

MST

● Graded

Student

ANSHUMAN SINGH

Total Points

60 / 60 pts

Autograder Score

60.0 / 60.0

Passed Tests

Test 1 (10/10)

Test 2 (10/10)

Test 3 (5/5)

Test 5 (5/5)

Test 7 (5/5)

Test 9 (5/5)

Autograder Results

Test 1 (10/10)

Test 2 (10/10)

Test 3 (5/5)

Test 5 (5/5)

Test 7 (5/5)

Test 9 (5/5)

Submitted Files

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  // Disjoint set union
5
6  typedef struct{
7      int *parent;
8      int *rank;
9      int n;
10 } DJS;
11
12 DJS *dsu;
13
14 void create(int n){
15     dsu =(DJS*)malloc(sizeof(DJS));
16     dsu->parent =(int*)malloc((n+1)*sizeof(int));
17     dsu->rank =(int*)malloc((n+1)*sizeof(int));
18     dsu->n =n;
19     for(int i=1;i<=n;i++){
20         dsu->parent[i]=i;
21         dsu->rank[i]=0;
22     }
23 }
24
25 int find(int x){
26     while(x!=dsu->parent[x])
27         x=dsu->parent[x];
28     return x;
29 }
30
31 void merge(int i,int j){
32     if(find(i)==find(j))
33         return;
34     int hi=find(i);
35     int hj=find(j);
36     if(dsu->rank[hi]>dsu->rank[hj])
37         dsu->parent[hj]=hi;
38     else{
39         dsu->parent[hi]=hj;
40         if(dsu->rank[hi]==dsu->rank[hj])
41             dsu->rank[hj]++;
42     }
43 }
44
45 int report(int i,int j){
46     if(find(i)==find(j))
47         return 1;
48     else
49         return 0;
```

```

50 }
51
52 void freeSet(DJS *dsu){
53     free(dsu->parent);
54     free(dsu->rank);
55     free(dsu);
56 }
57
58 // MST
59
60 typedef struct{
61     int u, v, wt;
62 }Edge;
63
64 void swap(Edge *a, Edge *b){
65     Edge temp = *a;
66     *a = *b;
67     *b = temp;
68 }
69
70 void heapify(Edge *a, int n, int i){
71     int k=i;
72     if(2*i+1<n&&a[2*i+1].wt>a[k].wt)
73         k=2*i+1;
74     if(2*i+2<n&&a[2*i+2].wt>a[k].wt)
75         k=2*i+2;
76     if(k!=i){
77         swap(&a[i], &a[k]);
78         heapify(a, n, k);
79     }
80 }
81
82 void heapSort(Edge *a, int n){
83     for(int i =n/2-1; i>=0; i--){
84         heapify(a, n, i);
85     }
86     for (int i =n-1; i>=0; i--){
87         swap(&a[0], &a[i]);
88         heapify(a, i, 0);
89     }
90
91 int MST(Edge *edges, int tot_edges, int n) {
92     create(n);
93     int mst_wt=0;
94     int ctr=0;
95
96     for(int j=0; j<tot_edges&&ctr<n-1; j++){
97         Edge edge=edges[j];
98         if(find(edge.u)!=find(edge.v)){
99             merge(edge.u, edge.v);
100             mst_wt+=edge.wt;
101             ctr++;

```

```

102     }
103 }
104 freeSet(dsu);
105
106 return mst_wt;
107 }
108
109
110
111 int main(){
112     int flag;
113     scanf("%d",&flag);
114     if(flag==0){
115         int n,q;
116         scanf("%d %d",&n,&q);
117         create(n);
118         while(q-->0){
119             int type,i,j;
120             scanf("%d %d %d",&type,&i,&j);
121             if(type==0){
122                 merge(i,j);
123             }
124             else{
125                 int ans=report(i,j);
126                 printf("%d ",ans);
127             }
128         }
129         freeSet(dsu);
130     }
131     else{
132         int n;
133         scanf("%d",&n);
134         Edge *edges=(Edge *)malloc(n*n*sizeof(Edge));
135         int tot_edges=0;
136         for(int i=1;i<=n;i++){
137             int j,wt;
138             while(1){
139                 scanf("%d",&j);
140                 if(j== -1)
141                     break;
142                 scanf("%d",&wt);
143                 int k=tot_edges;
144                 edges[k].u=i;
145                 edges[k].v=j;
146                 edges[k].wt=wt;
147                 tot_edges++;
148             }
149         }
150         heapSort(edges,tot_edges);
151         int mst_wt = MST(edges, tot_edges, n);
152         printf("%d ", mst_wt);
153     }

```

```
154 | return 0;  
155 | }
```
