```

#define maxn 500
using namespace std;

int color[105][105];
int n, m;
bool inrange(int i, int j){
    if(i >= 1 && i <= n && j >= 1 && j <= n)
        return true;
    else
        return false;
}

int dis[105][105];
int vis[105][105];
struct Point{
    int cost;
    int x, y;
    Point(int x, int y, int cost){
        this->x = x;
        this->y = y;
        this->cost = cost;
    }
    friend bool operator<(const Point &a, const Point &b){
        return a.cost > b.cost;
    }
};

int px[4] = {0, 0, 1, -1};
int py[4] = {1, -1, 0, 0};

void bfs(){
    for(int i = 1; i <= n; i++){
        for(int j = 1; j <= n; j++){
            dis[i][j] = INT_MAX;
            vis[i][j] = 0;
        }
    }
    priority_queue<Point> Q;
    Q.push(Point(1, 1, 0));
    dis[1][1] = 0;
    while(!Q.empty()){
        int nowx = Q.top().x;
        int nowy = Q.top().y;
        int nowcost = Q.top().cost;
        Q.pop();
        vis[nowx][nowy] = 1;
        if(nowx == n && nowy == n){
            break;
        }
        for(int i = 0; i < 4; i++){
            int tox = nowx + px[i];
            int toy = nowy + py[i];
            if(inrange(tox, toy) && !vis[tox][toy]){
                if(color[tox][toy] == 0){//空白
                    if(tox == n && toy == n){
                        if(nowcost + 2 < dis[tox][toy]){
                            dis[tox][toy] = nowcost + 2;
                            Q.push(Point(tox, toy, dis[tox][toy]));
                        }
                        continue;
                    }
                }
                for(int j = 0; j < 4; j++){
                    int totox = tox + px[j];
                    int totoy = toy + py[j];
                    if(inrange(totox, totoy) && !vis[totox][totoy] && color[totox][totoy] != 0){

```


