1 prim pri ed POJ 1251 Jungle Roads

1. **const** **int** maxn = 30010;
2. **int** n, m;
3. **int** par[maxn];
5. **int** Find(**int** x){
6. **if**(par[x] == x)
7. **return** x;
8. **return** par[x] = Find(par[x]);
9. }
11. **void** Union(**int** x, **int** y){
12. par[Find(x)] = Find(y);
13. }
15. **void** init(){
16. **for**(**int** i = 0; i <= n; ++i){
17. par[i] = i;
18. }
19. }
21. **int** arr[maxn];
22. **int** cot = 0;
24. **int** main(){
25. **while**(~scanf("%d%d", &n, &m)){
26. **if**(!n && !m)
27. **return** 0;
28. **int** ans = 0;
29. **int** k;
30. init();
31. **for**(**int** i = 0; i < m; ++i){
32. scanf("%d", &k);
33. **for**(**int** j = 0; j < k; ++j){
34. scanf("%d", &arr[cot++]);
35. **if**(j){
36. // cout<<arr[cot-2]<<"        "<<arr[cot-1]<<endl;
37. Union(arr[cot-2], arr[cot-1]);//cot==arr.size()
38. }
39. }
40. }
41. **int** p = Find(0);
42. sort(arr, arr+n);
43. **int** res = unique(arr, arr+n) - arr;
44. **for**(**int** i = 0; i < res; ++i){
45. **if**(Find(arr[i]) == p){
46. ans++;
47. }
48. }
49. printf("%d\n", ans);
50. }
51. **return** 0;
52. }

4 D ZOJ 1586 QS Network

1. **const** **int** maxn = 110;
2. **const** **int** inf = 0x3f3f3f3f;
4. **int** n;
5. **int** G[maxn][maxn];
6. **int** vst[maxn];
7. **int** d[maxn];
8. **int** sum;
10. **void** Prim(**int** s){
11. **for**(**int** i = 0; i <= n; ++i){
12. vst[i] = 0;
13. d[i] = G[s][i];
14. }
15. vst[s] = 1;
16. **for**(**int** i = 0; i < n-1; ++i){
17. **int** Min = inf;
18. **int** k = s;
19. **for**(**int** j = 1; j <= n; ++j){
20. **if**(!vst[j] && Min > d[j]){
21. Min = d[j];
22. k = j;
23. }
24. }
25. vst[k] = 1;
26. sum += Min;
27. **for**(**int** j = 1; j <= n; ++j){
28. **if**(!vst[j] && d[j] > G[k][j])
29. d[j] = G[k][j];
30. }
31. }
32. }
34. **int** main(){
35. scanf("%d", &n);
36. **for**(**int** i = 1; i <= n; ++i){
37. **for**(**int** j = 1; j <= n; ++j){
38. **int** v;
39. scanf("%d", &v);
40. G[i][j] = v;
41. }
42. }
43. **int** q;
44. scanf("%d", &q);
45. **for**(**int** i = 0; i < q; ++i){
46. **int** x, y;
47. scanf("%d%d", &x, &y);
48. G[x][y] = G[y][x] = 0;
49. }
51. Prim(1);
52. printf("%d\n", sum);
53. **return** 0;
54. }

14 N HDU 1875 畅通工程再续

1. **const** **int** maxn = 110;
2. **const** **int** inf = 0x3f3f3f3f;
4. **int** n;
5. **double** G[maxn][maxn];
6. **int** vst[maxn];
7. **double** d[maxn];
8. **double** sum;
9. vector<pair<**int**, **int**> > point;
11. **void** Prim(**int** s){
12. **for**(**int** i = 0; i < n; ++i){
13. vst[i] = 0;
14. d[i] = G[s][i];
15. }
16. vst[s] = 1;
17. **for**(**int** i = 0; i < n-1; ++i){
18. **double** Min = inf;
19. **int** k = s;
20. **for**(**int** j = 0; j < n; ++j){
21. **if**(!vst[j] && Min > d[j]){
22. Min = d[j];
23. k = j;
24. }
25. }
26. vst[k] = 1;
27. sum += Min;
28. **for**(**int** j = 0; j < n; ++j){
29. **if**(G[k][j])
30. **if**(!vst[j] && d[j] > G[k][j])
31. d[j] = G[k][j];
32. }
33. }
34. }
36. **double** dis(**double** x1, **double** y1, **double** x2, **double** y2){
37. **return** sqrt( 1.0\*(x1-x2)\*(x1-x2) + 1.0\*(y1-y2)\*(y1-y2) );
38. }
40. **void** init(){
41. sum = 0;
42. point.clear();
43. **for**(**int** i = 0; i < n; ++i)
44. **for**(**int** j = 0; j < n; ++j)
45. G[i][j] = inf;
46. }
48. **int** main(){
49. **int** t;
50. scanf("%d", &t);
51. **while**(t--){
52. scanf("%d", &n);
53. init();
54. **for**(**int** i = 0; i < n; ++i){
55. **int** x, y;
56. scanf("%d%d", &x, &y);
57. point.push\_back(make\_pair(x, y));
58. }
59. **for**(**int** i = 0; i < point.size(); ++i){
60. **for**(**int** j = i+1; j < point.size(); ++j){
61. **double** v = dis(point[i].first, point[i].second, point[j].first, point[j].second);
62. **if**( v >= 10.0 && v <= 1000.0){
63. G[i][j] = G[j][i] = v\*100;
64. }
65. }
66. }
67. Prim(1);
68. **if**(sum < inf)
69. printf("%.1f\n", sum);
70. **else**
71. printf("oh!\n");
72. }
73. **return** 0;
74. }

8 POJ 1751 Highways

1. **const** **int** maxn = 800;
2. **const** **int** inf = 0x3f3f3f3f;
4. **int** n, m;
5. **int** G[maxn][maxn];
6. **int** vst[maxn];
7. **int** d[maxn];
8. **int** sum;
9. vector<pair<**int**,**int**> > point;
10. **int** path[maxn];
12. **void** Prim(**int** s) {
13. **for**(**int** i = 0; i <= n; ++i) {
14. vst[i] = 0;
15. d[i] = G[s][i];
16. }
17. vst[s] = 1;
18. **for**(**int** i = 0; i < n-1; ++i) {
19. **int** Min = inf;
20. **int** k = s;
21. **for**(**int** j = 1; j <= n; ++j) {
22. **if**(!vst[j] && Min > d[j]) {
23. Min = d[j];
24. k = j;
25. }
26. }
27. vst[k] = 1;
28. sum += Min;
29. **if**(Min){
30. printf("%d %d\n", path[k], k);
31. }
32. **for**(**int** j = 1; j <= n; ++j) {
33. **if**(!vst[j] && d[j] > G[k][j]) {
34. d[j] = G[k][j];
35. path[j] = k;
36. }
37. }
38. }
39. }
41. **int** dis(**int** x1, **int** y1, **int** x2, **int** y2) {
42. **return** (x1-x2)\*(x1-x2) + (y1-y2)\*(y1-y2);
43. }
45. **int** main() {
46. **int** x, y;
47. scanf("%d", &n);
48. sum = 0;
50. **for**(**int** i = 0; i <= n; ++i) {
51. path[i] = 1;
52. **for**(**int** j = 0; j <= n; ++j) {
53. G[i][j] = inf;
54. }
55. }
56. **for**(**int** i = 0; i < n; ++i) {
57. scanf("%d%d", &x, &y);
58. point.push\_back(make\_pair(x, y));
59. }
60. scanf("%d", &m);
61. **for**(**int** i = 0; i < m; ++i) {
62. scanf("%d%d", &x, &y);
63. G[x][y] = G[y][x] = 0;
64. }
66. **for**(**int** i = 0; i < n; ++i) {
67. **for**(**int** j = i+1; j < n; ++j) {
68. **if**(G[i+1][j+1]) {//死于下标
69. **int** v = dis(point[i].first, point[i].second, point[j].first, point[j].second);
70. G[i+1][j+1] = G[j+1][i+1] = v;//死于下标
71. }
72. }
73. }
75. Prim(1);
76. **return** 0;
77. }

10 J POJ 3026 Borg Maze

1. **const** **int** maxn = 110;
2. **const** **int** inf = 0x3f3f3f3f;
4. **int** t, m, n;
5. **char** mp[maxn][maxn];
6. **int** alens[maxn][maxn];
7. **int** time[maxn][maxn];
8. **int** step[][2] = {{1, 0}, {-1, 0}, {0, 1}, {0, -1}};
9. **int** G[maxn][maxn];
11. **typedef** pair<**int**,**int**> Point;
13. **bool** outBorder(**int** x, **int** y){
14. **if**(x < 0 || x >= n || y < 0 || y >= m)
15. **return** **true**;
16. **return** **false**;
17. }
19. **void** bfs(**int** x, **int** y){
20. memset(time, -1, **sizeof**(time));// 置为-1代替了vst功能。
21. queue<Point> q;
22. q.push(make\_pair(x, y));
23. time[x][y] = 0;
24. **while**(!q.empty()){
25. Point p = q.front();
26. q.pop();
27. **int** xx = p.first;
28. **int** yy = p.second;
29. **if**(alens[xx][yy]){
30. G[ alens[x][y] ][ alens[xx][yy] ] = time[xx][yy];
31. }
32. **for**(**int** i = 0; i < 4; ++i){
33. **int** cur\_x = xx + step[i][0];
34. **int** cur\_y = yy + step[i][1];// 代替了vst功能。
35. **if**(time[cur\_x][cur\_y] != -1 || mp[cur\_x][cur\_y] == '#' || outBorder(cur\_x, cur\_y))
36. **continue**;
37. time[cur\_x][cur\_y] = time[xx][yy] + 1;
38. q.push(make\_pair(cur\_x, cur\_y));
39. }
40. }
42. }
44. **bool** vst[maxn];
45. **int** d[maxn];
46. **int** sum;
48. **void** Prim(**int** s, **int** n){
49. **for**(**int** i = 0; i <= n; ++i){
50. vst[i] = 0;
51. d[i] = G[s][i];
52. }
53. vst[s] = 1;
54. **for**(**int** i = 0; i < n-1; ++i){
55. **int** Min = inf;
56. **int** k = s;
57. **for**(**int** j = 0; j < n; ++j){
58. **if**(!vst[j] && Min > d[j]){
59. Min = d[j];
60. k = j;
61. }
62. }
63. vst[k] = 1;
64. sum += Min;
65. **for**(**int** j = 0; j < n; ++j){
66. **if**(!vst[j] && d[j] > G[k][j])
67. d[j] = G[k][j];
68. }
69. }
70. }
72. **int** main(){
73. scanf("%d", &t);
74. **while**(t--){
75. **int** cot = 0;
76. sum = 0;
77. memset(alens, -1, **sizeof**(alens));
78. scanf("%d%d", &m, &n);
79. gets(mp[0]);
80. **for**(**int** i = 0; i < n; ++i){
81. gets(mp[i]);
82. **for**(**int** j = 0; j < m; ++j){
83. **if**(mp[i][j] == 'S' || mp[i][j] == 'A'){
84. alens[i][j] = cot++;
85. }
86. }
87. }
88. **for**(**int** i = 0; i < n; ++i)
89. **for**(**int** j = 0; j < m; ++j)
90. **if**(alens[i][j] != -1)
91. bfs(i, j);
93. Prim(0, cot);
94. printf("%d\n", sum);
95. }
96. **return** 0;
97. }

11 H POJ 1679 The Unique MST

1. **const** **int** inf = 0x3f3f3f3f;
2. **const** **int** maxn = 110;
4. **int** n, m, G[maxn][maxn];
5. **int** d[maxn];
6. **bool** vst[maxn];
7. **int** maxD[maxn][maxn];   //MST中从i->j的最大权值
8. **int** pre[maxn];          //某一点父节点
9. **bool** mst[maxn][maxn];   //该点是否已经在MST中
10. **typedef** pair<**int**, **int**> P;
12. **int** Prim(**int** s) {
13. **for**(**int** i = 0; i <= n; ++i){
14. d[i] = inf;
15. vst[i] = 0;
16. pre[i] = s;
17. **for**(**int** j = 0; j <= n; ++j){
18. mst[i][j] = 0;
19. maxD[i][j] = 0;
20. }
21. }
22. fill(pre, pre+maxn, s);
23. priority\_queue<P, vector<P>, greater<P> > q;
24. q.push(P(d[s]=0, s));
25. **int** res = 0;
26. **while**(!q.empty()) {
27. P cur = q.top();
28. q.pop();
29. **int** u = cur.second;
30. **if**(vst[u]) **continue**;
31. vst[u] = **true**, res += d[u];
32. mst[u][pre[u]] = mst[pre[u]][u] = **true**; //加入到MST中
33. **for**(**int** v = 1; v <= n; ++v) {
34. **if**(vst[v] && G[u][v]<inf)          //只更新MST中的
35. maxD[u][v] = maxD[v][u] = max(maxD[pre[u]][u], d[u]);
36. **if**(G[u][v] < d[v]) {
37. d[v] = G[u][v];
38. pre[v] = u;                     //更新父节点
39. q.push(P(d[v], v));
40. }
41. }
42. }
43. **return** res;
44. }
46. **int** main() {
47. **int** T, x, y, z;
48. scanf("%d", &T);
49. **while**(T--) {
50. //        fill(G[0], G[0]+maxn\*maxn, inf);
51. memset(G, inf, **sizeof**(G));
52. scanf("%d%d", &n, &m);
53. **while**(m--) {
54. scanf("%d%d%d", &x, &y, &z);
55. G[x][y] = G[y][x] = z;
56. }
57. **int** ans = Prim(1);
58. **bool** flag = **false**;  //次小生成树是否等于最小生成树
59. **for**(**int** u = 1; u <= n && !flag; ++u) {
60. **for**(**int** v = 1; v <= n; ++v) {
61. **if**(mst[u][v] || G[u][v]==inf)
62. **continue**;
63. **if**(G[u][v] == maxD[u][v]) {
64. flag = **true**;
65. **break**;
66. }
67. }
68. }
69. **if**(flag)
70. printf("Not Unique!\n");
71. **else**
72. printf("%d\n", ans);
73. }
74. **return** 0;
75. }