SSSP • Graded

Student

SRUTHI SUBRAMANIAN

**Total Points** 

50 / 50 pts

**Autograder Score** 

50.0 / 50.0

**Passed Tests** 

Test 1 (10/10)

Test 2 (20/20)

Test 3 (20/20)

## **Autograder Results**

Test 1 (10/10)

Test 2 (20/20)

Test 3 (20/20)

**Submitted Files** 

**≛** Download

```
//Final Code for SSSP
1
2
     #include <stdio.h>
3
4
     int main() {
5
       int n;
6
       scanf("%d",&n);
7
       int adjmat[n+1][n+1];
8
       int a=0;
9
       int b;
10
       int inf=5;
11
       int size=n;
12
       for (int i=0;i<n;i++){
          for (int j=0; j< n; j++){
13
14
            adjmat[i][j]=0;
15
          }
16
       }
17
       for (int i=0;i<n;i++){
18
          scanf("%d", &a);
19
          while (a!=-1){
20
            scanf("%d",&b);
21
            adjmat[i][a]=b;
22
            inf=inf+b:
23
            scanf("%d", &a);
24
         }
25
       }
26
       int source;
27
       scanf("%d", &source);
28
       int Heap[n+1];
29
       int d[n+1];
30
       for (int i=0;i<n;i++){
31
          d[i]=inf;
32
          Heap[i]=i;
33
34
35
       int* MinHeapify(int* heap, int ind) {
36
            //1 indexing!!
37
            int minind = ind;
38
            if (ind * 2 <= size && d[heap[2 * ind - 1]] < d[heap[ind - 1]]) {
39
               minind = 2 * ind;
40
            }
41
            if (ind * 2 + 1 <= size && d[heap[2 * ind]] < d[heap[minind - 1]]) {
42
               minind = 2 * ind + 1;
43
            }
44
            if (minind != ind) {
45
               //swap minind term and X and then call MinHeapify on minind
46
               int X = ind;
47
               int t = heap[X - 1];
               heap[X - 1] = heap[minind - 1];
48
49
               heap[minind - 1] = t;
```

```
50
               return MinHeapify(heap, minind);
51
            }
52
            else {
53
               return heap;
54
            }
55
          }
        int ExtractMin(int* heap){
56
57
            int temp;
58
            temp = heap[0];
59
            heap[0]=heap[size-1];
60
            size=size-1;
61
            MinHeapify(heap, 1);
62
            return temp;
63
          }
64
       int find(int* heap, int ind){
          for (int k=0; k < size; k++){
65
66
            if (heap[k]==ind){
67
               return k+1;
68
            }
69
          }
70
71
       int DecreaseKey(int* heap, int ind, int key){
72
          d[ind-1]=key;
73
          int X=find(heap,ind-1);
74
          int parent;
75
          int temp;
76
          while(1){
77
            if(X==1){
78
               return 0;
79
            }
80
            parent =X/2;
81
            if (d[heap[X-1]]>d[heap[parent-1]]){
82
               return 0;
83
            }
84
            else{
85
               temp=heap[parent-1];
86
               heap[parent-1]=heap[X-1];
87
               heap[X-1]=temp;
               X=parent;
88
89
            }
90
          }
91
92
       int final[n+1];
93
        DecreaseKey(Heap,source+1,0);
94
       int u = ExtractMin(Heap);
95
       int m=0;
96
       final[m]=u;
97
        m=m+1;
98
       for (int j=0; j< n; j++){
99
          if (adjmat[u][j]!=0){
100
            if (d[j]>d[u]+adjmat[u][j]){
101
               DecreaseKey(Heap,j+1,d[u]+adjmat[u][j]);
```

```
102
            }
          }
103
104
        }
105
        while (size>0){
          u=ExtractMin(Heap);
106
107
          final[m]=u;
108
          m=m+1;
          for (int j=0; j<n; j++){
109
110
            if (adjmat[u][j]!=0){
111
              if (d[j]>d[u]+adjmat[u][j]){
                 Decrease Key (Heap, j+1, d[u]+adjmat[u][j]);\\
112
113
              }
            }
114
          }
115
116
        }
117
        for(int i=0;i<n;i++){
          printf("%d %d ",final[i],d[final[i]]);
118
119
        }
       return 0;
120
121 }
```