专题五 并查集 1 POJ 2236 Wireless Network

1. **const** **int** maxn = 1010;
2. **int** n;
3. **double** d;
4. **int** par[maxn];
5. **int** parr[maxn];
6. **int** cot = 0;
7. **int** x[maxn];
8. **int** y[maxn];
10. **double** dis(**int** x1,**int** y1, **int** x2, **int** y2){
11. **return** sqrt(**double**((**double**) (x1-x2)\*(x1-x2) + (**double**)(y1-y2)\*(y1-y2)  ));
12. }
13. **int** Find(**int** x){
14. **if**(par[x] != x)
15. par[x] = Find(par[x]);
16. **return** par[x];
17. }
18. **void** Union(**int** x, **int** y){ par[Find(x)] = Find(y);  }
20. **int** main(){
21. scanf("%d%lf", &n, &d);
22. **for**(**int** i = 0; i <= n; ++i)
23. par[i] = i;
24. **for**(**int** i = 0; i < n; ++i){
25. scanf("%d %d", &x[i], &y[i]);
26. }
28. **char** ch[10];
29. **int** p, q;
30. **while**(~scanf("%s", ch)){
31. **switch**(ch[0]){
32. **case** 'O':
33. scanf("%d", &p);
34. parr[cot++] = --p;
35. **for**(**int** i = 0; i < cot; ++i){
36. **if**(parr[i] != p){
37. **double** distance = dis(x[parr[i]], y[parr[i]],x[p],y[p]);
38. **if**(distance  <= d)
39. Union(parr[i], p);
40. }
41. }
42. **break**;
43. **case** 'S':
44. scanf("%d%d", &p, &q);
45. **if**( Find(--p) == Find(--q) )
46. printf("SUCCESS\n");
47. **else**{
48. printf("FAIL\n");
49. }
50. **break**;
51. }
52. }
53. **return** 0;
54. }

专题五 并查集 2 POJ 1611 The Suspects

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]);
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. }

专题五 并查集 3 HDU 3038 How Many Answers Are Wrong

1. **const** **int** maxn = 1010;
2. **int** t, n, m;
3. **int** par[maxn];
5. **void** init(){
6. **for**(**int** i = 0; i <= n; ++i)
7. par[i] = i;
8. }
10. **int** Find(**int** x){
11. **if**(par[x] == x)
12. **return** x;
13. **return** par[x] = Find(par[x]);
14. }
16. **void** Union(**int** x, **int** y){
17. par[Find(x)] = Find(y);
18. }
20. **int** res[maxn];
22. **int** main(){
23. scanf("%d", &t);
24. **while**(t--){
25. memset(res, 0, **sizeof**(res));
26. scanf("%d%d",&n,&m);
27. init();
28. **int** p,q;
29. **for**(**int** i = 0; i < m; ++i){
30. scanf("%d%d", &p, &q);
31. Union(p, q);
32. }
33. **int** ans = 0;
34. **for**(**int** i = 1; i <= n; ++i){
35. **if**(!res[ Find(i)  ]){
36. ans ++;
37. res[Find(i)] = 1;
38. }
39. **else**{
40. res[Find(i)] ++ ;
41. }
42. }
43. printf("%d\n", ans);
44. }
45. **return** 0;
46. }

专题五 并查集 4 HDU 3038 How Many Answers Are Wrong

1. **typedef** **long** **long** ll;
2. **const** **int** maxn = 200005;
4. **int** par[maxn];
5. ll d[maxn];
6. **int** ans;
8. **int** Find(**int** x){
9. **if**(x != par[x]){
10. **int** k = par[x];
11. par[x] = Find(par[x]);
12. d[x] += d[k];
13. }
14. **return** par[x];
15. }
17. **void** init(**int** n){
18. memset(d,0,**sizeof**(d));
19. **for**(**int** i = 0; i <= n; i++)
20. par[i] = i;
21. }
23. **void** Union(**int** x, **int** y, **int** v){
24. **int** xx = Find(x);
25. **int** yy = Find(y);
26. **if**(xx == yy){
27. **if**(d[x] + v != d[y])
28. ans++;
29. }
30. **else**{
31. par[yy] = xx;
32. d[yy] = d[x] - d[y]  + v;
33. }
34. }
36. **int** main(){
37. **int** n, m;
38. **while**(scanf("%d%d", &n, &m) != EOF){
39. ans = 0;
40. init(n);
41. **int** x, y, v;
42. **for**(**int** i = 0; i < m; i++){
43. scanf("%d%d%d", &x, &y, &v);
44. x -= 1;
45. Union(x, y, v);
46. }
47. printf("%d\n", ans);
48. }
49. **return** 0;
50. }

专题五 并查集 5 POJ 1182 食物链

1. **const** **int** maxn = 50010;
2. **int** par[maxn\*3];
3. **int** n, m;
4. **int** ans;
6. **void** init(){
7. **for**(**int** i = 0; i <= 3\*n; ++i)
8. par[i] = i;
9. }
11. **int** Find(**int** x){
12. **if**(par[x] == x)
13. **return** x;
14. **return** par[x] = Find(par[x]);
15. }
17. **void** Union(**int** x, **int** y){
18. par[Find(x)] = Find(y);
19. }
21. **int** main(){
22. scanf("%d%d", &n, &m);
23. ans = 0;
24. init();
25. **for**(**int** i = 0; i < m; ++i){
26. **int** v, a, b;
27. scanf("%d%d%d", &v, &a, &b);
28. **if**(a < 1 || a > n || b < 1 || b > n){
29. ans++;
30. **continue**;
31. }
32. **if**(v == 2 && a == b){
33. ans++;
34. **continue**;
35. }
36. **if**(v == 1){
37. **if**(Find(a) == Find(b+n) || Find(b) == Find(a+n))
38. ans++;
39. **else**{
40. Union(a, b);
41. Union(a+n, b+n);
42. Union(a+2\*n, b+2\*n);
43. }
44. }
45. **else** **if**(v == 2){
46. **if**(Find(a) == Find(b) || Find(b) == Find(a+n))
47. ans++;
48. **else**{
49. Union(a, b+n);
50. Union(a+n, b+2\*n);
51. Union(a+2\*n, b);
52. }
53. }
54. }
55. printf("%d", ans);
56. **return** 0;
57. }

专题五 并查集 6 POJ 1417 True Liars

1. **const** **int** maxn = 1010;
2. **const** **int** maxm = 610;
4. **int** par[maxn],Rank[maxn],p[maxn],vst[maxn];
5. **int** w0[maxn],w1[maxn];
6. **int** dp[maxm][maxm];
7. **int** n,p1,p2;
9. **int** Find(**int** x){
10. **if**(x!=par[x]){
11. **int** tmp = par[x];
12. par[x] = Find(par[x]);
13. Rank[x] = Rank[x]^Rank[tmp];
14. }
15. **return** par[x];
16. }
18. **void** Union(**int** x,**int** y,**int** k){
19. **int** xx =Find(x);
20. **int** yy = Find(y);
21. **if**(xx!=yy){
22. par[xx] = yy;
23. Rank[xx]=Rank[x]^Rank[y]^k;
24. }
25. }
27. **int** main(){
28. **while**(scanf("%d%d%d",&n,&p1,&p2)){
29. **if**(n+p1+p2 == 0)
30. **break**;
31. **for**(**int** i=1;i<=p1+p2;i++){
32. par[i] = i;
33. Rank[i] = 0;
34. vst[i] = 0;
35. w0[i] = w1[i] = 0;
36. }
37. memset(dp,0,**sizeof**(dp));
38. **for**(**int** i=0;i<n;i++){
39. **int** a,b;**char** ch[5];
40. scanf("%d%d%s",&a,&b,ch);
41. **if**(ch[0]=='y')
42. Union(a,b,0);
43. **else**
44. Union(a,b,1);
45. }
46. **int** cnt = 1;//组数
47. **for**(**int** i=1;i<=p1+p2;i++){
48. **if**(!vst[i]){
49. **int** xx = Find(i);
50. **for**(**int** j=i;j<=p1+p2;j++){
51. **if**(Find(j) == xx && !vst[j]){
52. vst[j] = 1;
53. **if**(Rank[j] == 0)
54. w0[cnt]++;//第i组中与根同类的数量
55. **else**
56. w1[cnt]++;//第i组中与根异类的数量
57. }
58. }
59. p[cnt] = xx;//第i组的根为xx
60. cnt++;
61. }
62. }
64. dp[0][0]=1;
65. **for**(**int** i=1; i < cnt; ++i){
66. **int** Min=min(w0[i], w1[i]);
67. **for**(**int** j = p1; j >= Min; --j){
68. **if**(dp[i-1][j-w0[i]]){
69. dp[i][j]+=dp[i-1][j-w0[i]];
70. }
71. **if**(dp[i-1][j-w1[i]]){
72. dp[i][j]+=dp[i-1][j-w1[i]];
73. }
74. }
75. }
77. **if**(dp[cnt-1][p1]!=1){
78. printf("no\n");
79. **continue**;
80. }
81. **int** ans[maxn];
82. **int** num = 0;
83. **int** Mp = p1;//神圣人的数量
84. **for**(**int** i=cnt-1;i>=1;i--){
85. **if**(dp[i-1][Mp-w0[i]]==1){//w0存的全是权值为0的
86. **for**(**int** j=1;j<=p1+p2;j++){
87. **if**(Find(j)==p[i]&&Rank[j]==0)
88. ans[num++]=j;
89. }
90. Mp-=w0[i];
91. }
92. **else** {//w1存的全是权值为1的
93. **for**(**int** j=1;j<=p1+p2;j++){
94. **if**(Find(j)==p[i]&&Rank[j]==1)
95. ans[num++]=j;
96. }
97. Mp-=w1[i];
98. }
99. }
100. sort(ans,ans+num);
101. **for**(**int** i=0;i<num;i++)
102. printf("%d\n",ans[i]);
103. puts("end");
104. }
105. **return** 0;
106. }

专题五 并查集 7 POJ 1456 Supermarket

1. **const** **int** maxn = 10010;
2. **int** par[maxn];
4. **struct** node{
5. **int** v;
6. **int** t;
7. }arr[maxn];
9. **void** init(){
10. **for**(**int** i = 0; i <= maxn; ++i)
11. par[i] = i;
12. }
14. **int** Find(**int** x){
15. **if**(par[x] == x)
16. **return** x;
17. **return** par[x] = Find(par[x]);
18. }
20. **bool** cmp(node a, node b){
21. **return** a.v > b.v;
22. }
24. **int** main(){
25. **int** n;
26. **while**(cin>>n){
27. **int** ans = 0;
28. init();
29. **for**(**int** i = 0; i < n; ++i){
30. cin>>arr[i].v>>arr[i].t;
31. }
32. sort(arr, arr+n, cmp);
33. **for**(**int** i = 0; i < n; ++i){
34. **int** k = Find(arr[i].t);
35. **if**(k){
36. ans += arr[i].v;
37. par[k] = k - 1;
38. }
39. }
40. cout<<ans<<endl;
41. }
42. **return** 0;
43. }

专题五 并查集 8 POJ 1733 Parity game

1. #define Fi(i,a,b) for(int i=a;i<=b;i++)
2. **const** **int** N=4e4+7;
3. **int** n,m,a[N],d[N],fa[N],ans,cnt;
4. **struct** Node {
5. **int** l,r,v;
6. } p[N];
7. **void** ready() {
8. cin>>n>>m;
9. **char** str[5];
10. Fi(i,1,m) {
11. scanf("%d%d%s",&p[i].l,&p[i].r,str);
12. p[i].v=(str[0]=='e'?0:1);
13. a[++cnt]=p[i].l-1;
14. a[++cnt]=p[i].r;
15. }
16. sort(a+1,a+1+cnt);
17. n=unique(a+1,a+cnt+1)-a-1;
18. }
19. **inline** **int** find(**int** x) {
20. **return** fa[x]==x?x:fa[x]=find(fa[x]);
21. }
22. **int** main() {
23. ready();
24. Fi(i,1,2\*n)fa[i]=i;
25. Fi(i,1,m) {
26. **int** x=lower\_bound(a+1,a+n+1,p[i].l-1)-a;
27. **int** y=lower\_bound(a+1,a+n+1,p[i].r)-a;
28. **int** xo=x,xe=x+n;
29. **int** yo=y,ye=y+n;
30. **if**(p[i].v==0) {
31. **if**(find(xo)==find(ye)) {
32. cout<<i-1;
33. **return** 0;
34. }
35. fa[find(ye)]=find(xe);
36. fa[find(yo)]=find(xo);
37. } **else** {
38. **if**(find(xo)==find(yo)) {
39. cout<<i-1;
40. **return** 0;
41. }
42. fa[find(yo)]=find(xe);
43. fa[find(ye)]=find(xo);
44. }
45. }
46. cout<<m;
47. **return** 0;
48. }

专题五 并查集 9 POJ 1984 Navigation Nightmare

1. **const** **int** maxn = 40010;
3. **struct** Data{
4. **int** x, y, v;
5. **char** ch;
6. Data(){}
7. Data(**int** xx, **int** yy, **int** vv, **char** c):x(xx), y(yy), v(vv), ch(c){}
8. }data[maxn];
10. **struct** Query{
11. **int** x, y, t, num;
12. Query(){}
13. Query(**int** xx, **int** yy, **int** tt, **int** nn):x(xx), y(yy), t(tt), num(nn){}
14. }query[maxn];
16. **int** cmp(Query q1, Query q2){
17. **return** q1.t < q2.t;
18. }
20. **int** par[maxn],ew[maxn],ns[maxn];
21. **int** n, m, q;
23. **int** Find(**int** x){
24. **if**(x!=par[x]){
25. **int** tmp = par[x];
26. par[x] = Find(par[x]);
27. ew[x] += ew[tmp];
28. ns[x] += ns[tmp];
29. }
30. **return** par[x];
31. }
33. **void** Union(**int** x, **int** y, **int** k, **char** ch){
34. **int** xx = Find(x);
35. **int** yy = Find(y);
36. **if**(xx != yy){
37. par[xx] = yy;
38. **if**(ch == 'E' || ch == 'W'){
39. **if**(ch == 'E')   ew[xx] = ew[y]-ew[x] + k;//东为正
40. **if**(ch == 'W')   ew[xx] = ew[y]-ew[x] - k;
41. ns[xx] = ns[y] - ns[x];//不增加,只更新
42. }
43. **else**{
44. **if**(ch == 'N')   ns[xx] = ns[y]-ns[x] + k;//北为正
45. **if**(ch == 'S')   ns[xx] = ns[y]-ns[x] - k;
46. ew[xx] = ew[y] - ew[x];//不增加,只更新
47. }
49. }
50. }
52. map<**int**, **int**> mp;
54. **int** main(){
55. **for**(**int** i = 0; i < maxn; ++i)
56. par[i] = i;
57. scanf("%d%d", &n, &m);
58. **int** x, y, v;
59. **char** ch[5];
60. **for**(**int** i = 0; i < m; ++i){
61. scanf("%d%d%d%s", &x, &y, &v, ch);
62. data[i] = Data(x, y, v, ch[0]);
63. }
64. scanf("%d", &q);
65. **for**(**int** i = 0; i < q; ++i){
66. scanf("%d%d%d", &x, &y, &v);
67. query[i] = Query(x, y, v, i);
68. }
70. sort(query, query+q, cmp);
72. **int** time = 0;
74. **for**(**int** i = 0; i < q; ++i){
75. **int** t = query[i].t;
76. **while**(time < t){
77. Union(data[time].x, data[time].y, data[time].v, data[time].ch);
78. time ++;
79. }
81. x = query[i].x;
82. y = query[i].y;
83. **int** xx, yy;
84. xx = Find(x);
85. yy = Find(y);
87. **if**(xx != yy)
88. v = -1;
89. **else**
90. v = abs(ew[x]-ew[y]) + abs(ns[x]-ns[y]);
91. mp[query[i].num] = v;
92. }
94. **for**(**int** i = 0; i < q; ++i)
95. printf("%d\n", mp[i]);
97. **return** 0;
98. }

专题五 并查集 10 扩展域 POJ 2492 A Bug's Life

1. **const** **int** maxn = 2010;
2. **int** par[maxn<<1];
3. **int** t, m, n;
4. **int** flag;
6. **int** Find(**int** x){
7. **if**(par[x] == x)
8. **return** x;
9. **return** par[x] = Find(par[x]);
10. }
12. **void** Union(**int** x, **int** y){
13. **int** xx = Find(x);
14. **int** yy = Find(y);
15. **if**(xx != yy)//不同性
16. par[xx] = yy;
17. }
19. **void** init(){
20. flag = 0;
21. **for**(**int** i = 0; i <= n\*2; ++i)
22. par[i] = i;
23. }
25. **int** main(){
26. scanf("%d", &t);
27. **for**(**int** T = 1; T <= t; ++T){
28. scanf("%d%d", &n, &m);
29. init();
30. **int** x, y;
31. **for**(**int** i = 0; i < m; ++i){
32. scanf("%d%d", &x, &y);
33. **int** xx = Find(x);
34. **int** yy = Find(y);
35. **if**(xx == yy)
36. flag ++;
37. **else**{
38. Union(x, y+n);
39. Union(x+n, y);
40. }
41. }
42. **if**(flag)
43. printf("Scenario #%d:\nSuspicious bugs found!\n\n", T);
44. **else**
45. printf("Scenario #%d:\nNo suspicious bugs found!\n\n", T);
46. }
47. }

专题五 并查集 10 种类并查集 POJ 2492 A Bug's Life

1. **const** **int** maxn = 2010;
2. **int** par[maxn<<1];
3. **int** t, m, n;
4. **int** flag;
6. **int** Find(**int** x){
7. **if**(par[x] == x)
8. **return** x;
9. **return** par[x] = Find(par[x]);
10. }
12. **void** Union(**int** x, **int** y){
13. par[Find(x)] = Find(y);
14. }
16. **void** init(){
17. flag = 0;
18. **for**(**int** i = 0; i <= n\*2; ++i)
19. par[i] = i;
20. }
22. **int** main(){
23. scanf("%d", &t);
24. **for**(**int** T = 1; T <= t; ++T){
25. scanf("%d%d", &n, &m);
26. init();
27. **int** x, y;
28. **for**(**int** i = 0; i < m; ++i){
29. scanf("%d%d", &x, &y);
30. **int** xx = Find(x);
31. **int** yy = Find(y);
32. **if**(xx == yy)
33. flag ++;
34. **else**{
35. Union(x, y+n);
36. Union(x+n, y);
37. }
38. }
39. **if**(flag)
40. printf("Scenario #%d:\nSuspicious bugs found!\n\n", T);
41. **else**
42. printf("Scenario #%d:\nNo suspicious bugs found!\n\n", T);
43. }
44. }

专题五 并查集 11 POJ 2912 Rochambeau

1. **const** **int** maxn = 2010;
2. **int** par[maxn<<1];
3. **int** t, m, n;
4. **int** flag;
6. **int** Find(**int** x){
7. **if**(par[x] == x)
8. **return** x;
9. **return** par[x] = Find(par[x]);
10. }
12. **void** Union(**int** x, **int** y){
13. **int** xx = Find(x);
14. **int** yy = Find(y);
15. **if**(xx != yy)//不同性
16. par[xx] = yy;
17. }
19. **void** init(){
20. flag = 0;
21. **for**(**int** i = 0; i <= n\*2; ++i)
22. par[i] = i;
23. }
25. **int** main(){
26. scanf("%d", &t);
27. **for**(**int** T = 1; T <= t; ++T){
28. scanf("%d%d", &n, &m);
29. init();
30. **int** x, y;
31. **for**(**int** i = 0; i < m; ++i){
32. scanf("%d%d", &x, &y);
33. **int** xx = Find(x);
34. **int** yy = Find(y);
35. **if**(xx == yy)
36. flag ++;
37. **else**{
38. Union(x, y+n);
39. Union(x+n, y);
40. }
41. }
42. **if**(flag)
43. printf("Scenario #%d:\nSuspicious bugs found!\n\n", T);
44. **else**
45. printf("Scenario #%d:\nNo suspicious bugs found!\n\n", T);
46. }
47. }

专题五 并查集 12 ZOJ 3261 Connections in Galaxy War

1. **const** **int** maxn = 10010;//编号范围
2. **const** **int** maxm = 50010;//操作次数
3. **const** **int** maxg = 20010;//建边数量
4. **const** **int** Hash = 10000;
6. **int** m, n, q;
7. **int** par[maxn];
8. **int** val[maxn];
9. map<**int**, **bool**> mp;//存储破坏的边
10. pair<**int**, **int**> G[maxg];//存储边
11. **int** ans[maxm];
13. **struct** Data{
14. **char** op;
15. **int** x, y;
16. }data[maxm];
18. **int** Find(**int** x){
19. **if**(par[x] == x)
20. **return** x;
21. **return** par[x] = Find(par[x]);
22. }
24. **void** Union(**int** x, **int** y){
25. **int** xx = Find(x);
26. **int** yy = Find(y);
27. **if**(val[xx] > val[yy] || (val[xx] == val[yy] && xx < yy))
28. par[yy] = xx;
29. **if**(val[xx] < val[yy] || (val[xx] == val[yy] && yy < xx))
30. par[xx] = yy;
31. }
33. **void** init(){
34. mp.clear();
35. memset(ans, 0, **sizeof**(ans));
36. **for**(**int** i = 0; i < maxn; ++i)
37. par[i] = i;
38. }
40. **int** main(){
41. **bool** flag = **false**;
42. **while**(~scanf("%d", &n)){
43. init();
44. **for**(**int** i = 0; i < n; ++i){
45. scanf("%d", &val[i]);
46. }
48. scanf("%d", &m);//m个边战前建立
49. **int** x, y;
50. **for**(**int** i = 0; i < m; ++i){
51. scanf("%d%d", &x, &y);
52. **if**(x > y)//与删除保持一致
53. swap(x, y);
54. G[i] = make\_pair(x, y);
55. }
57. scanf("%d", &q);//操作次数
58. **char** ch[10];
59. **for**(**int** i = 0; i < q; ++i){
60. scanf("%s", ch);
61. **if**(ch[0] == 'q'){//查询
62. scanf("%d", &x);
63. data[i].op = 'q';
64. data[i].x = x;
65. }
66. **else**{//删除
67. scanf("%d%d", &x, &y);
68. data[i].op = 'd';
69. **if**(x > y)
70. swap(x, y);
71. data[i].x = x;
72. data[i].y = y;
73. mp[ x \* Hash + y ] = 1;
74. }
75. }
77. **for**(**int** i = 0; i < m; ++i){
78. **if**(!mp[ G[i].first \* Hash + G[i].second ])
79. Union(G[i].first, G[i].second);
80. // else{
81. // printf(" delete %4d%4d\n", G[i].first, G[i].second);
82. // }
83. }
85. **int** cot = 0;
87. **for**(**int** i = q-1; i >= 0; --i){
88. **if**(data[i].op == 'q'){
89. x = data[i].x;
90. **int** xx = Find(x);
91. **if**(val[xx] > val[x])//选择功率大的
92. ans[cot++] = xx;
93. **else**
94. ans[cot++] = -1;
95. // printf(" ans %4d%4d%4d%4d\n", i, x, xx, ans[i]);
96. }
97. **else**{
98. Union(data[i].x, data[i].y);
99. }
100. }
101. **if**(flag)
102. printf("\n");
103. **else**
104. flag = **true**;
105. **for**(**int** i = cot-1; i >= 0; --i)
106. printf("%d\n", ans[i]);
107. }
108. **return** 0;
109. }

专题五 并查集 13 HDU 1272 小希的迷宫

1. **const** **int** maxn = 100010;
3. **int** flag;
4. **int** arr[maxn];
5. **int** par[maxn];
6. **int** vst[maxn];
8. **void** init() {
9. **for**(**int** i = 0; i < maxn; ++i){
10. par[i] = i;
11. vst[i] = 0;
12. }
13. }
15. **int** Find(**int** x) {
16. **if**(par[x] == x)
17. **return** x;
18. **return** par[x] = Find(par[x]);
19. }
21. **void** Union(**int** x, **int** y) {
22. **int** xx = Find(x);
23. **int** yy = Find(y);
24. **if**(xx == yy)
25. flag = 1;
26. par[xx] = yy;
27. }
29. **int** main() {
30. **int** x, y;
31. **while**(~scanf("%d%d", &x, &y)) {
32. **if**(x == -1 && y == -1)
33. **break**;
34. **if**(!x && !y){
35. printf("Yes\n");
36. **continue**;
37. }
38. init();
39. Union(x, y);
40. vst[x] = vst[y] = 1;
41. **while**(~scanf("%d%d", &x, &y)) {
42. **if**(!x && !y)
43. **break**;
44. vst[x] = vst[y] = 1;
45. **if**(!flag)
46. Union(x, y);
47. }
49. **if**(flag) {
50. printf("No\n");
51. **continue**;
52. }
54. **int** ans = 0;
56. **for**(**int** i = 0; i < maxn; ++i)
57. **if**( vst[i] && par[i] == i)
58. ans ++;
60. **if**(ans == 1)
61. printf("Yes\n");
62. **else**
63. printf("No\n");
64. }
65. **return** 0;
66. }

专题五 并查集 14 POJ 1308 Is It A Tree

1. **int** main(**void**){
2. **int** a,b;
3. **int** c=1;
4. **while**(~scanf("%d%d",&a,&b) && (a!=-1 && b!=-1)){
5. **if**(a==0 && b==0){
6. printf("Case %d is a tree.\n",c++);
7. **continue**;
8. }
9. set<**int**> p;
10. p.insert(a);
11. p.insert(b);
12. **int** m=1;
13. **while**(scanf("%d%d",&a,&b) && (a || b)){
14. p.insert(a);
15. p.insert(b);
16. m++;
17. }
18. **if**(p.size()-m!=1){
19. printf("Case %d is not a tree.\n",c++);
20. }
21. **else**{
22. printf("Case %d is a tree.\n",c++);
23. }
24. }
25. **return** 0;
26. }