Data Structures and Algorithms for Engineers

Module 7: Graphs

Lecture 4: Minimum Spanning Tree, Prim's algorithm, Kruskal's algorithm

David Vernon
Carnegie Mellon University Africa

vernon@cmu.edu
www.vernon.eu

Minimum Spanning Tree

- A spanning tree of a graph $G = (V, E)$ is a subset of edges from E forming a tree connecting all vertices of V

- For edge-weighted graphs we are interested in the minimum spanning tree*:*

  – The tree whose **sum of edge weights** is as small as possible

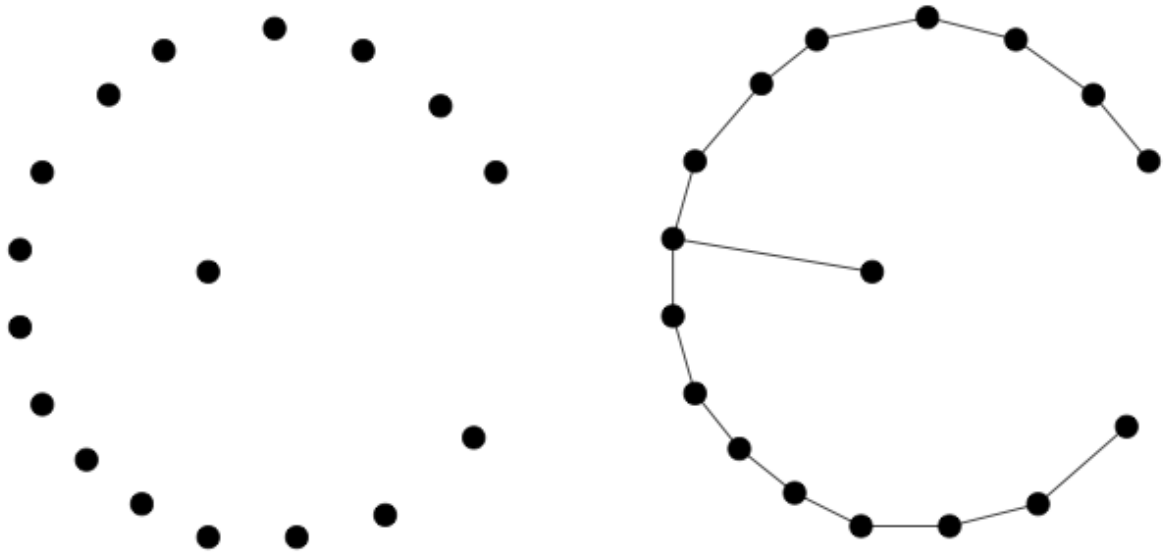
Minimum Spanning Tree

Example application:

- find the smallest amount of a connector (road, wire, pipe)

  required to connect a set of points (cities, houses, components)

Minimum Spanning Tree

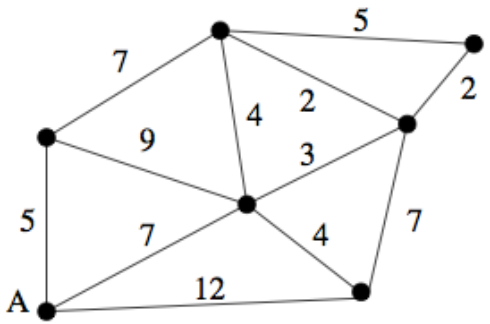


Assuming that the edge weights represent distance between a vertex ...

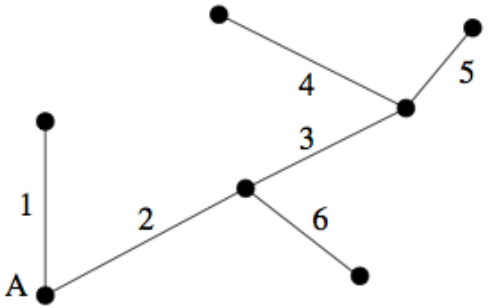
Minimum Spanning Tree

- A minimum spanning tree minimizes the total length (weight) over all possible spanning trees
- There can be more than one minimum spanning tree in a graph
- Two main algorithms
  - Prim's Algorithm
  - Kruskal's Algorithm

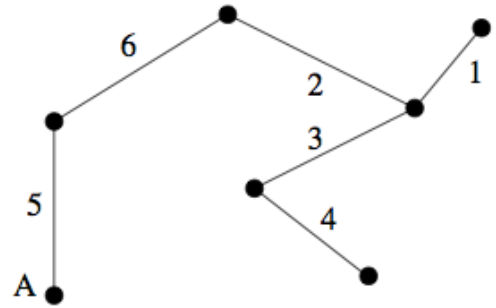
Minimum Spanning Tree



G



Prim(G,A)



Kruskal(G)

Numbers denote order of insertion

Minimum Spanning Tree

Prim's Algorithm

Greedy algorithmic strategy:

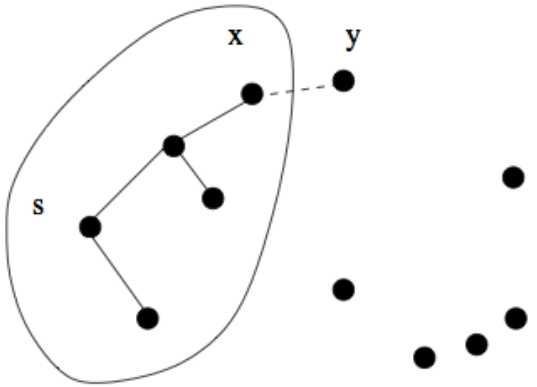
- Make a decision about what to do next by selecting the best local option from all available choices without regard to global structure
- Do it
- Repeat until finished

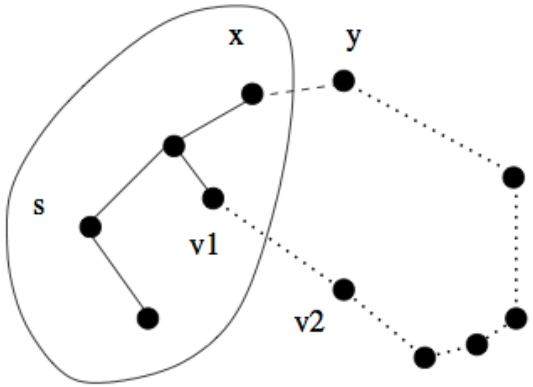
Minimum Spanning Tree

Prim's Algorithm

Repeatedly select the smallest weight edge that will enlarge the number of vertices in the tree

- Start from one (any) vertex,
- Grow the rest of the tree, one edge at a time,
- Until all vertices are included





Minimum Spanning Tree

Prim's Algorithm

Prim-MST(G):

Select an arbitrary vertex s to start the tree from

While (there are still non-tree vertices remaining)

Select the edge of minimum weight between a tree vertex and a non-tree vertex

Add the selected edge and vertex to the tree T_prim

Any tree vertex

Any non-tree vertex

Minimum Spanning Tree

```
/* Prim's algorithm                                                    */

prim(graph *g, int start) {
   int i;                 /* counter                                 */
   edgenode *p;           /* temporary pointer                       */
   bool intree[MAXV+1];   /* is the vertex in the tree yet?          */
   int distance[MAXV+1]; /* cost of adding to tree                   */
   int parent[MAXV+1];    /* parent vertex                           */
   int v;                 /* current vertex to process               */
   int w;                 /* candidate next vertex                   */
   int weight;            /* edge weight                             */
   int dist;              /* best current distance from start */

   for (i=1; i<=g->nvertices; i++) {
      intree[i] = FALSE;
      distance[i] = MAXINT;
      parent[i] = -1;
   }

   distance[start] = 0;
   v = start;
```

Minimum Spanning Tree

```
    while (intree[v] == FALSE) { // keep going until all vertices
       intree[v] = TRUE;         // are in the tree
       p = g->edges[v];
       while (p != NULL) {       // compute the distances and parents
          w = p->y;
          weight = p->weight;
          if ((weight < distance[w]) && (intree[w] == FALSE)) {
             distance[w] = weight;
             parent[w] = v;
          }
          p = p->next;
       }
       v = 1;
       dist = MAXINT;                    // find the vertex with
       for (i=1; i<=g->nvertices; i++)   // the minimum distance
          if ((intree[i] == FALSE) && (distance[i] < dist)) {
             dist = distance[i];
             v = i;
          }
    }
}
```

Minimum Spanning Tree

```
while (intree[v] == FALSE) { // keep going until all vertices
   intree[v] = TRUE;         // are in the tree
   p = g->edges[v];
   while (p != NULL) {       // compute the distances and parents
      w = p->v;
      weight = p->weight;
      if ((weight < distance[w]) && (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }
   v = 1;
   dist = MAXINT;                    // find the vertex with
   for (i=1; i<=g->nvertices; i++)   // the minimum distance
      if ((intree[i] == FALSE) && (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
   }
}
```

Extract weights of all edges from the new tree-vertex v

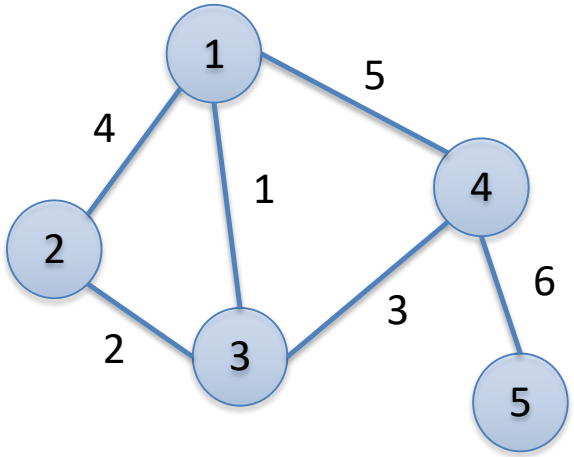
Update the distance array if the edge is to a non-tree vertex and the weight is less than the current weight

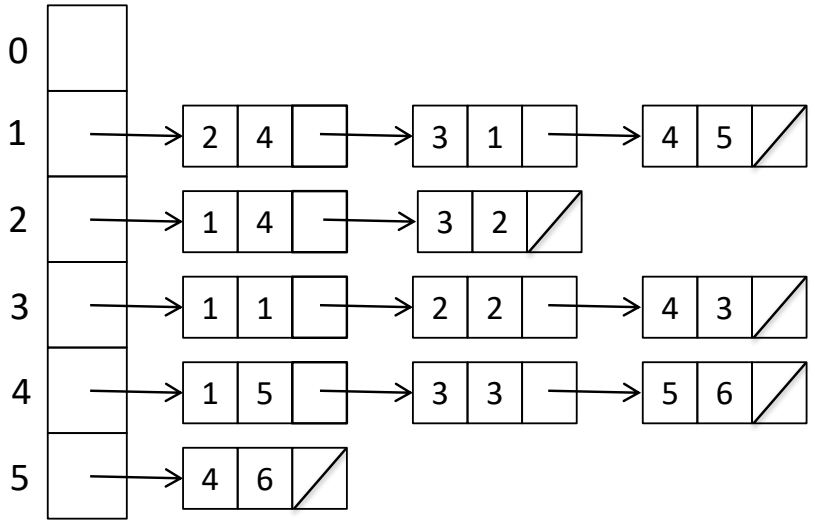
Minimum Spanning Tree

```
        while (intree[v] == FALSE) { // keep going until all vertices
           intree[v] = TRUE;         // are in the tree
           p = g->edges[v];
           while (p != NULL) {       // compute the distances and parents
              w = p->y;
              weight = p->weight;
              if ((weight < distance[w]) && (intree[w] == FALSE)) {
                 distance[w] = weight;
                 parent[w] = v;
              }
              p = p->next;
           }
           v = 1;
           dist = MAXINT;                    // find the vertex with
           for (i=1; i<=g->nvertices; i++)   // the minimum distance
              if ((intree[i] == FALSE) && (distance[i] < dist)) {
                 dist = distance[i];
                 v = i;
              }
        }
     }
```

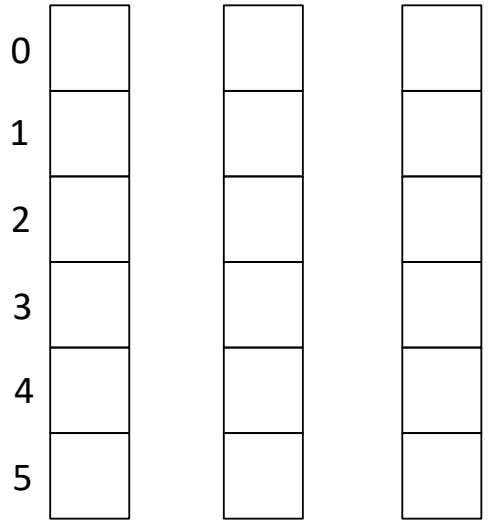
Find the edge between a tree vertex and a non-tree vertex with the minimum weight

This gives the vertex to be added to the MST

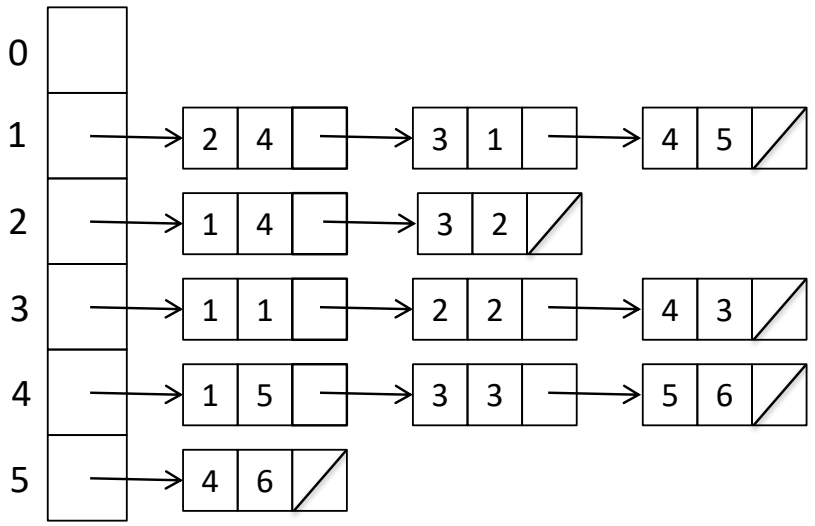


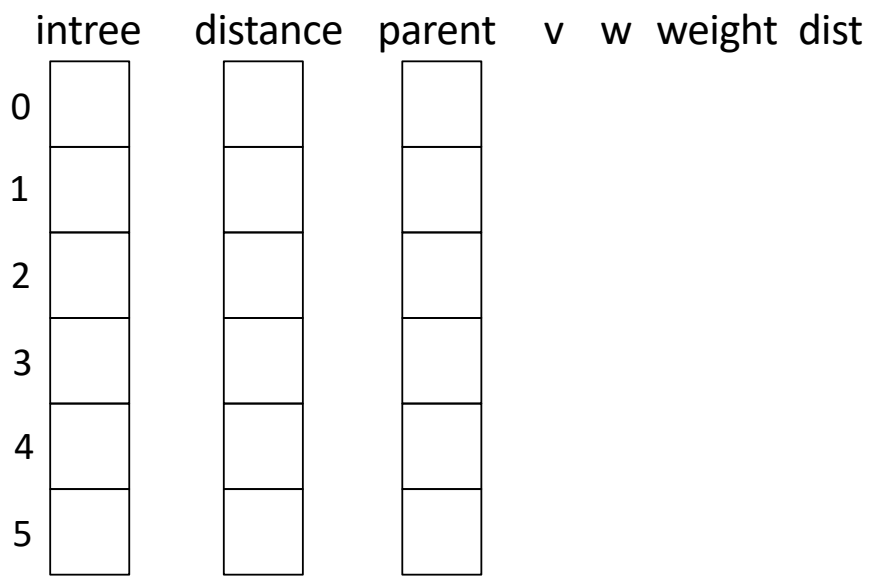


intree distance parent v w weight dist

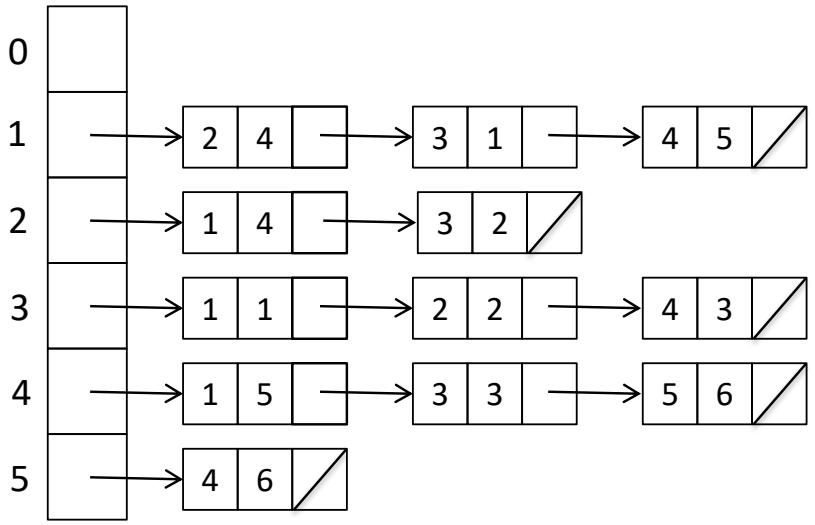


```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```





```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```

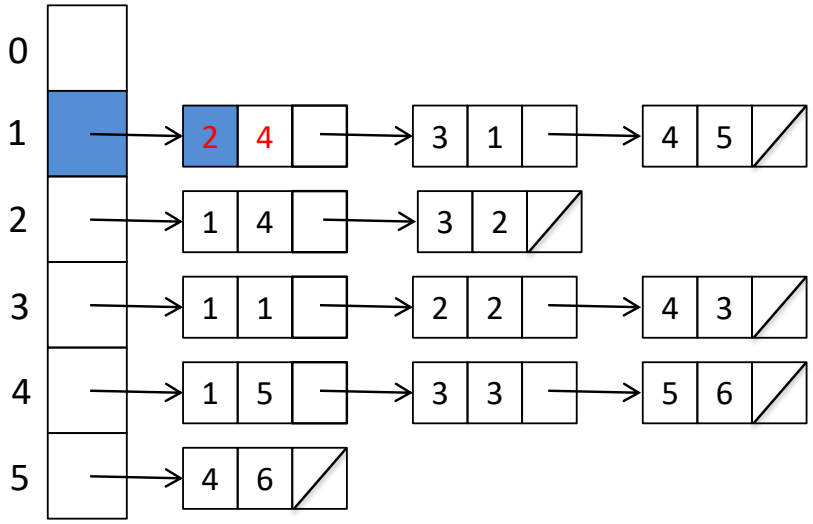


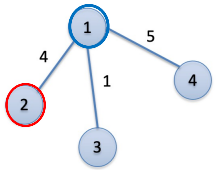
	intree	distance	parent
0			
1	F	∞	-1
2	F	∞	-1
3	F	∞	-1
4	F	∞	-1
5	F	∞	-1

v w weight dist

```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```

start = 1

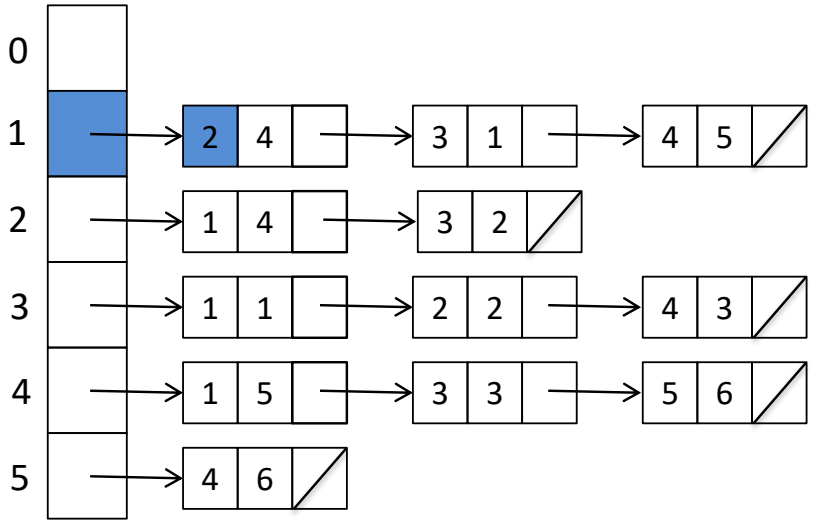


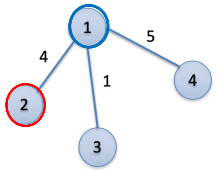


	intree	distance	parent
0			
1	T	0	-1
2	F	∞	-1
3	F	∞	-1
4	F	∞	-1
5	F	∞	-1

v	w	weight	dist
1	2	4	

```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```

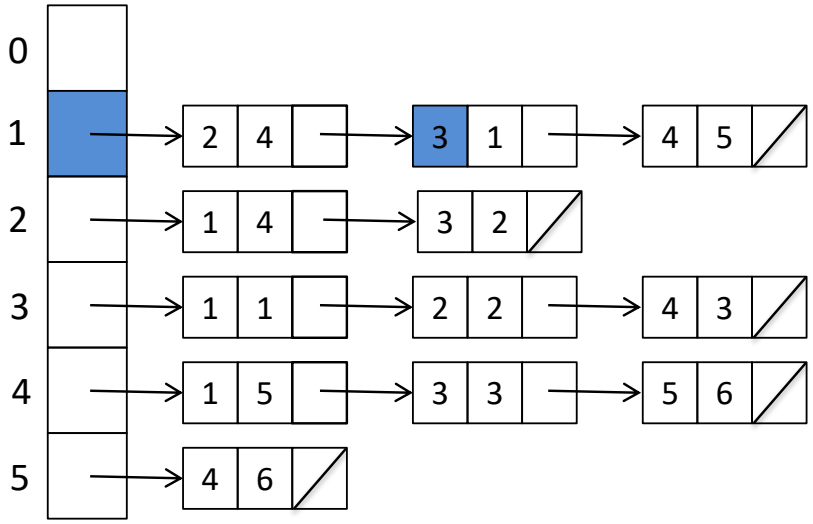


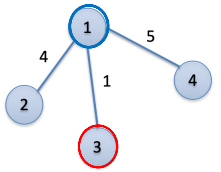


	intree	distance	parent
0			
1	T	0	-1
2	F	4	1
3	F	∞	-1
4	F	∞	-1
5	F	∞	-1

v	w	weight	dist
1	2	4	

```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```

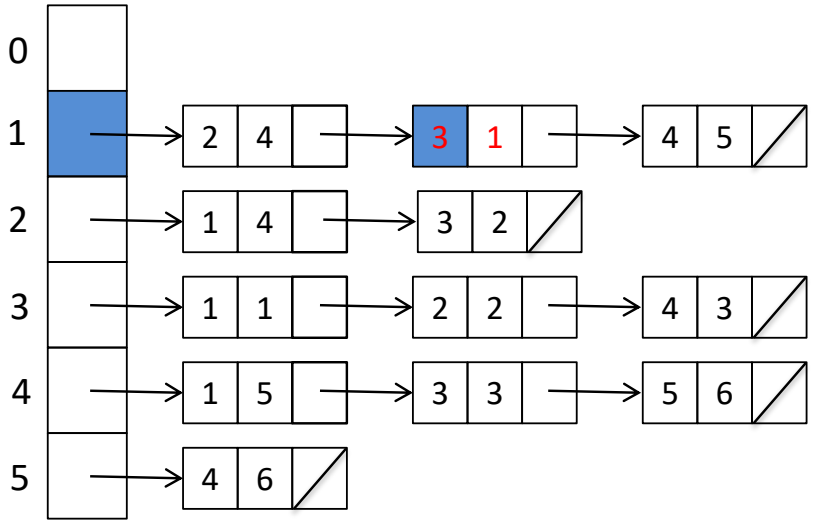


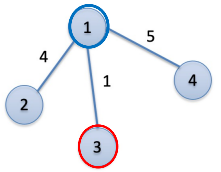


	intree	distance	parent
0			
1	T	0	-1
2	F	4	1
3	F	∞	-1
4	F	∞	-1
5	F	∞	-1

v	w	weight	dist
1	2	4	

```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```

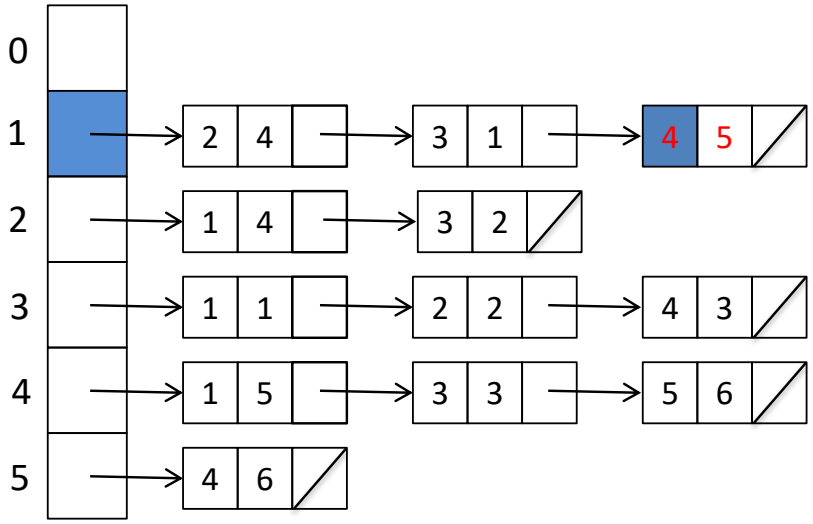


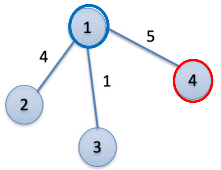


	intree	distance	parent
0			
1	T	0	-1
2	F	4	1
3	F	1	1
4	F	∞	-1
5	F	∞	-1

v	w	weight	dist
1	~~2~~	~~4~~	
	3	1	

```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```

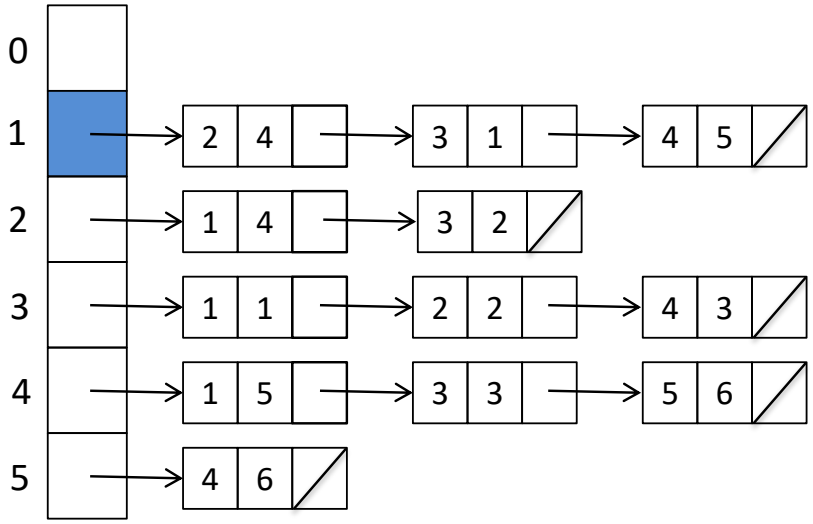


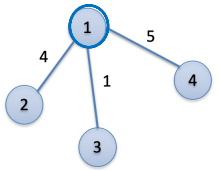


	intree	distance	parent
0			
1	T	0	-1
2	F	4	1
3	F	1	1
4	F	5	1
5	F	∞	-1

v	w	weight	dist
1	~~2~~	~~4~~	
	~~3~~	~~1~~	
	4	5	

```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```

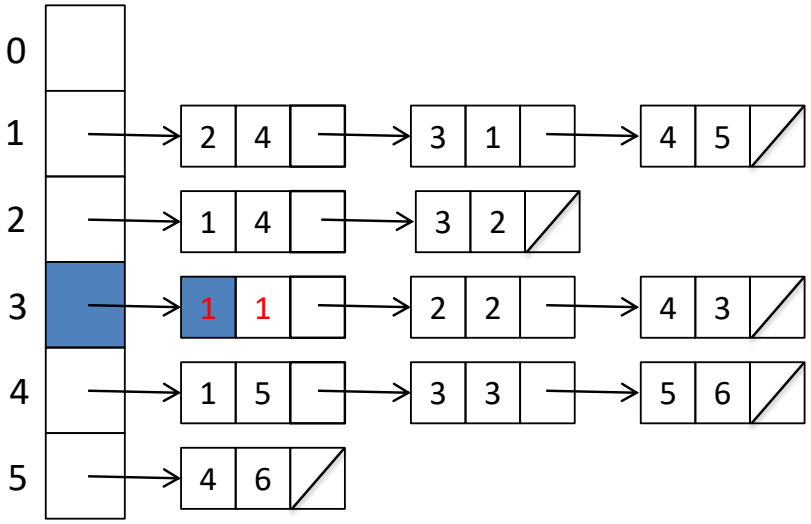


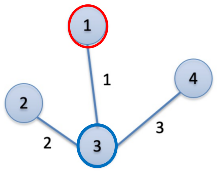


	intree	distance	parent
0			
1	T	0	-1
2	F	4	1
3	F	1	1
4	F	5	1
5	F	∞	-1

v	w	weight	dist
~~1~~	~~2~~	~~4~~	∞
1	~~3~~	~~1~~	4
2	4	5	1
3			

```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```

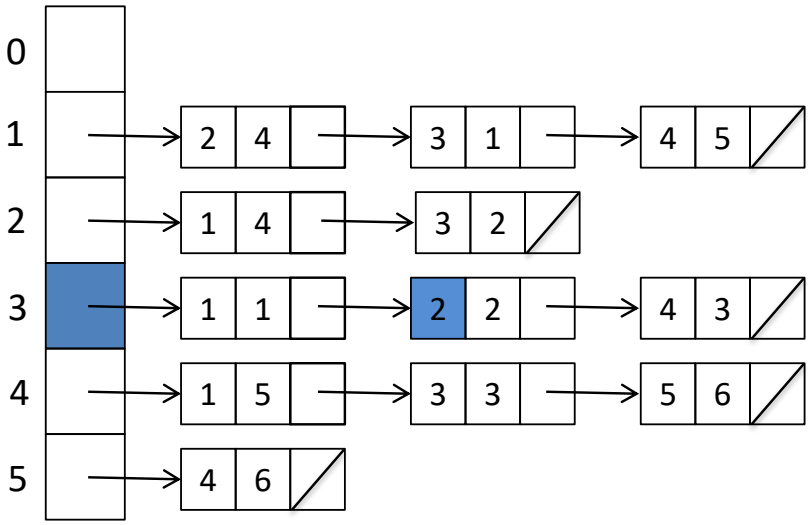


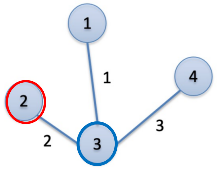


	intree	distance	parent
0			
1	T	0	-1
2	F	4	1
3	T	1	1
4	F	5	1
5	F	∞	-1

v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	

```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```

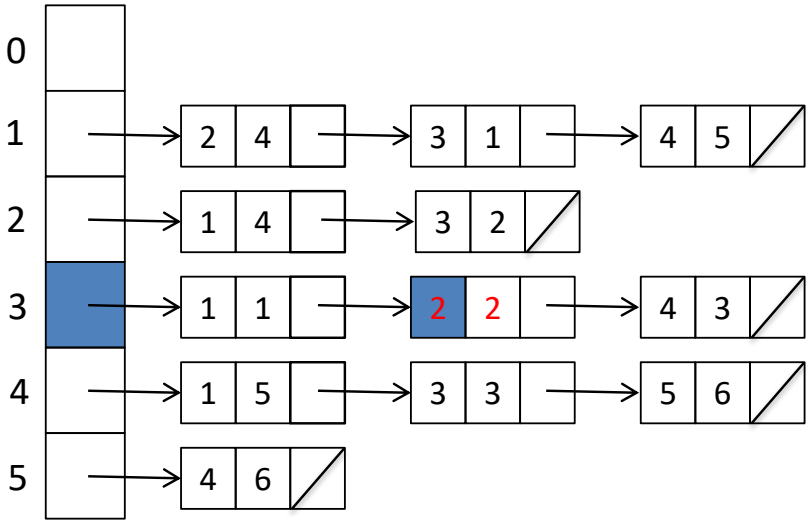


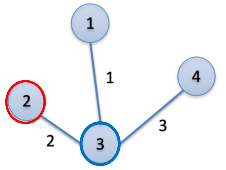


	intree	distance	parent
0			
1	T	0	-1
2	F	4	1
3	T	1	1
4	F	5	1
5	F	∞	-1

v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	

```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```

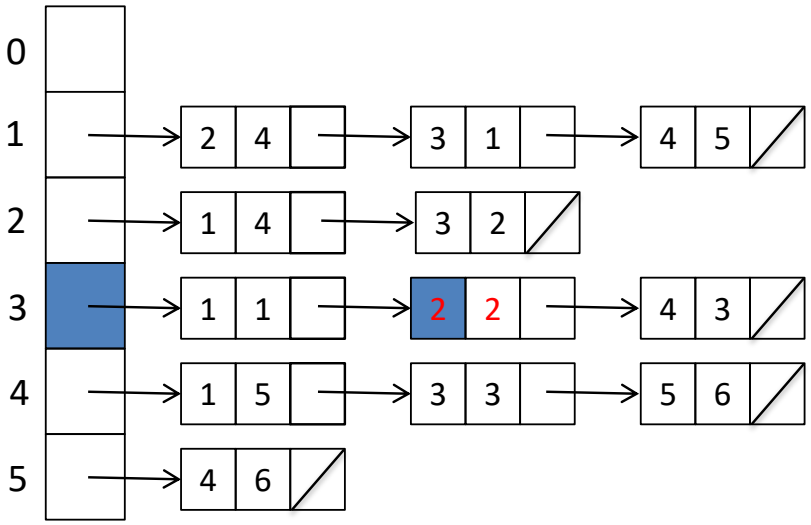


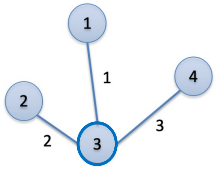


	intree	distance	parent
0			
1	T	0	-1
2	F	4	1
3	T	1	1
4	F	5	1
5	F	∞	-1

v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	
	2	2	

```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```

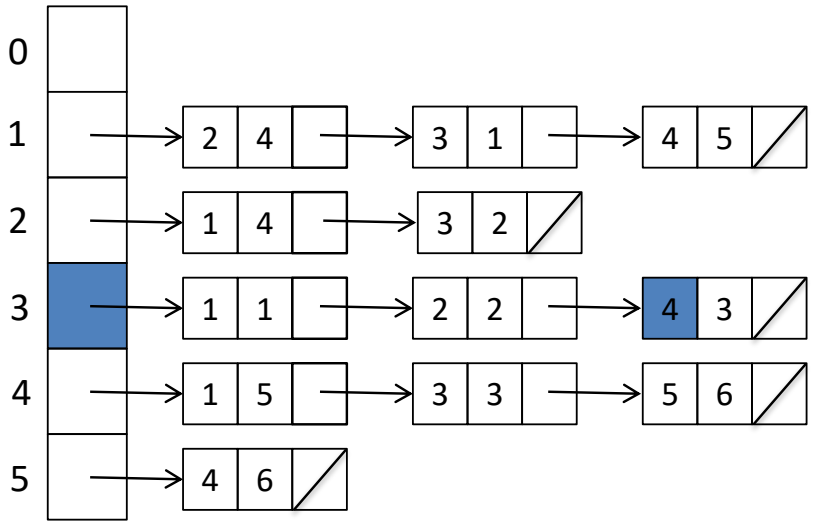


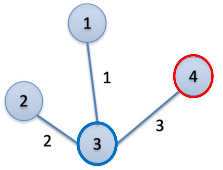


	intree	distance	parent
0			
1	T	0	-1
2	F	2	3
3	T	1	1
4	F	5	1
5	F	∞	-1

v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	
	2	2	

```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```

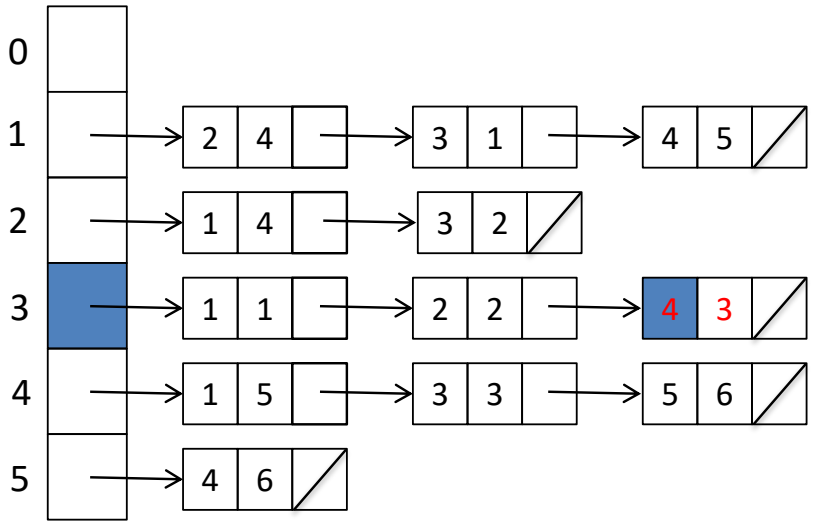


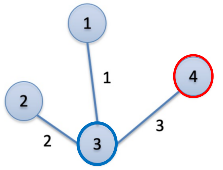


	intree	distance	parent
0			
1	T	0	-1
2	F	2	3
3	T	1	1
4	F	5	1
5	F	∞	-1

v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	
	2	2	

```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```

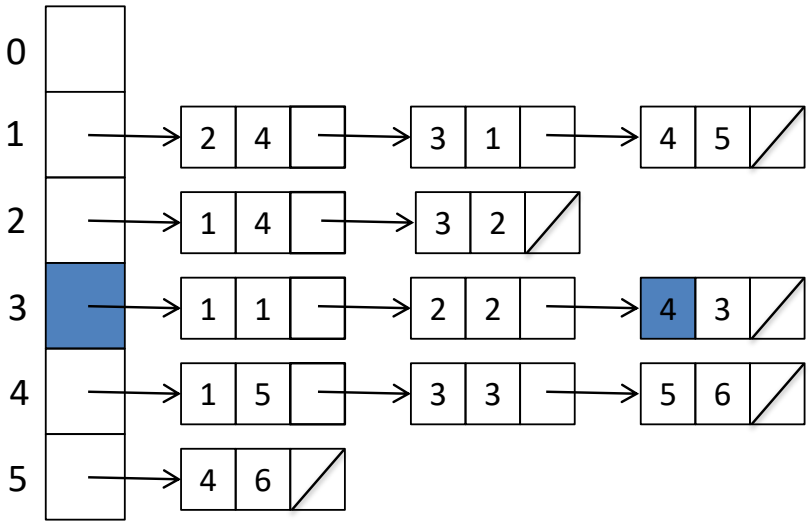


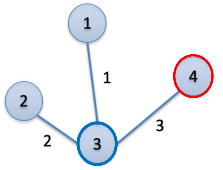


	intree	distance	parent
0			
1	T	0	-1
2	F	2	3
3	T	1	1
4	F	5	1
5	F	∞	-1

v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	
	2	2	
	4	3	

```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```

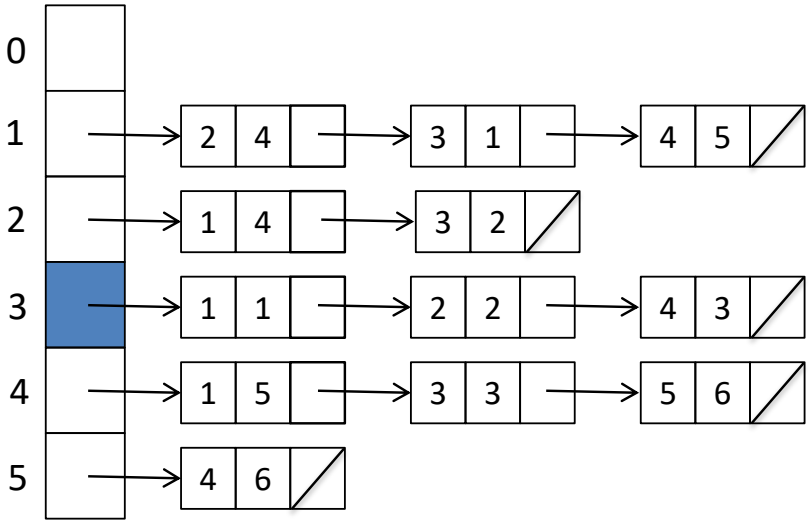


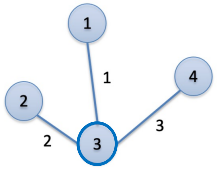


	intree	distance	parent
0			
1	T	0	-1
2	F	2	3
3	T	1	1
4	F	3	3
5	F	∞	-1

v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	
	2	2	
	4	3	

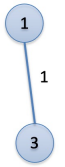
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



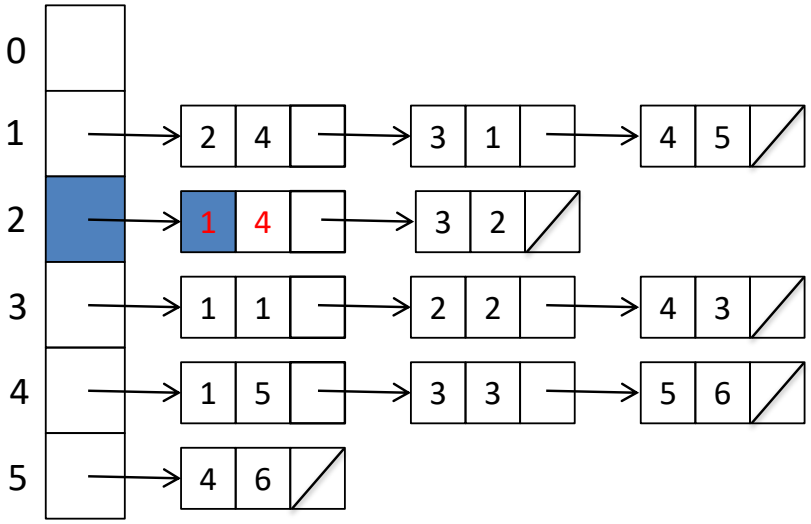


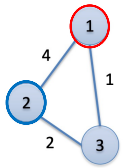
	intree	distance	parent
0			
1	T	0	-1
2	F	2	3
3	T	1	1
4	F	3	3
5	F	∞	-1

v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	∞
1	2	2	2
2	4	3	



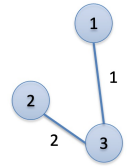
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



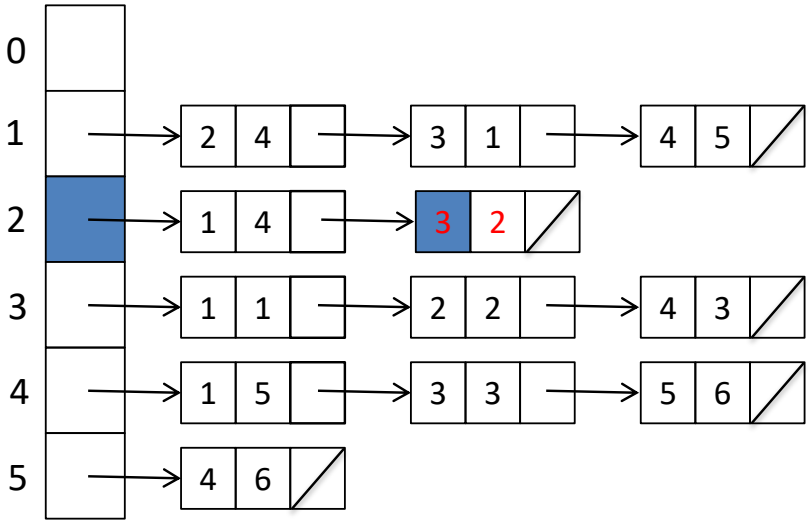


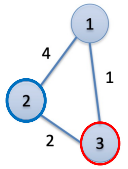
	intree	distance	parent
0			
1	T	0	-1
2	T	2	3
3	T	1	1
4	F	3	3
5	F	∞	-1

v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	∞
1	2	2	2
2	4	3	
	1	4	



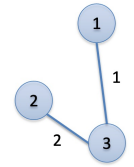
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



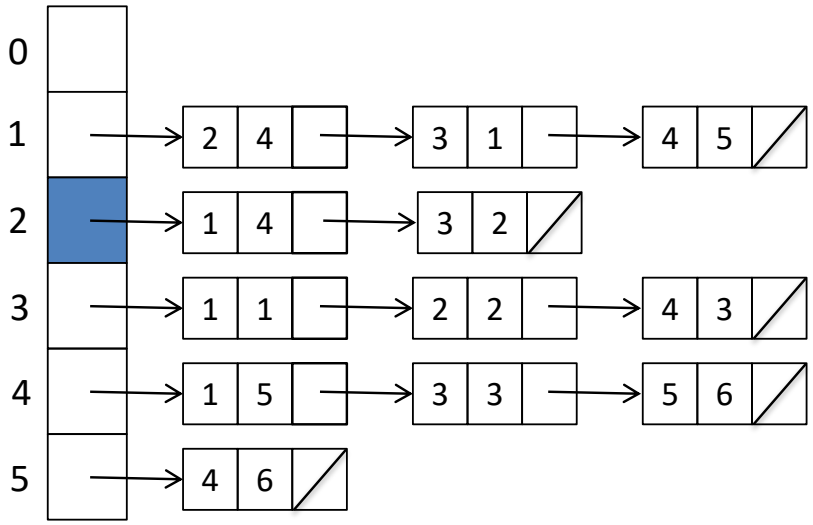


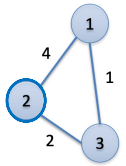
	intree	distance	parent
0			
1	T	0	-1
2	T	2	3
3	T	1	1
4	F	3	3
5	F	∞	-1

v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	∞
1	2	2	2
2	4	3	
	1	4	
	3	2	



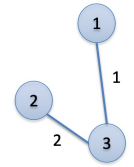
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



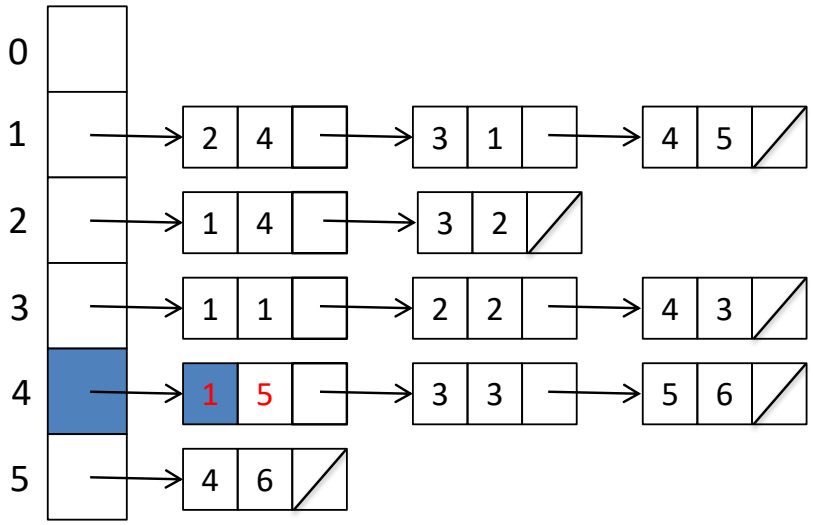


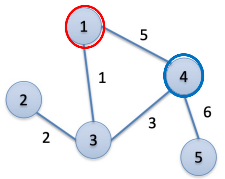
	intree	distance	parent
0			
1	T	0	-1
2	T	2	3
3	T	1	1
4	F	3	3
5	F	∞	-1

v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	∞
1	2	2	2
2	4	3	∞
1	1	4	3
4	3	2	



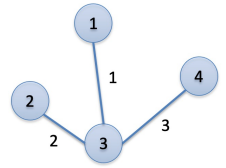
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



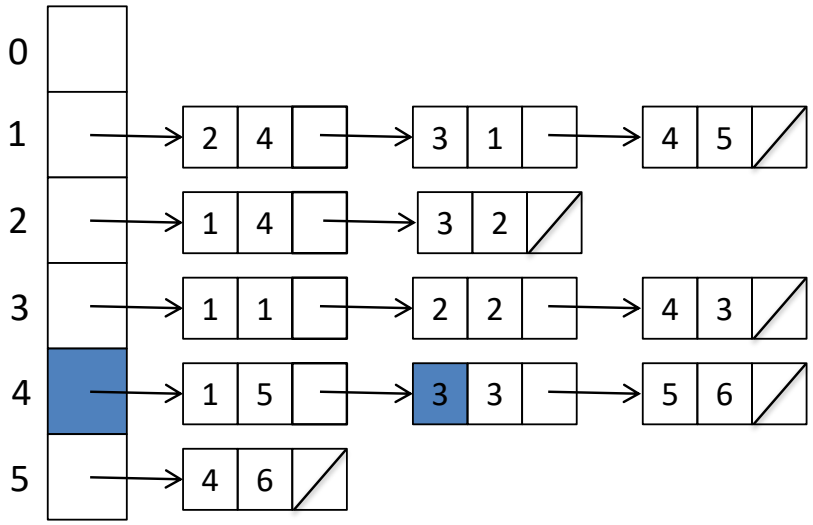


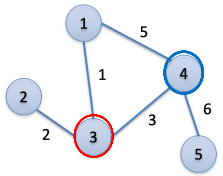
	intree	distance	parent
0			
1	T	0	-1
2	T	2	3
3	T	1	1
4	T	3	3
5	F	∞	-1

v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	∞
1	2	2	2
2	4	3	∞
1	1	4	3
4	3	2	
	1	5	



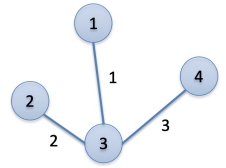
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



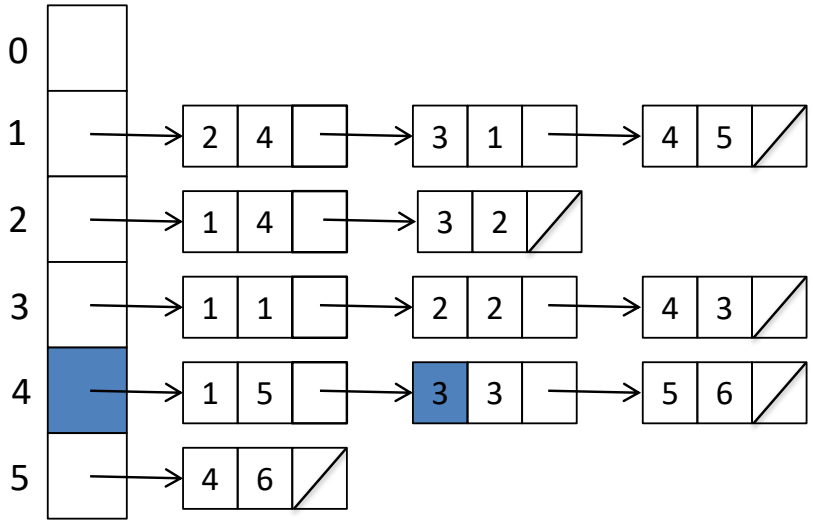


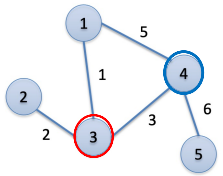
	intree	distance	parent
0			
1	T	0	-1
2	T	2	3
3	T	1	1
4	T	3	3
5	F	∞	-1

v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	∞
1	2	2	2
2	4	3	∞
1	1	4	3
4	3	2	
	1	5	



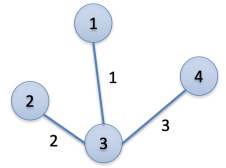
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



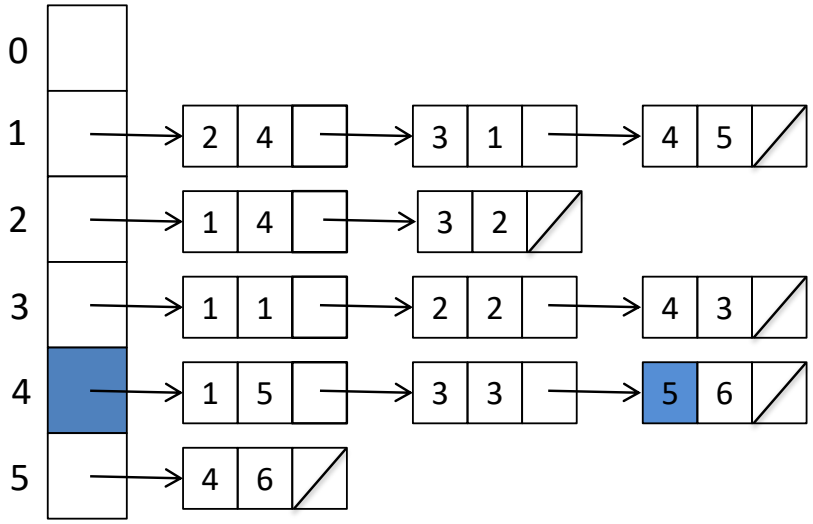


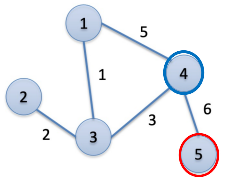
	intree	distance	parent
0			
1	T	0	-1
2	T	2	3
3	T	1	1
4	T	3	3
5	F	∞	-1

v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	∞
1	2	2	2
2	4	3	∞
1	1	4	3
4	3	2	
	1	5	
	3	3	



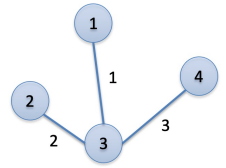
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



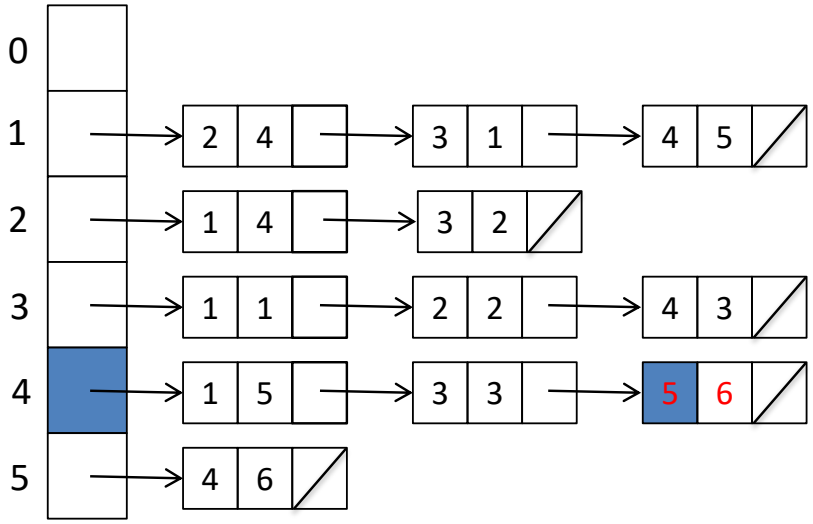


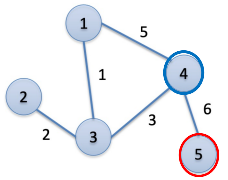
	intree	distance	parent
0			
1	T	0	-1
2	T	2	3
3	T	1	1
4	T	3	3
5	F	∞	-1

v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	∞
1	2	2	2
2	4	3	∞
1	1	4	3
4	3	2	
	1	5	
	3	3	



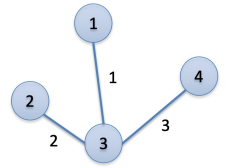
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



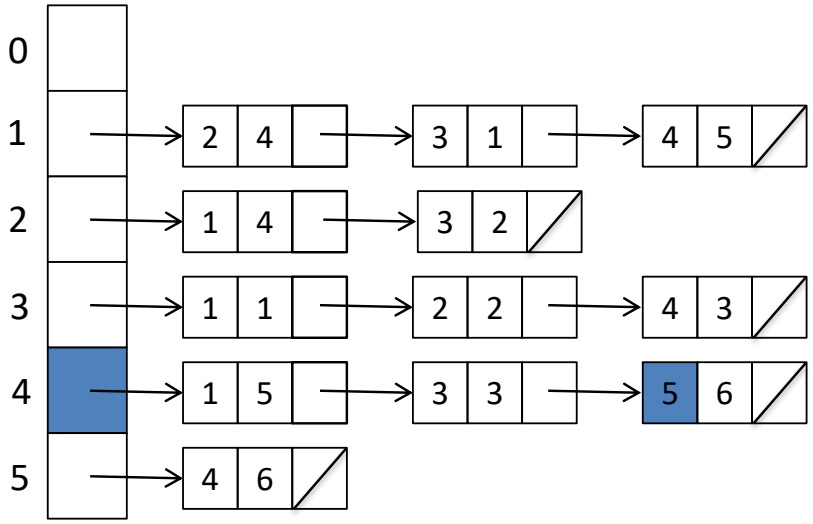


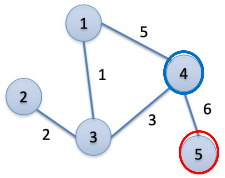
	intree	distance	parent
0			
1	T	0	-1
2	T	2	3
3	T	1	1
4	T	3	3
5	F	∞	-1

v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	∞
1	2	2	2
2	4	3	∞
1	1	4	3
4	3	2	
	1	5	
	3	3	
	5	6	



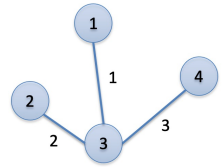
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



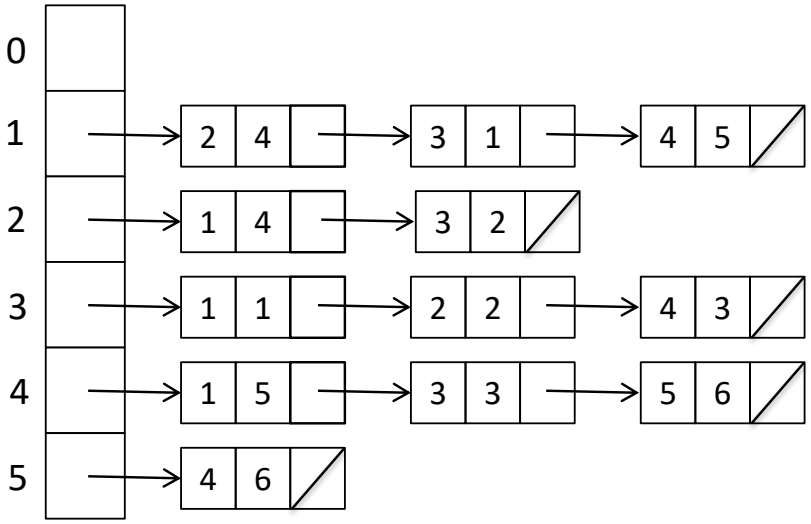


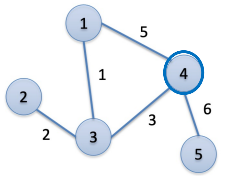
	intree	distance	parent
0			
1	T	0	-1
2	T	2	3
3	T	1	1
4	T	3	3
5	F	6	4

v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	∞
1	2	2	2
2	4	3	∞
1	1	4	3
4	3	2	
	1	5	
	3	3	
	5	6	



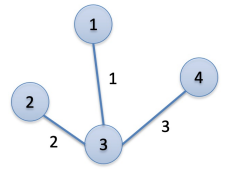
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



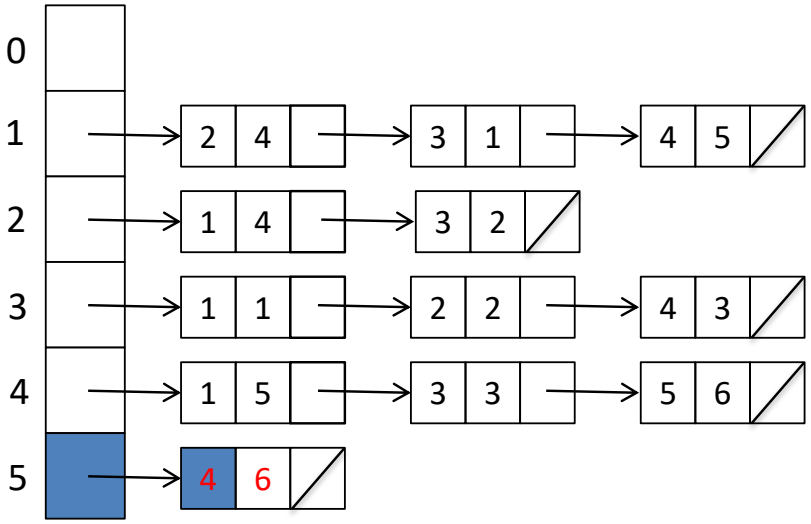


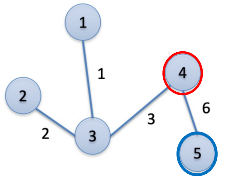
	intree	distance	parent
0			
1	T	0	-1
2	T	2	3
3	T	1	1
4	T	3	3
5	F	6	4

v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	∞
1	2	2	2
2	4	3	∞
1	1	4	3
4	3	2	∞
1	1	5	6
5	3	3	
	5	6	



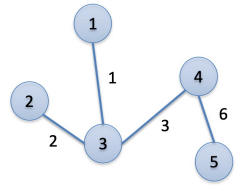
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



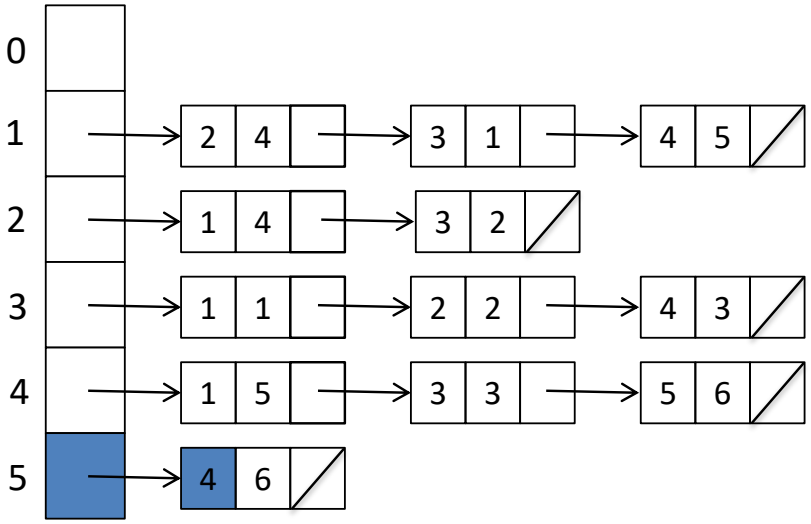


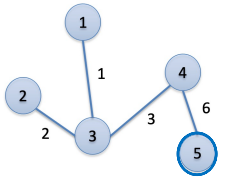
	intree	distance	parent
0			
1	T	0	-1
2	T	2	3
3	T	1	1
4	T	3	3
5	T	6	4

v	w	weight	dist
1	...	...	∞
1	4	6	4
2			1
3			∞
1			2
2			∞
1			3
4			∞
1			6
5			



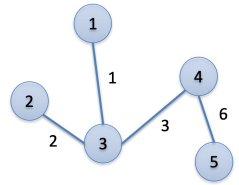
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



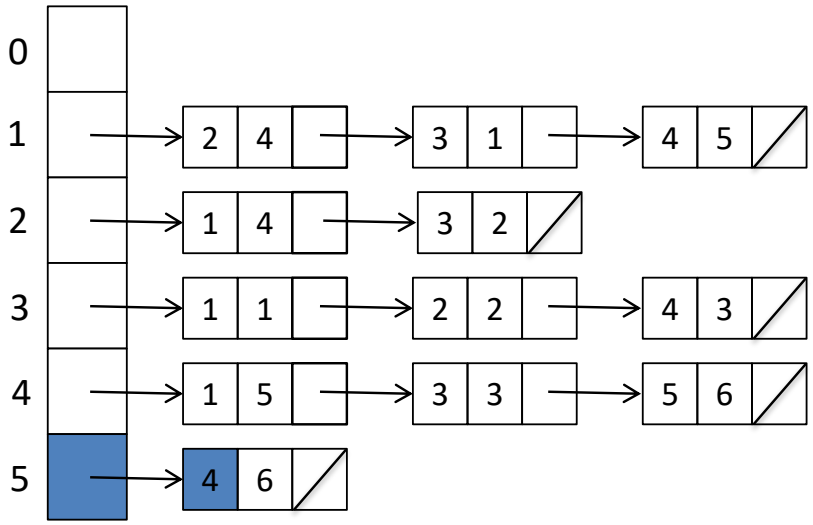


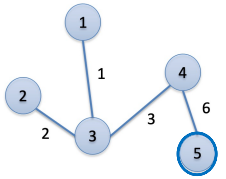
	intree	distance	parent
0			
1	T	0	-1
2	T	2	3
3	T	1	1
4	T	3	3
5	T	6	4

v	w	weight	dist
1	...	...	∞
1	4	6	4
2			1
3			∞
1			2
2			∞
1			3
4			∞
1			6
5			∞
1			



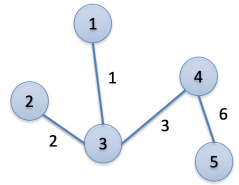
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```

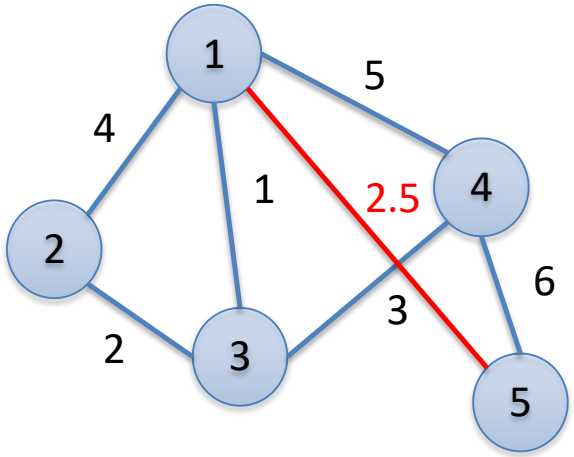


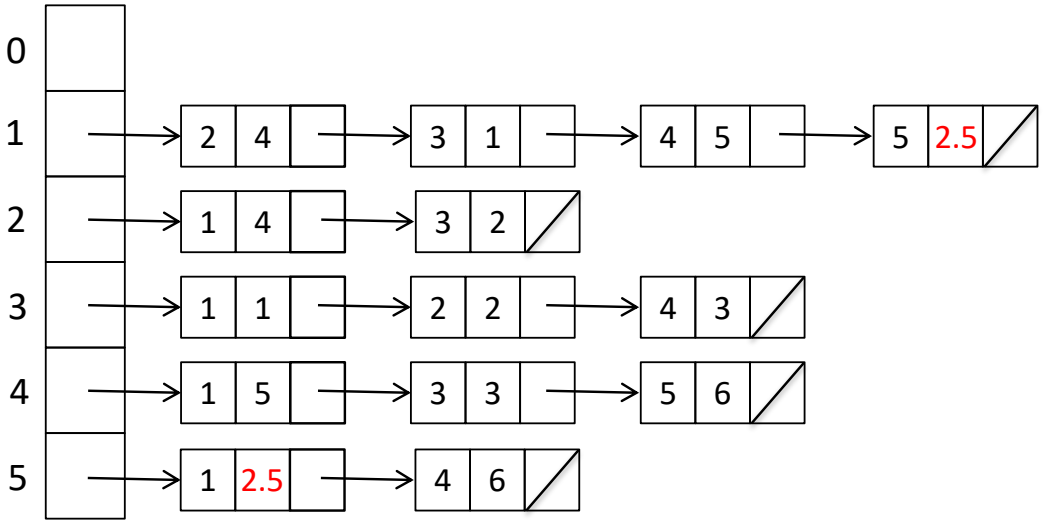


	intree	distance	parent
0			
1	T	0	-1
2	T	2	3
3	T	1	1
4	T	3	3
5	T	6	4

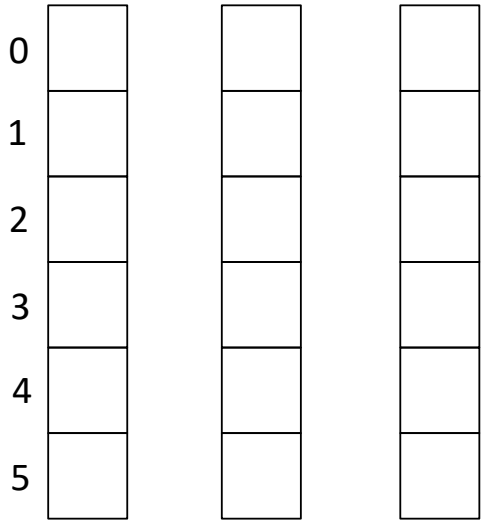
v	w	weight	dist
1	...	...	∞
1	4	6	4
2			1
3			∞
1			2
2			∞
1			3
4			∞
1			6
5			∞
1			



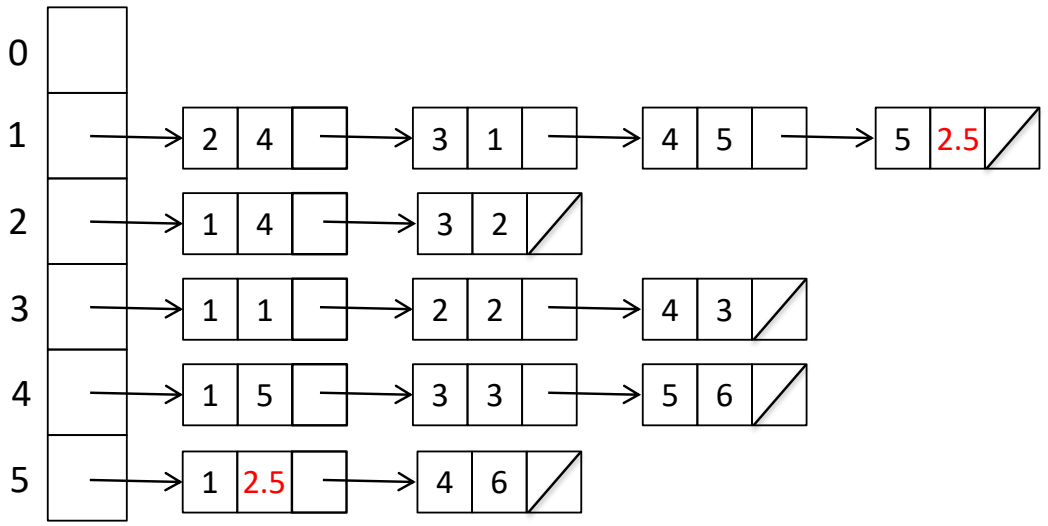


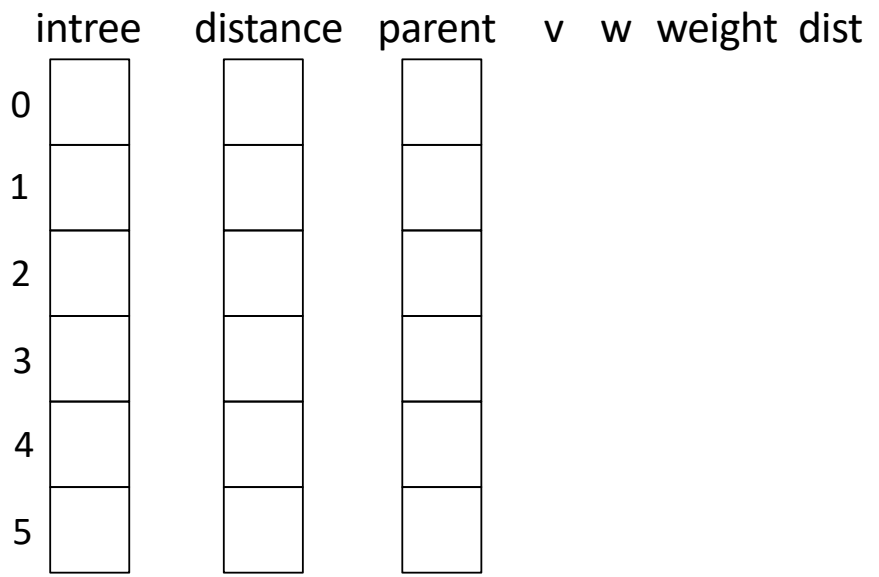


intree distance parent v w weight dist

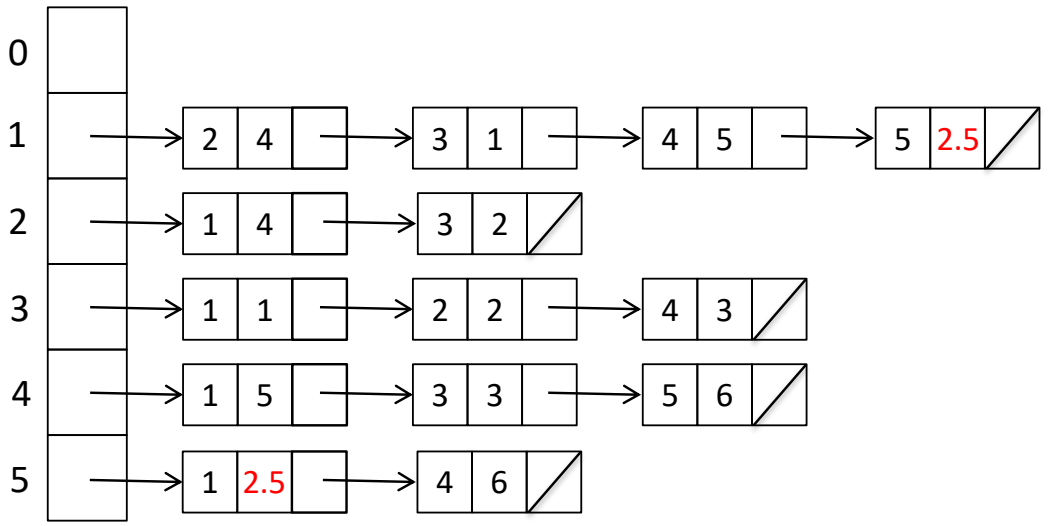


```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```





```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```

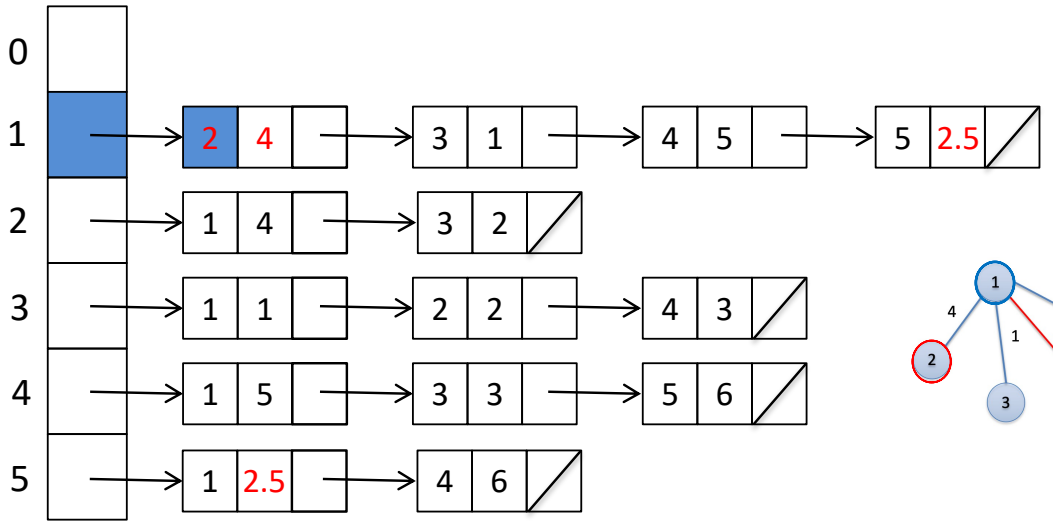


	intree	distance	parent
0			
1	F	∞	-1
2	F	∞	-1
3	F	∞	-1
4	F	∞	-1
5	F	∞	-1

v w weight dist

```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```

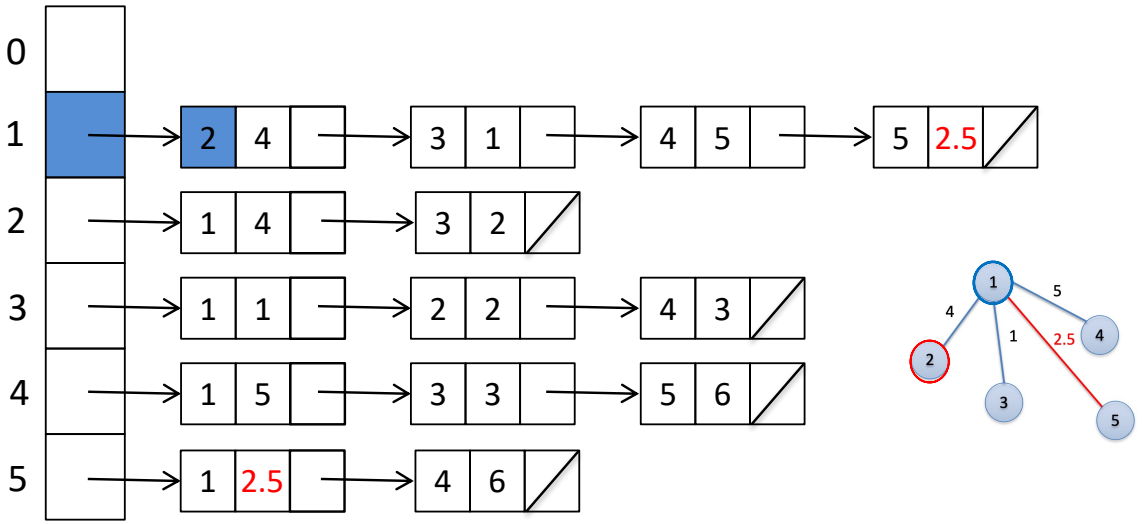
start = 1



	intree	distance	parent
0			
1	T	0	-1
2	F	∞	-1
3	F	∞	-1
4	F	∞	-1
5	F	∞	-1

v	w	weight	dist
1	2	4	

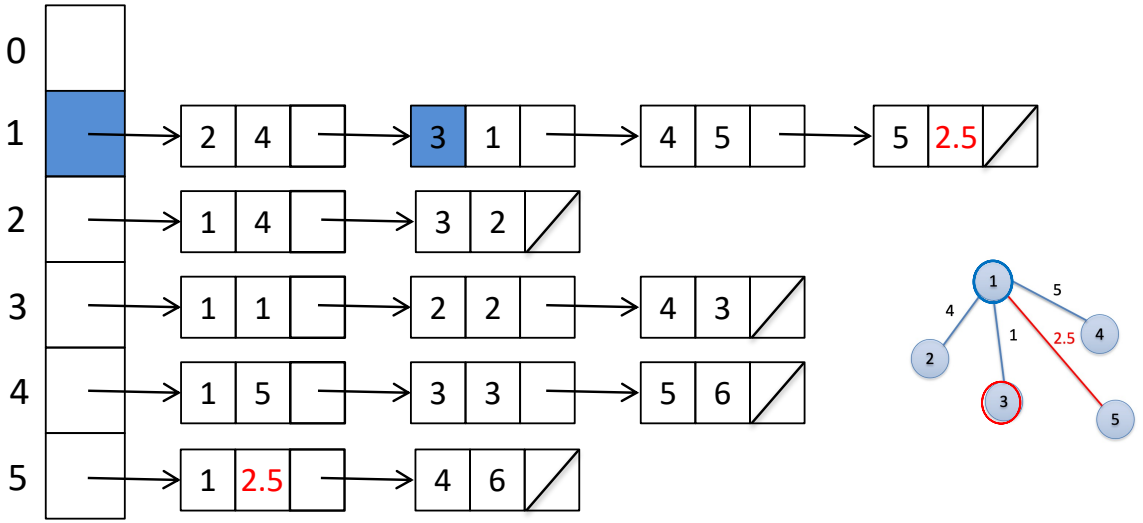
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



	intree	distance	parent
0			
1	T	0	-1
2	F	4	1
3	F	∞	-1
4	F	∞	-1
5	F	∞	-1

v	w	weight	dist
1	2	4	

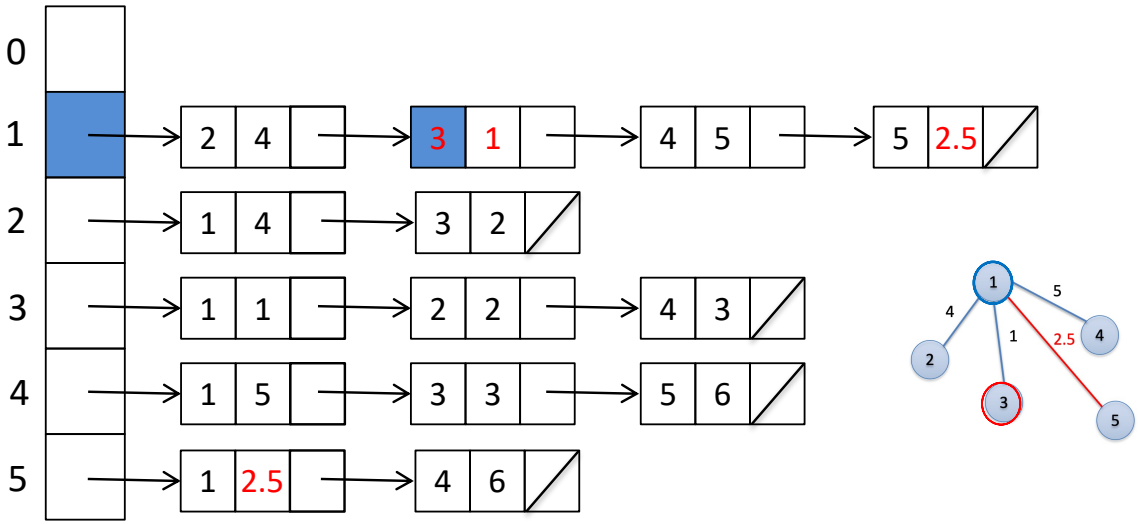
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



	intree	distance	parent
0			
1	T	0	-1
2	F	4	1
3	F	∞	-1
4	F	∞	-1
5	F	∞	-1

v	w	weight	dist
1	2	4	

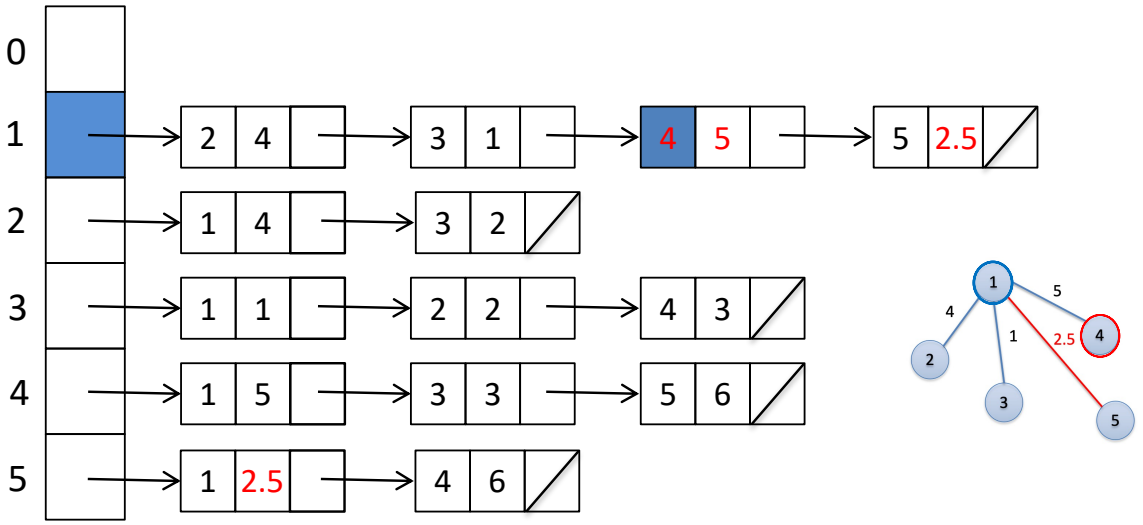
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



	intree	distance	parent
0			
1	T	0	-1
2	F	4	1
3	F	1	1
4	F	∞	-1
5	F	∞	-1

v	w	weight	dist
1	~~2~~	4	
	3	1	

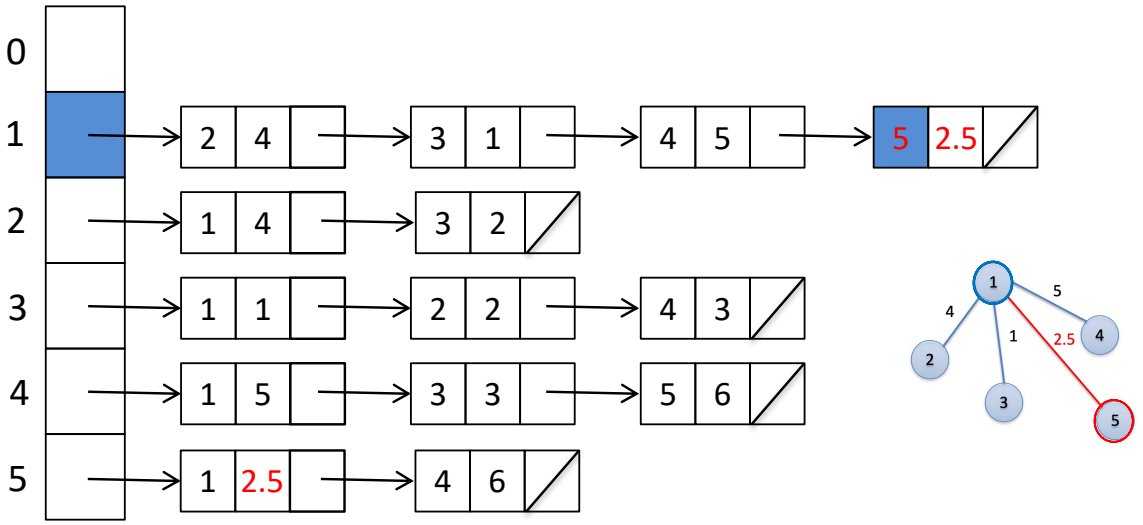
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



	intree	distance	parent
0			
1	T	0	-1
2	F	4	1
3	F	1	1
4	F	5	1
5	F	∞	-1

v	w	weight	dist
1	~~2~~	~~4~~	
	~~3~~	~~1~~	
	4	5	

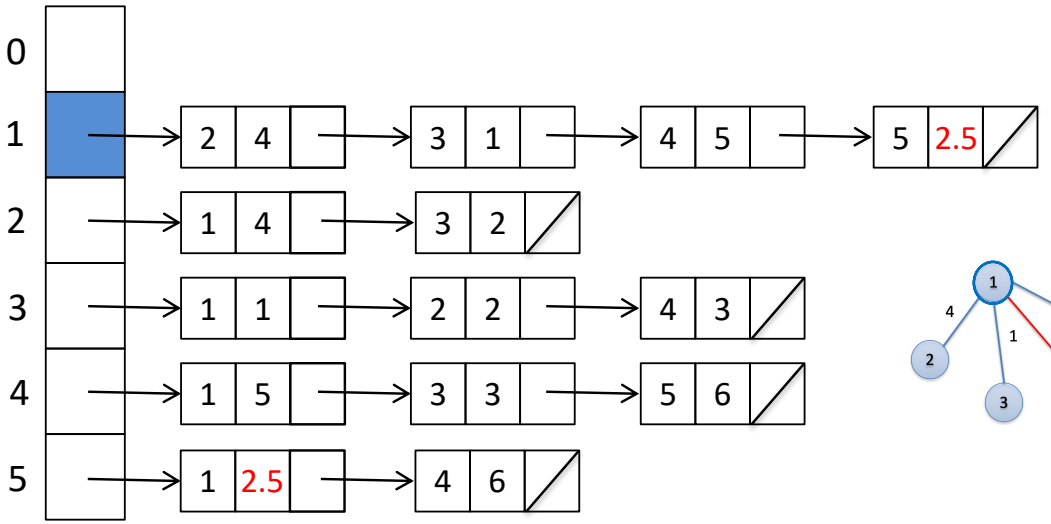
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



	intree	distance	parent
0			
1	T	0	-1
2	F	4	1
3	F	1	1
4	F	5	1
5	F	2.5	1

v	w	weight	dist
1	~~2~~	~~4~~	
	~~3~~	~~1~~	
	4	~~5~~	
	5	2.5	

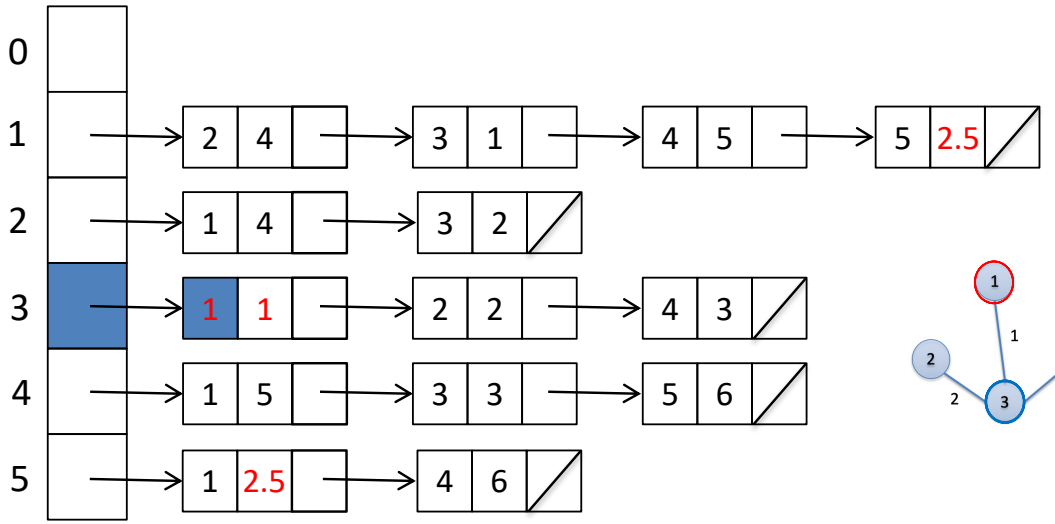
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



	intree	distance	parent
0			
1	T	0	-1
2	F	4	1
3	F	1	1
4	F	5	1
5	F	2.5	1

v	w	weight	dist
~~1~~	~~2~~	~~4~~	∞
1	~~3~~	~~1~~	4
2	4	5	1
3			

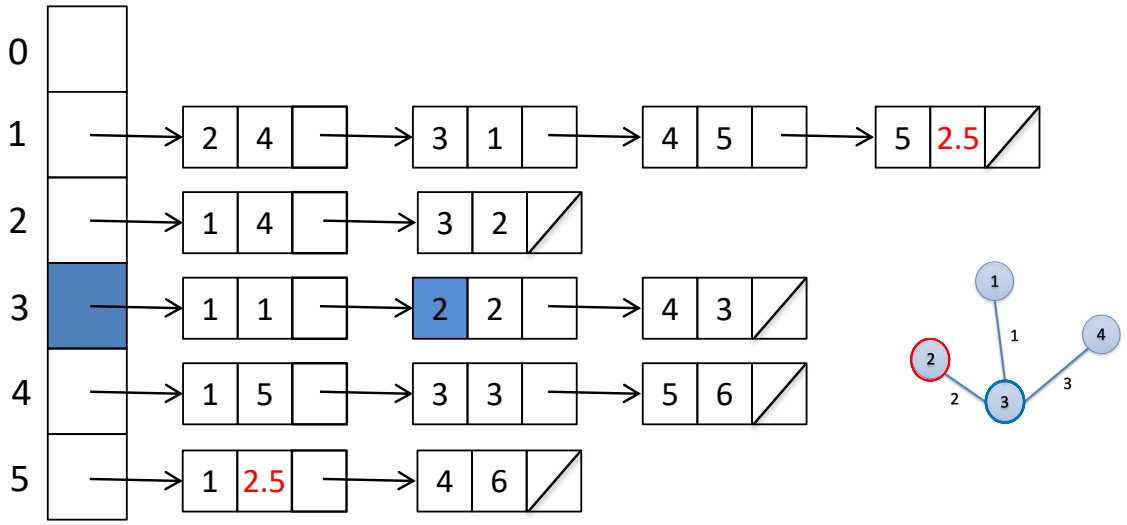
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



	intree	distance	parent
0			
1	T	0	-1
2	F	4	1
3	T	1	1
4	F	5	1
5	F	2.5	1

v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	

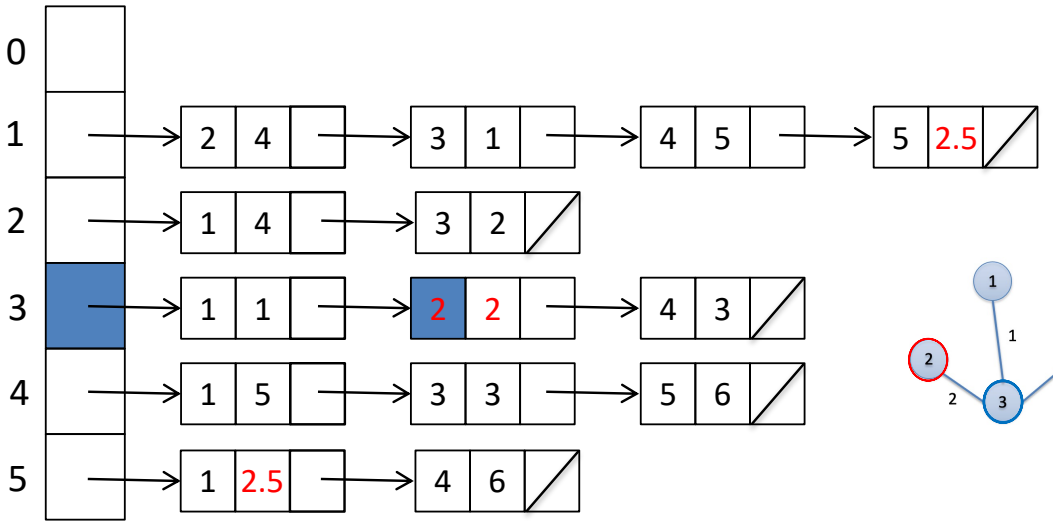
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



	intree	distance	parent
0			
1	T	0	-1
2	F	4	1
3	T	1	1
4	F	5	1
5	F	2.5	1

v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	

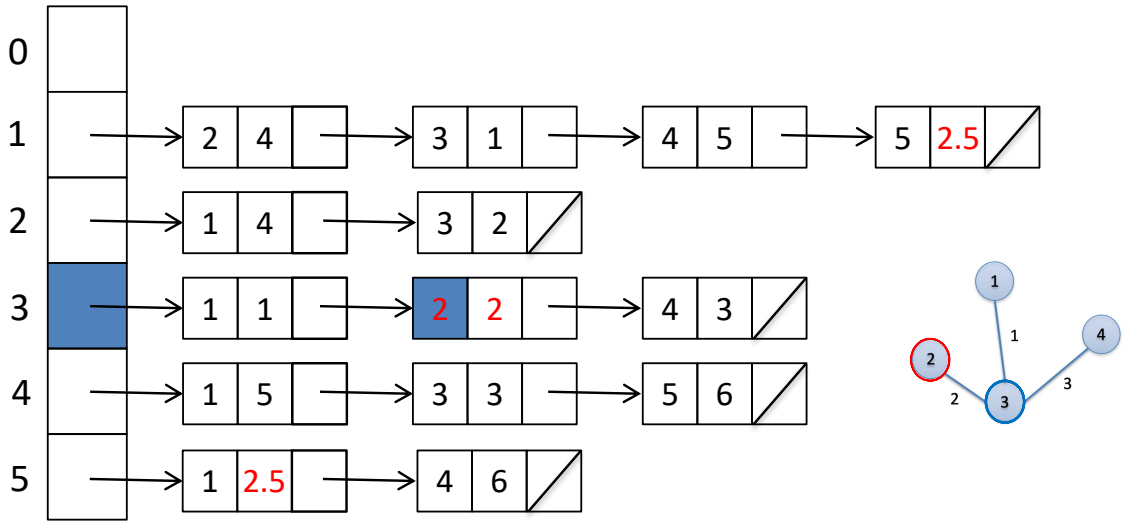
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



	intree	distance	parent
0			
1	T	0	-1
2	F	4	1
3	T	1	1
4	F	5	1
5	F	2.5	1

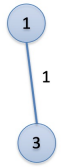
v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	
	2	2	

```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```

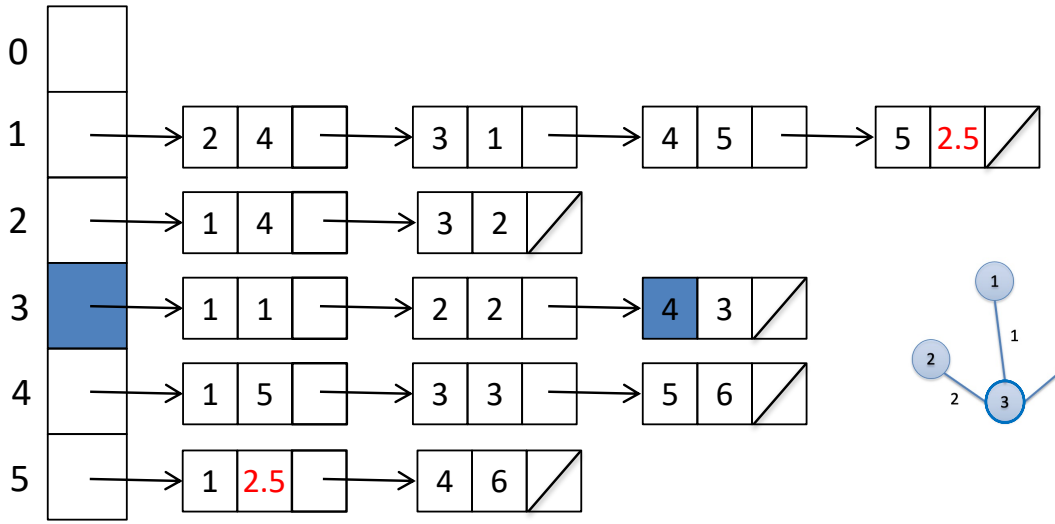


	intree	distance	parent
0			
1	T	0	-1
2	F	2	3
3	T	1	1
4	F	5	1
5	F	2.5	1

v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	
	2	2	



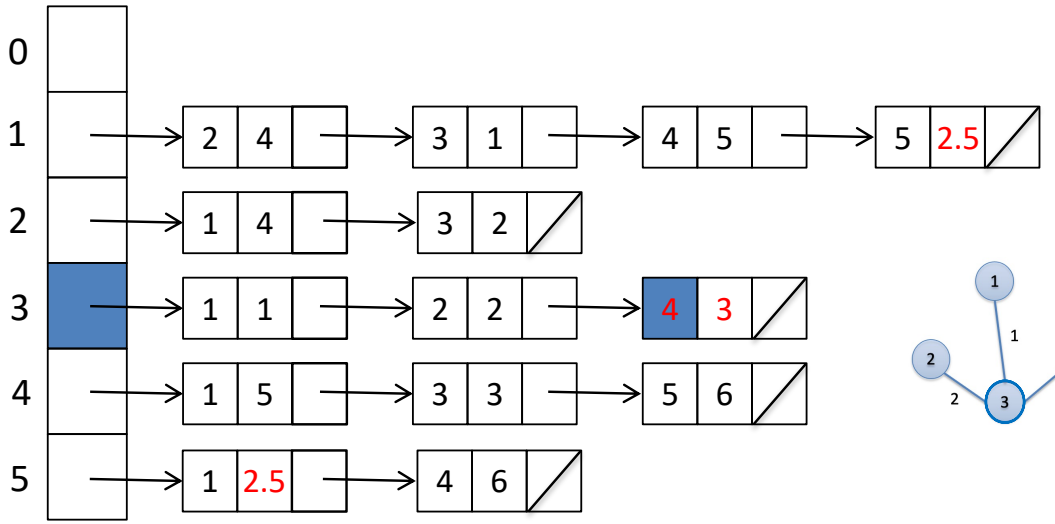
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



	intree	distance	parent
0			
1	T	0	-1
2	F	2	3
3	T	1	1
4	F	5	1
5	F	2.5	1

v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	
	2	2	

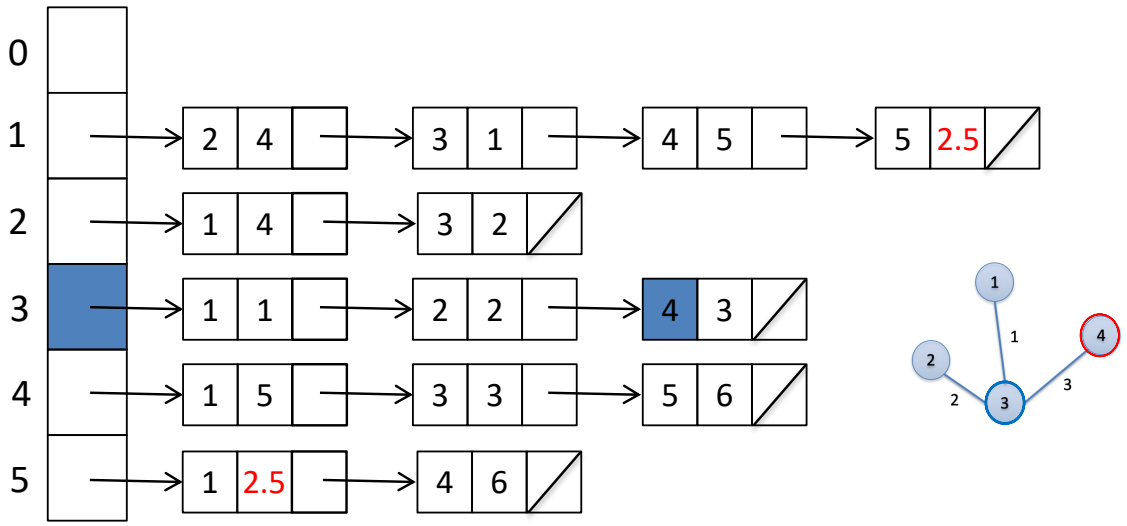
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



	intree	distance	parent
0			
1	T	0	-1
2	F	2	3
3	T	1	1
4	F	5	1
5	F	2.5	1

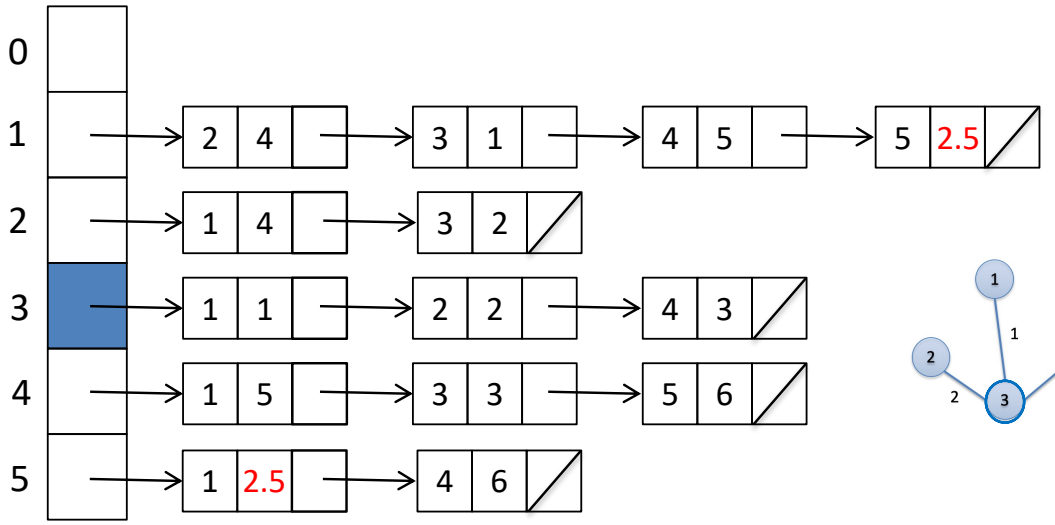
v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	
	2	2	
	4	3	

```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



	intree	distance	parent	v	w	weight	dist
0				1	2	4	∞
1	T	0	-1	1	3	1	4
2	F	2	3	2	4	5	1
3	T	1	1	3	1	1	
4	F	3	3		2	2	
5	F	2.5	1		4	3	

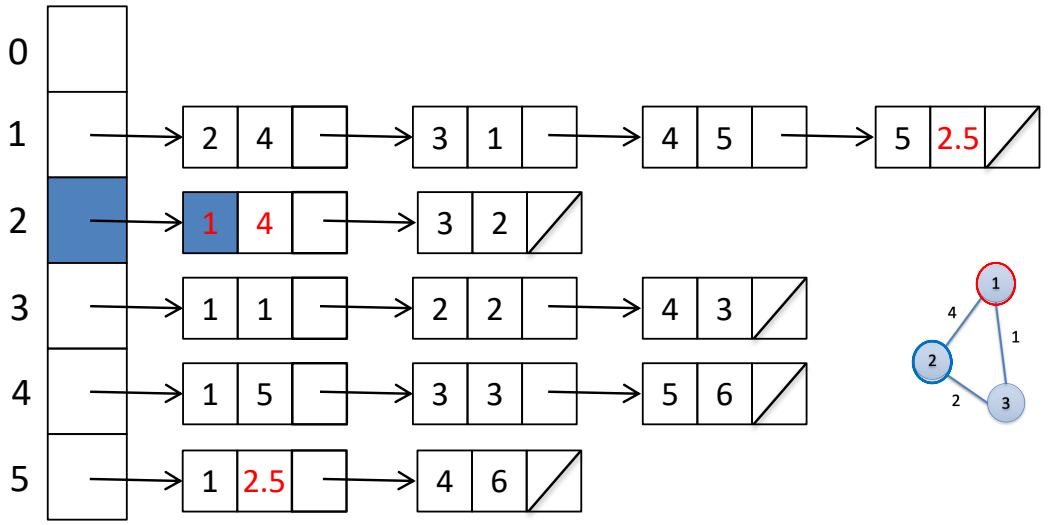
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



	intree	distance	parent
0			
1	T	0	-1
2	F	2	3
3	T	1	1
4	F	3	3
5	F	2.5	1

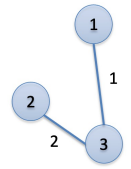
v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	∞
1	2	2	2
2	4	3	

```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```

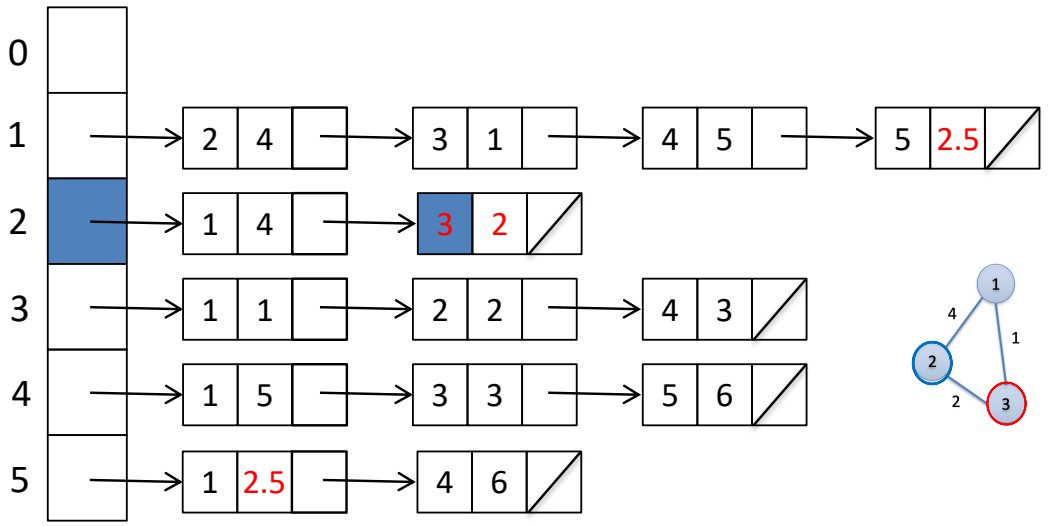


	intree	distance	parent
0			
1	T	0	-1
2	T	2	3
3	T	1	1
4	F	3	3
5	F	2.5	1

v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	∞
1	2	2	2
2	4	3	
	1	4	

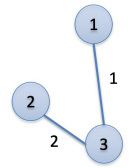


```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```

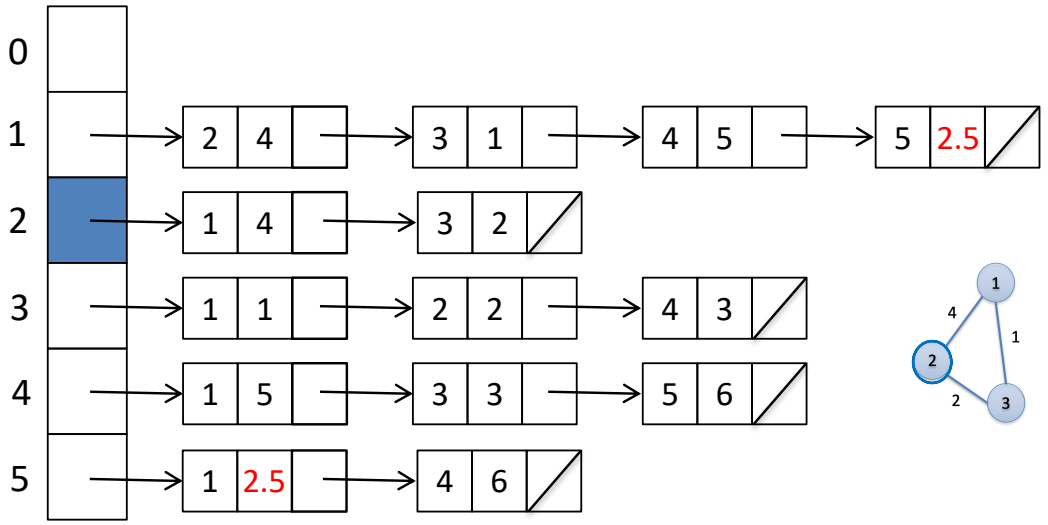


	intree	distance	parent
0			
1	T	0	-1
2	T	2	3
3	T	1	1
4	F	3	3
5	F	2.5	1

v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	∞
1	2	2	2
2	4	3	
	1	4	
	3	2	

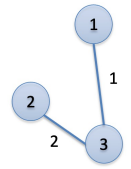


```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```

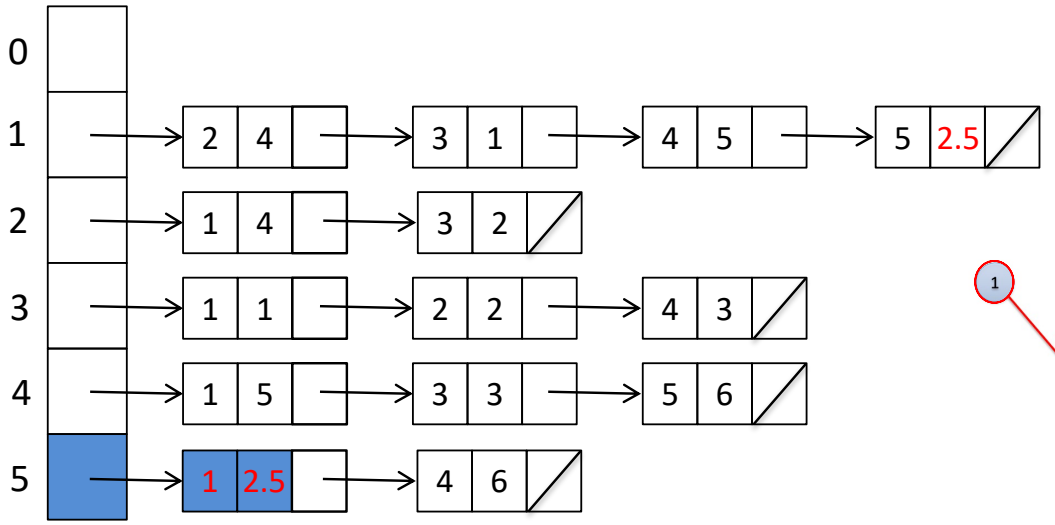


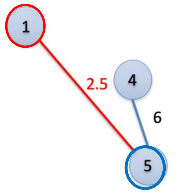
	intree	distance	parent
0			
1	T	0	-1
2	T	2	3
3	T	1	1
4	F	3	3
5	F	2.5	1

v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	∞
1	2	2	2
2	4	3	∞
1	1	4	3
4	3	2	2.5
5			



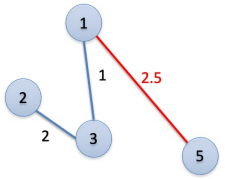
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



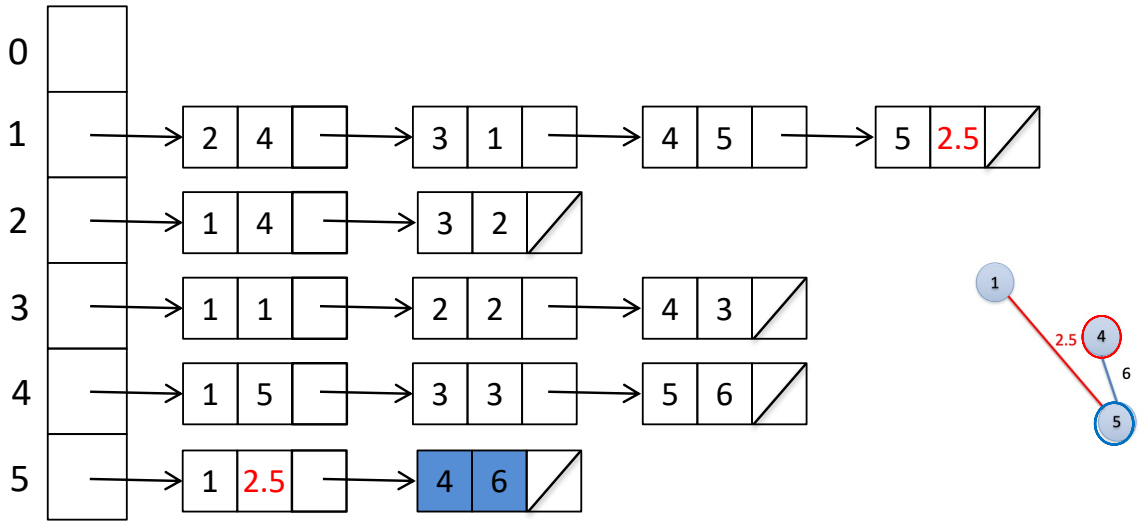


	intree	distance	parent
0			
1	T	0	-1
2	T	2	3
3	T	1	1
4	F	3	3
5	T	2.5	1

v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	∞
1	2	2	2
2	4	3	∞
1	1	4	3
4	3	2	2.5
5	1	2.5	

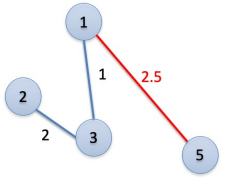


```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```

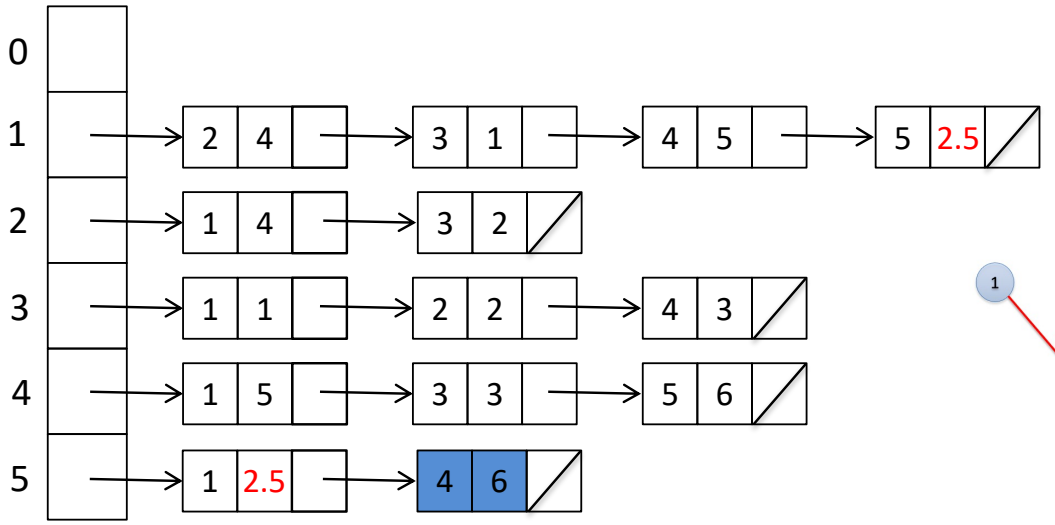


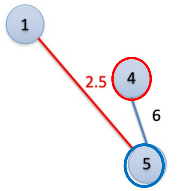
	intree	distance	parent
0			
1	T	0	-1
2	T	2	3
3	T	1	1
4	F	3	3
5	T	2.5	1

v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	∞
1	2	2	2
2	4	3	∞
1	1	4	3
4	3	2	2.5
5	1	2.5	



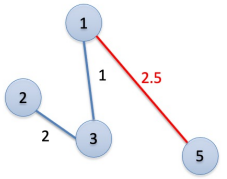
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



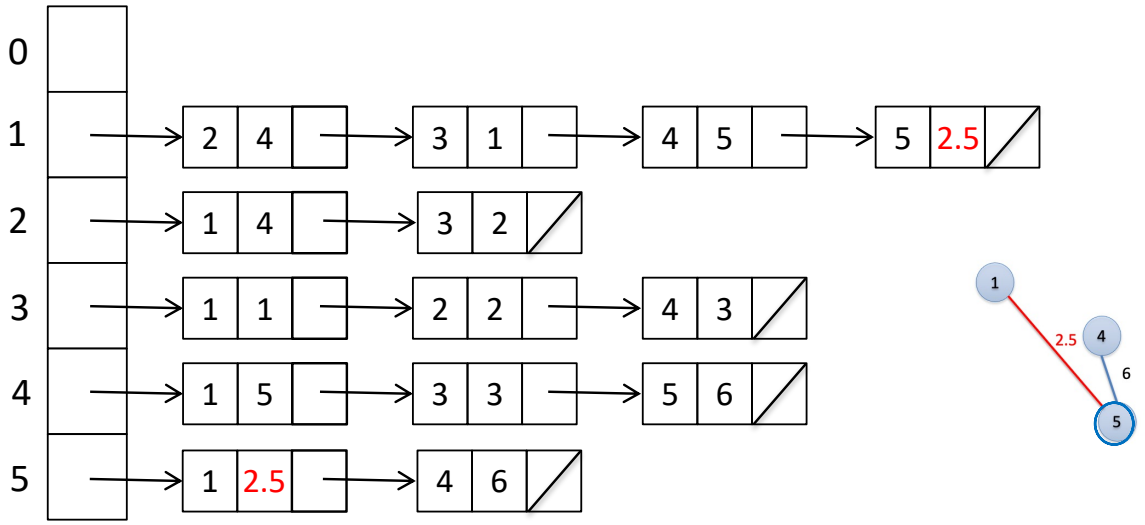


	intree	distance	parent
0			
1	T	0	-1
2	T	2	3
3	T	1	1
4	F	3	3
5	T	2.5	1

v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	∞
1	2	2	2
2	4	3	∞
1	1	4	3
4	3	2	2.5
5	1	2.5	
	4	6	

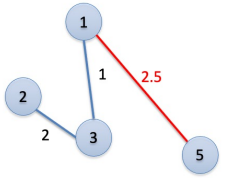


```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```

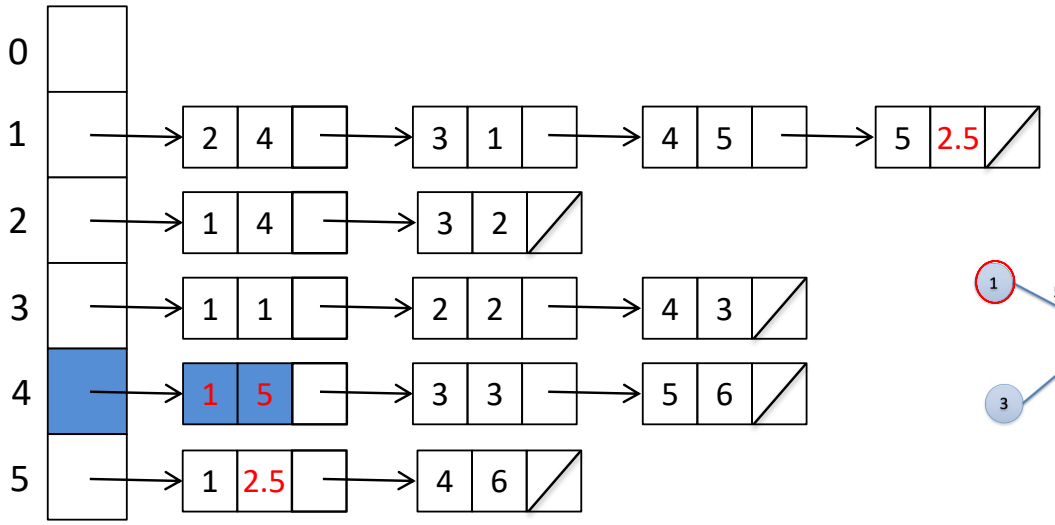


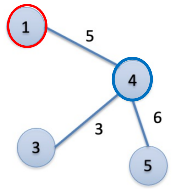
	intree	distance	parent
0			
1	T	0	-1
2	T	2	3
3	T	1	1
4	F	3	3
5	T	2.5	1

v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	∞
1	2	2	2
2	4	3	∞
1	1	4	3
4	3	2	2.5
5	1	2.5	3
4	4	6	



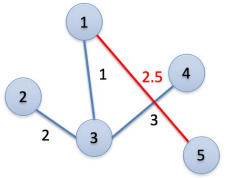
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



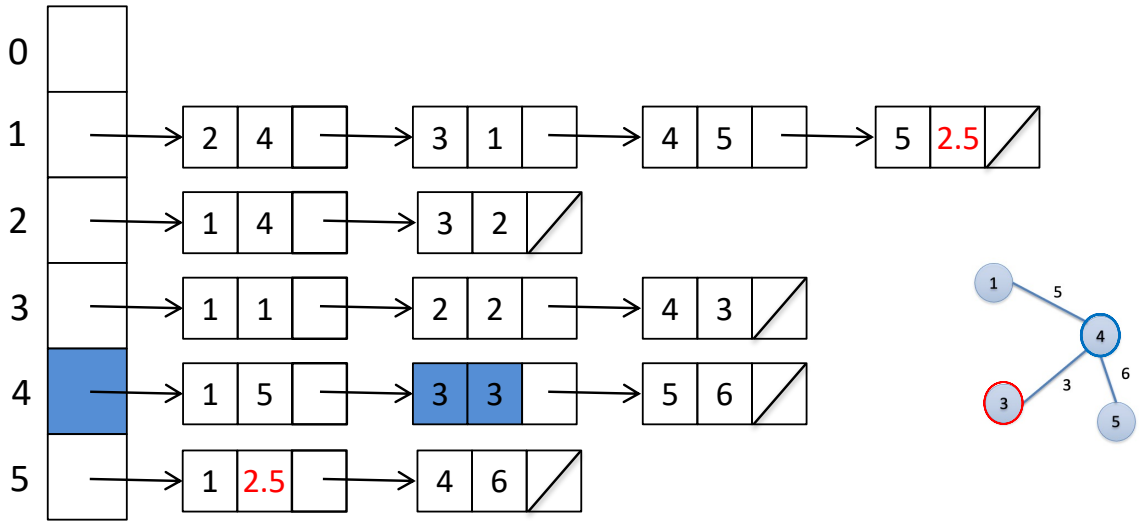


	intree	distance	parent
0			
1	T	0	-1
2	T	2	3
3	T	1	1
4	T	3	3
5	T	2.5	1

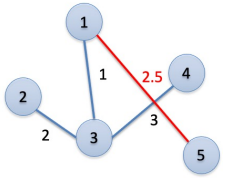
v	w	weight	dist
...	...	...	...
5	1	5	



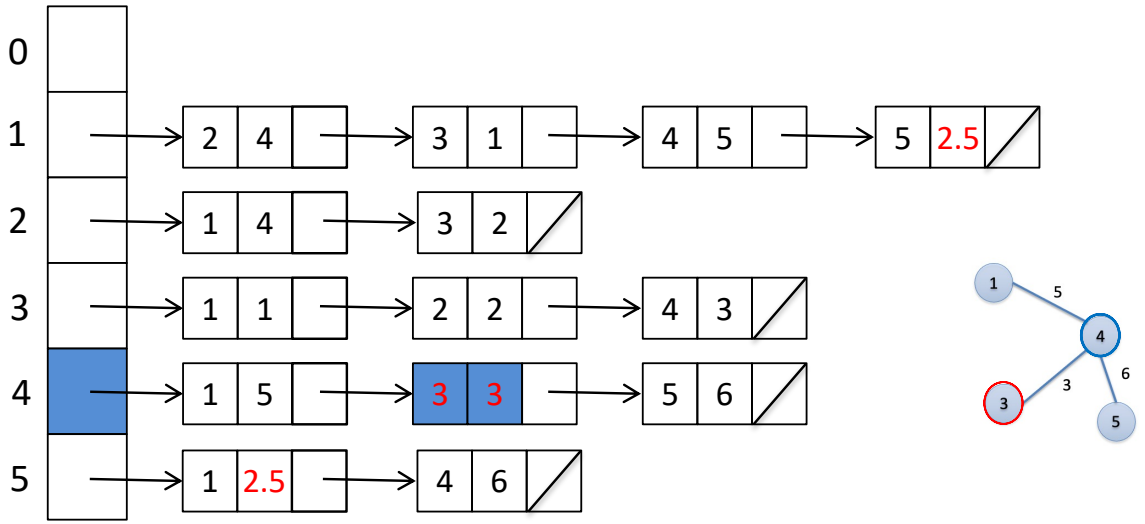
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



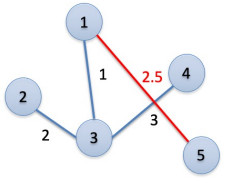
	intree	distance	parent	v	w	weight	dist
0				...	...	...	...
1	T	0	-1	5	1	5	
2	T	2	3				
3	T	1	1				
4	T	3	3				
5	T	2.5	1				



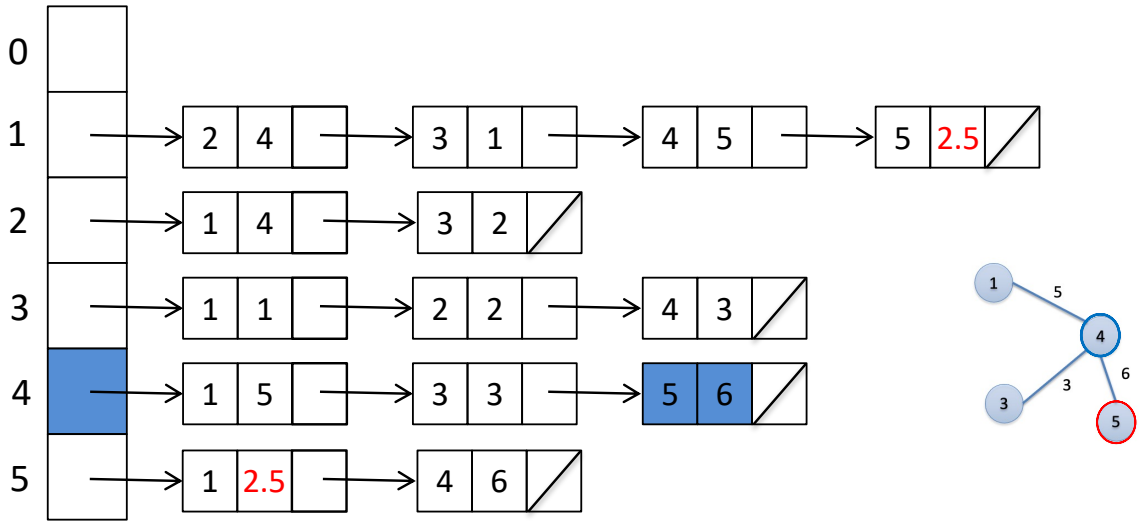
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



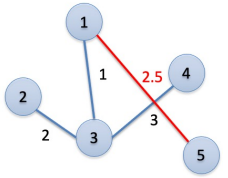
	intree	distance	parent	v	w	weight	dist
0				...	...	...	...
1	T	0	-1	5	1	5	
2	T	2	3		3	3	
3	T	1	1				
4	T	3	3				
5	T	2.5	1				



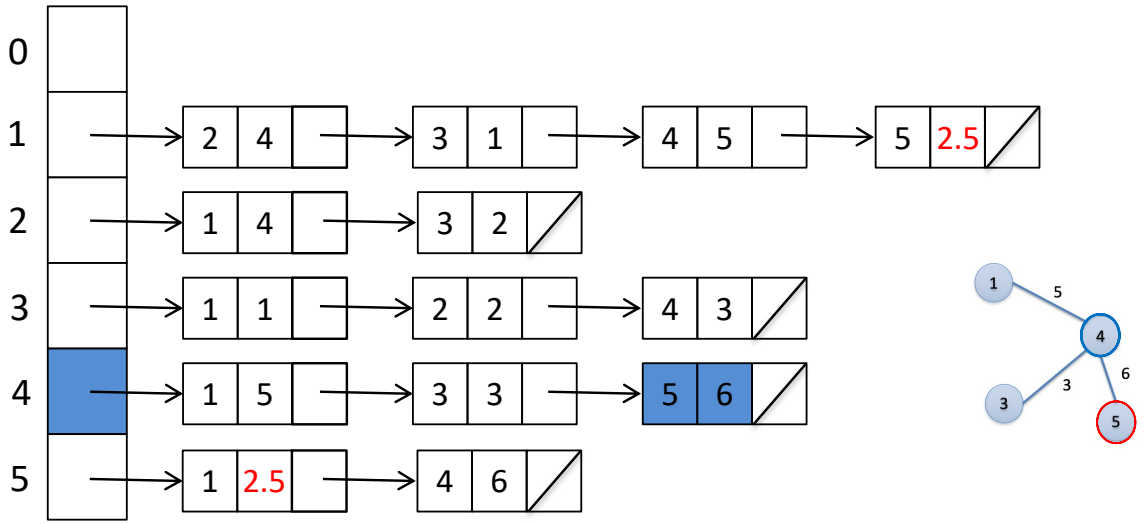
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



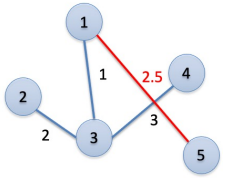
	intree	distance	parent	v	w	weight	dist
0				...	...	...	...
1	T	0	-1	5	1	5	
2	T	2	3		3	3	
3	T	1	1				
4	T	3	3				
5	T	2.5	1				



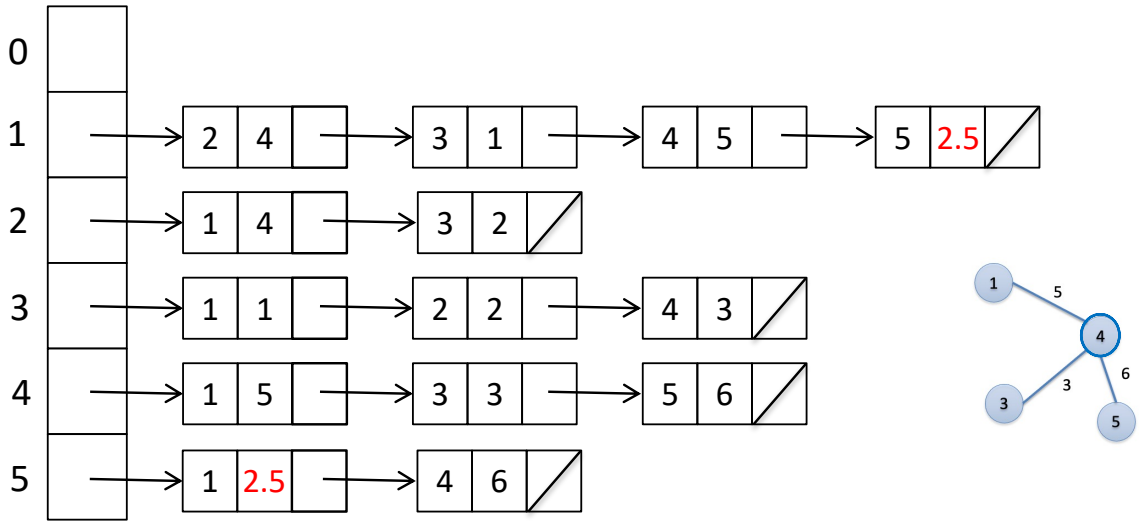
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



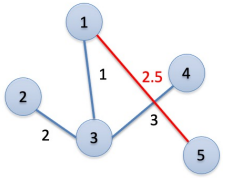
	intree	distance	parent	v	w	weight	dist
0				...	...	...	...
1	T	0	-1	5	1	5	
2	T	2	3		3	3	
3	T	1	1		5	6	
4	T	3	3				
5	T	2.5	1				



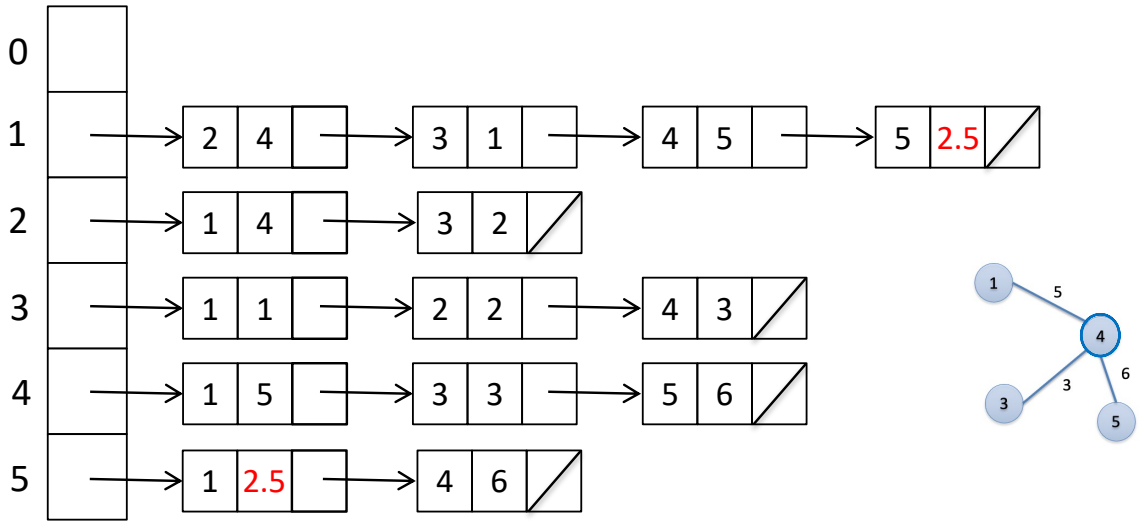
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



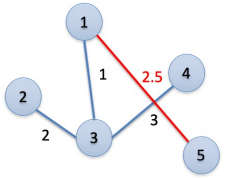
	intree	distance	parent	v	w	weight	dist
0				...	...	...	...
1	T	0	-1	5	1	5	
2	T	2	3	1	3	3	
3	T	1	1		5	6	
4	T	3	3				
5	T	2.5	1				

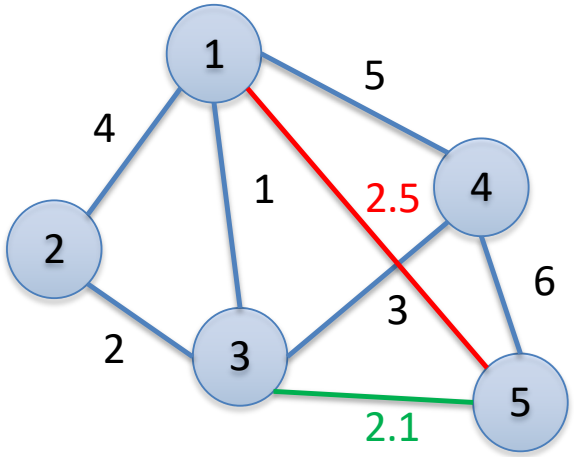


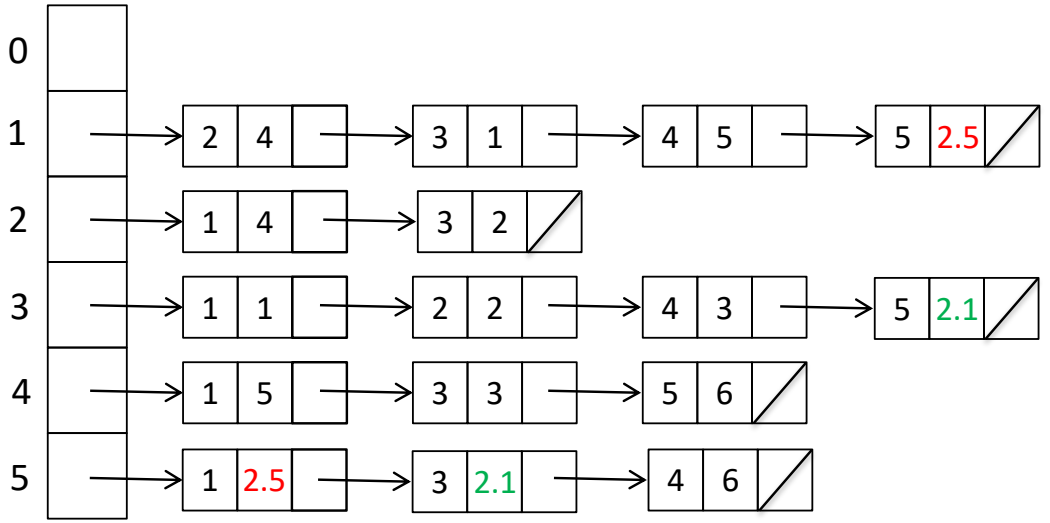
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```

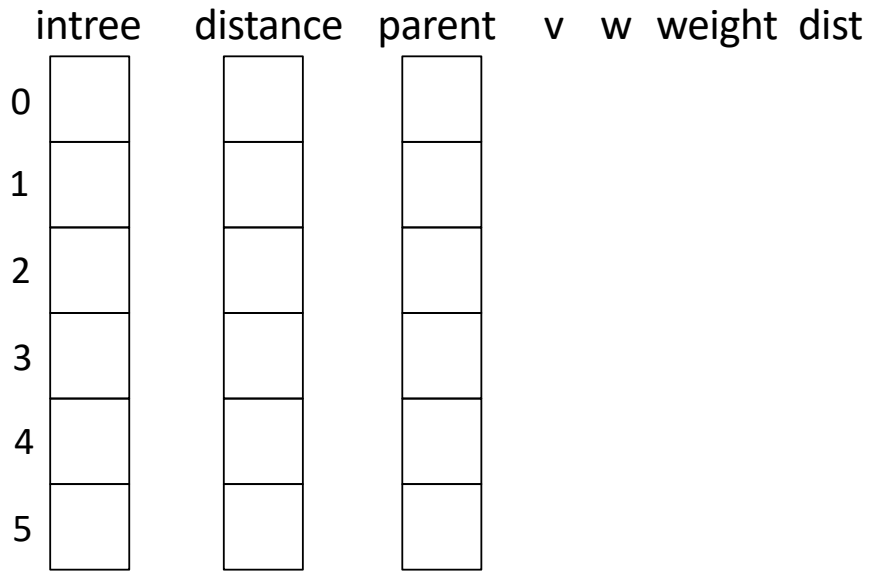


	intree	distance	parent	v	w	weight	dist
0				...	...	...	...
1	T	0	-1	5	1	5	
2	T	2	3	1	3	3	
3	T	1	1		5	6	
4	T	3	3				
5	T	2.5	1				

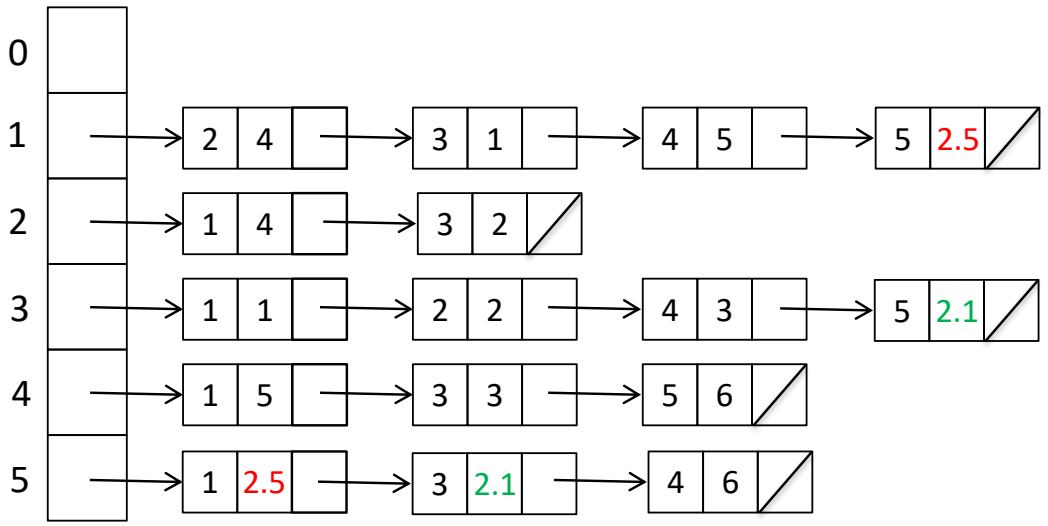


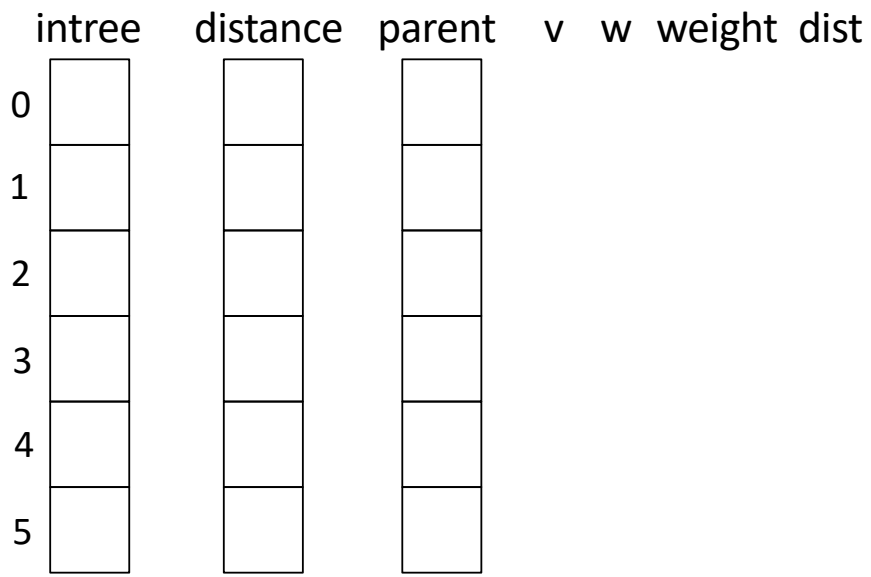

1
2
3
4
5
5
4
1
2.5
6
3
2
2.1


0
1
2
3
4
5
2 4
3 1
4 5
5 2.5
1 4
3 2
1 1
2 2
4 3
5 2.1
1 5
3 3
5 6
1 2.5
3 2.1
4 6

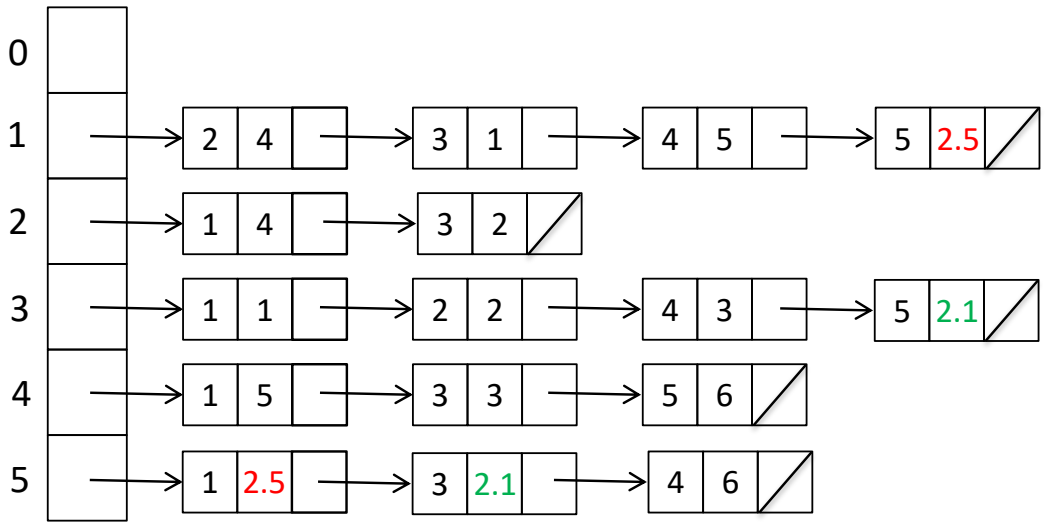

intree
distance
parent
v
w
weight
dist
0
1
2
3
4
5

```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```





```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```

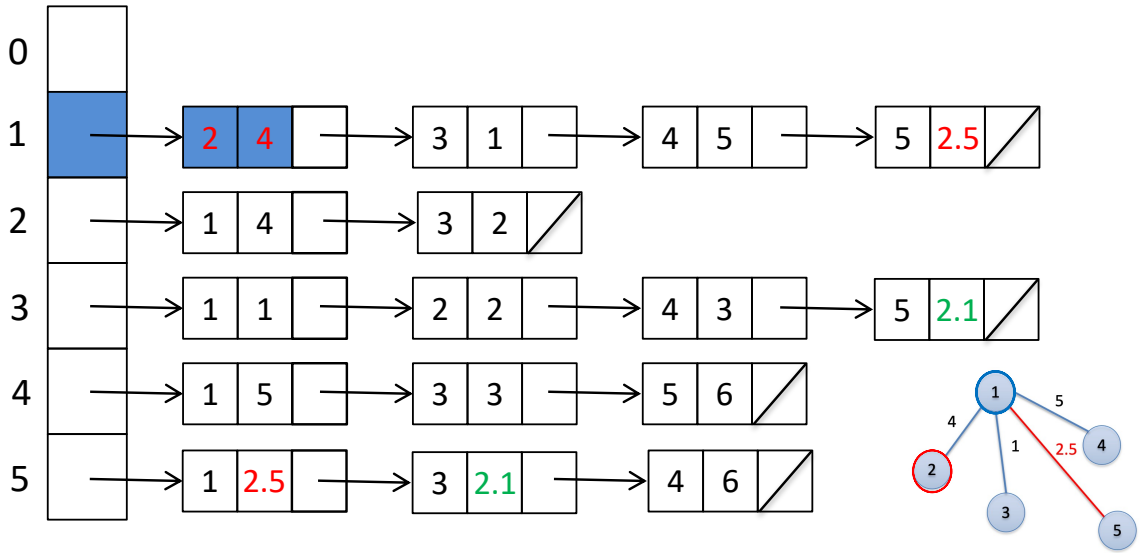


	intree	distance	parent
0			
1	F	∞	-1
2	F	∞	-1
3	F	∞	-1
4	F	∞	-1
5	F	∞	-1

v w weight dist

```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```

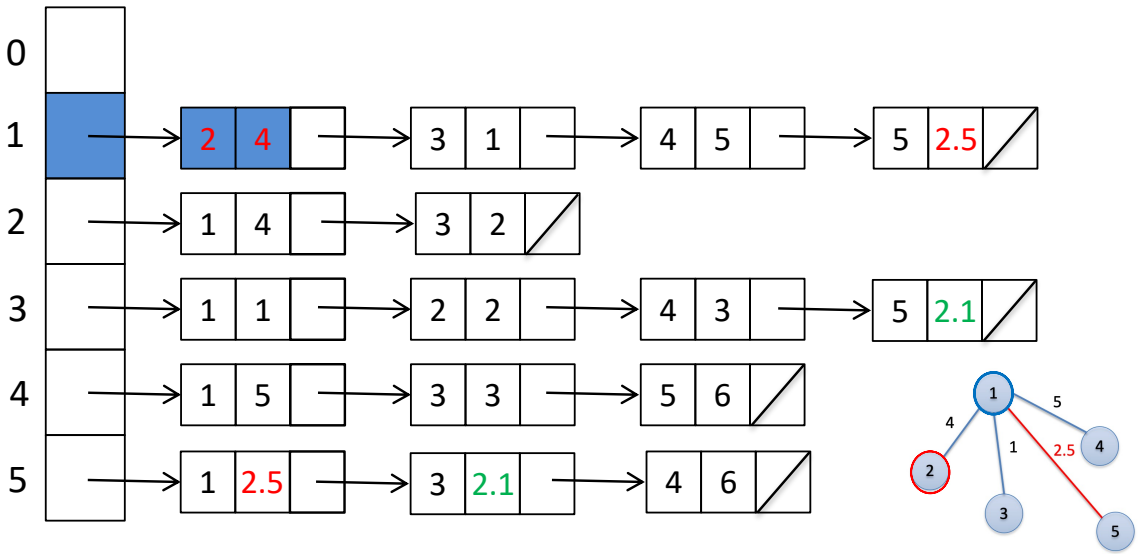
start = 1



	intree	distance	parent
0			
1	T	0	-1
2	F	∞	-1
3	F	∞	-1
4	F	∞	-1
5	F	∞	-1

v	w	weight	dist
1	2	4	

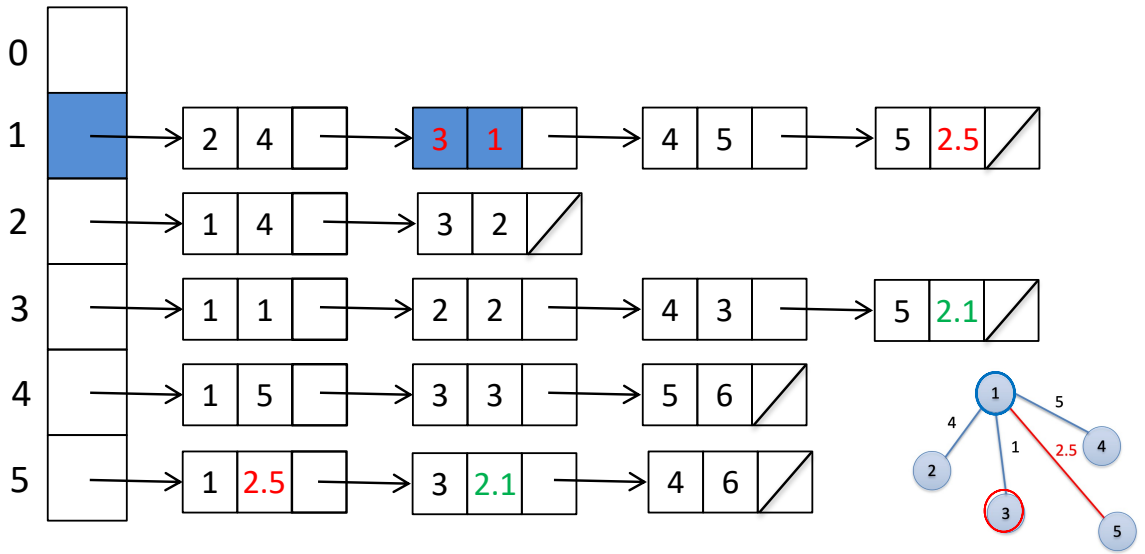
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



	intree	distance	parent
0			
1	T	0	-1
2	F	4	1
3	F	∞	-1
4	F	∞	-1
5	F	∞	-1

v	w	weight	dist
1	2	4	

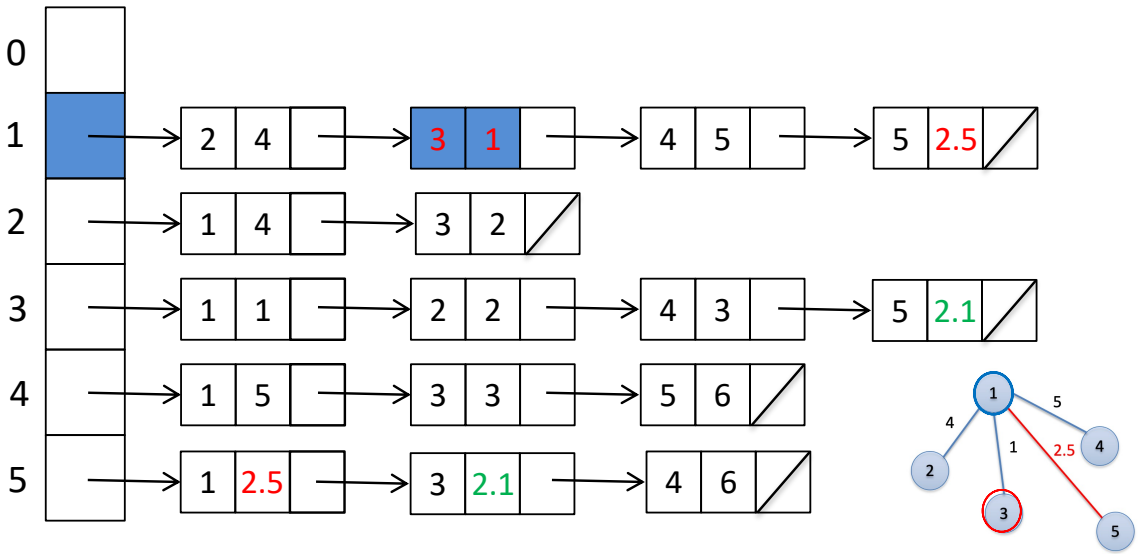
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



	intree	distance	parent
0			
1	T	0	-1
2	F	4	1
3	F	∞	-1
4	F	∞	-1
5	F	∞	-1

v	w	weight	dist
1	2	4	

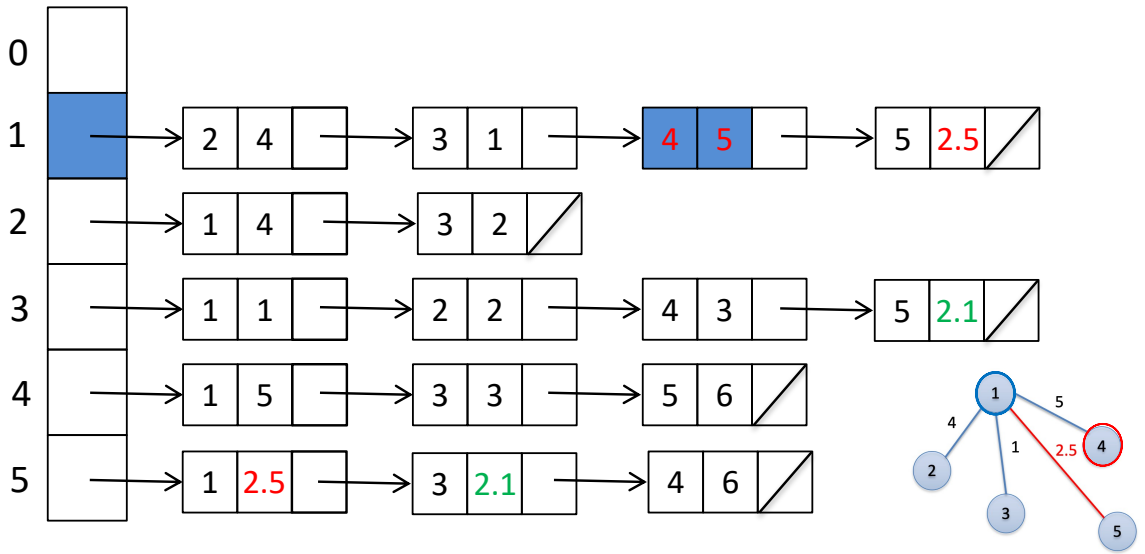
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



	intree	distance	parent
0			
1	T	0	-1
2	F	4	1
3	F	1	1
4	F	∞	-1
5	F	∞	-1

v	w	weight	dist
1	~~2~~	~~4~~	
	3	1	

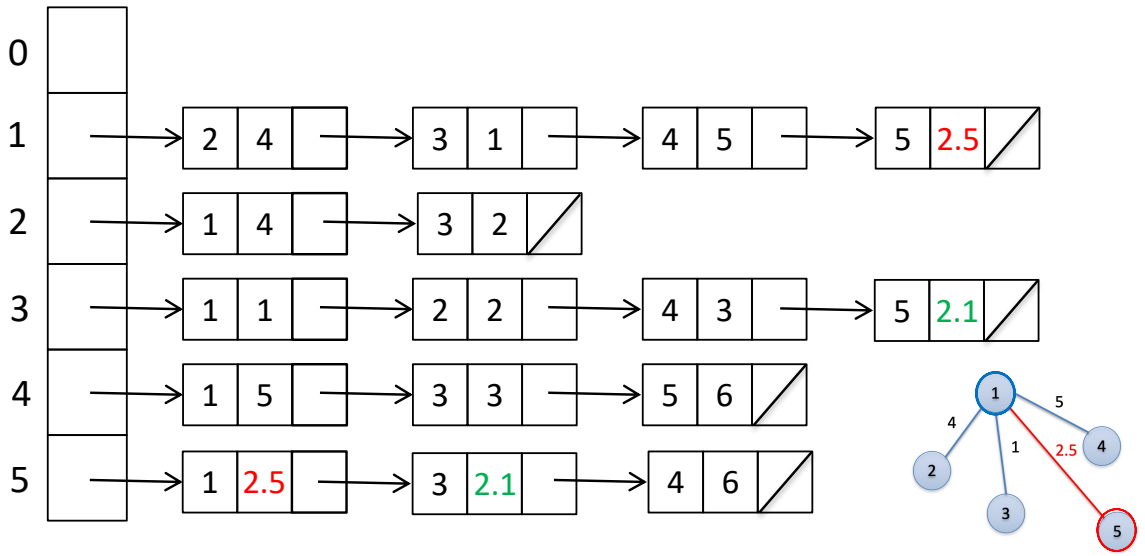
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



	intree	distance	parent
0			
1	T	0	-1
2	F	4	1
3	F	1	1
4	F	5	1
5	F	∞	-1

v	w	weight	dist
1	~~2~~	~~4~~	
	~~3~~	~~1~~	
	4	5	

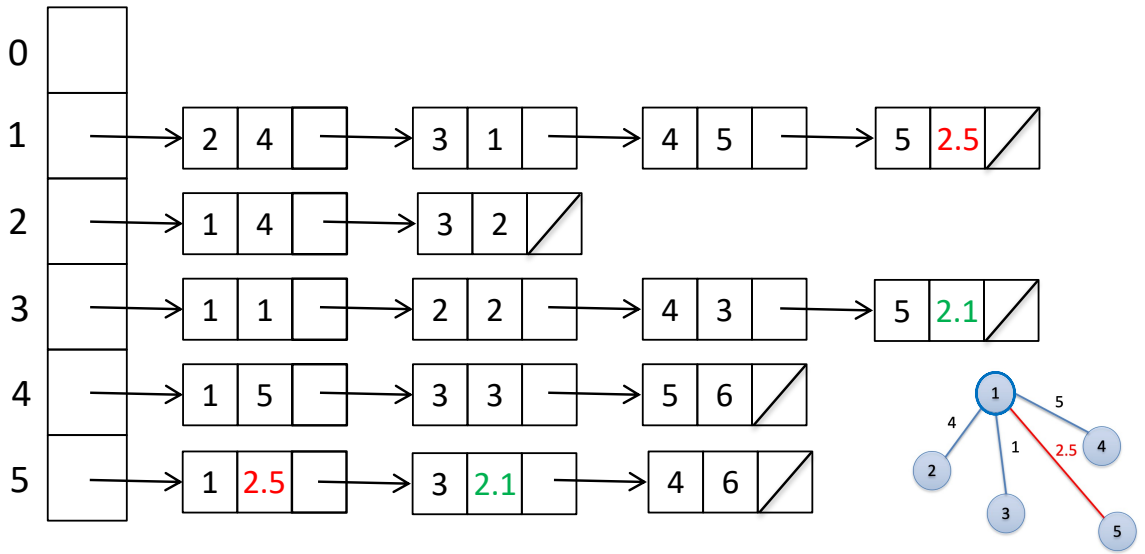
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



	intree	distance	parent
0			
1	T	0	-1
2	F	4	1
3	F	1	1
4	F	5	1
5	F	2.5	1

v	w	weight	dist
1	~~2~~	~~4~~	
	~~3~~	~~1~~	
	4	~~5~~	
	5	2.5	

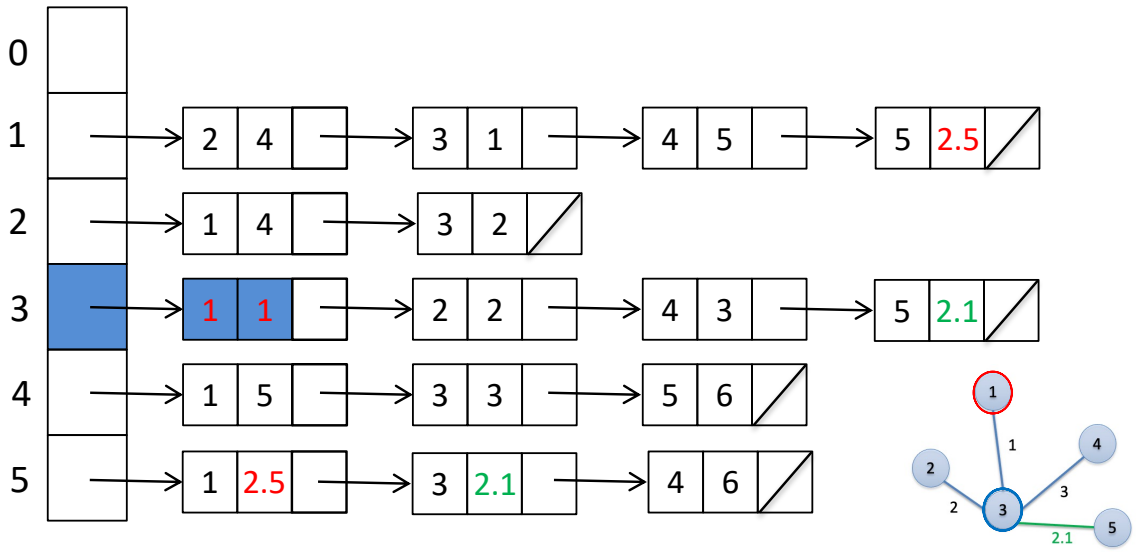
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



	intree	distance	parent
0			
1	T	0	-1
2	F	4	1
3	F	1	1
4	F	5	1
5	F	2.5	1

v	w	weight	dist
~~1~~	~~2~~	~~4~~	∞
1	~~3~~	~~1~~	4
2	4	5	1
3			

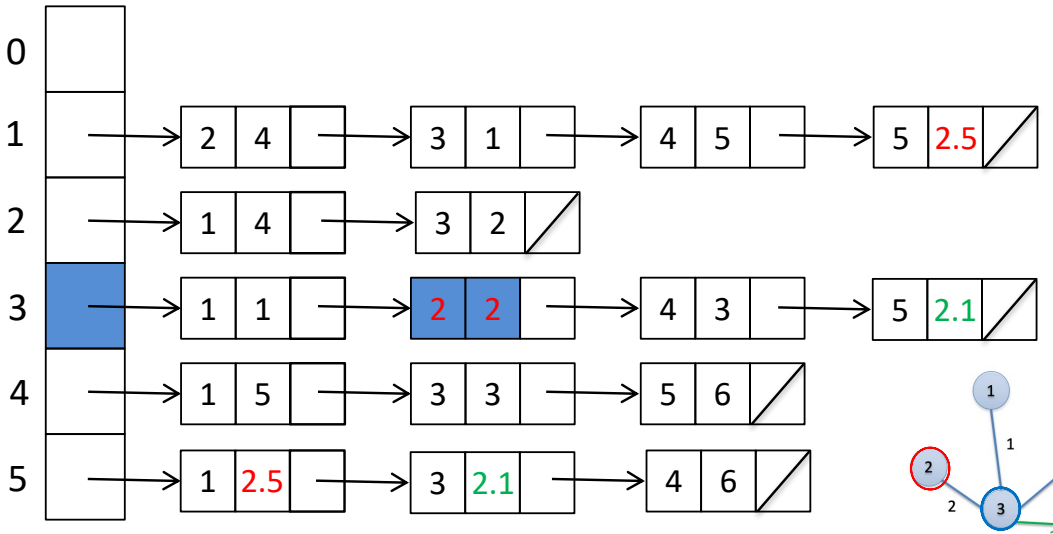
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



	intree	distance	parent
0			
1	T	0	-1
2	F	4	1
3	T	1	1
4	F	5	1
5	F	2.5	1

v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	

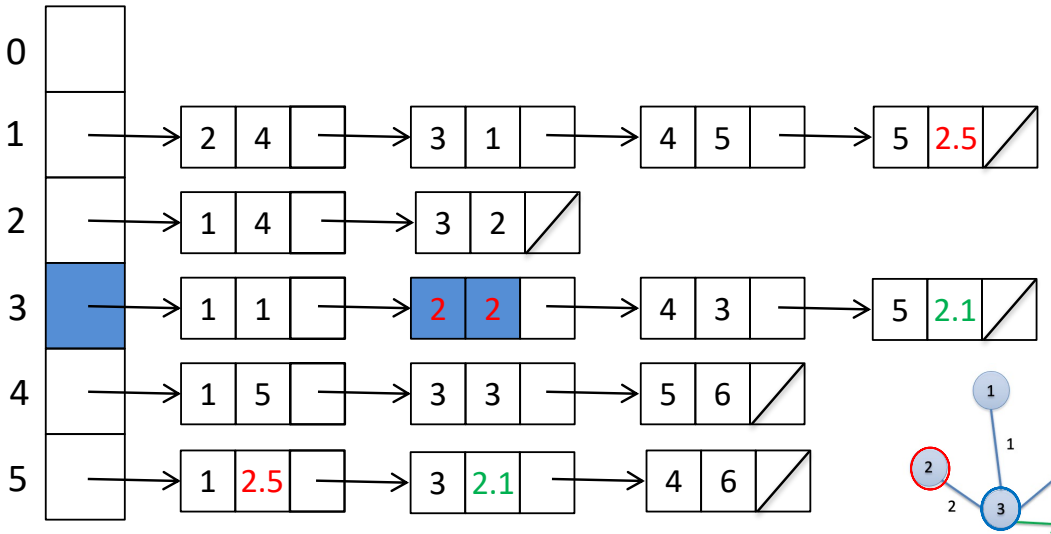
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



	intree	distance	parent
0			
1	T	0	-1
2	F	4	1
3	T	1	1
4	F	5	1
5	F	2.5	1

v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	

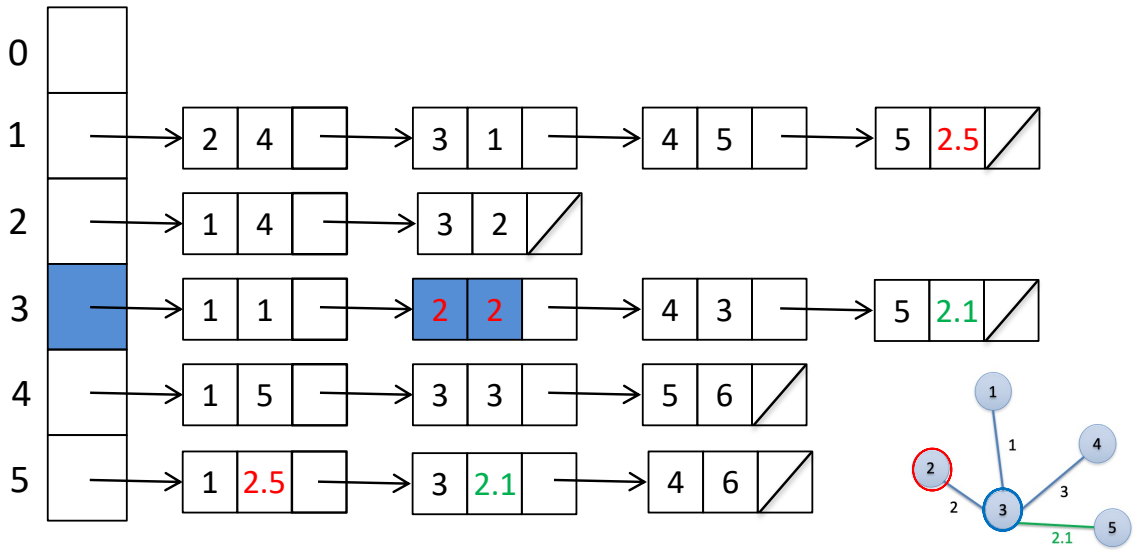
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



	intree	distance	parent
0			
1	T	0	-1
2	F	4	1
3	T	1	1
4	F	5	1
5	F	2.5	1

v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	
	2	2	

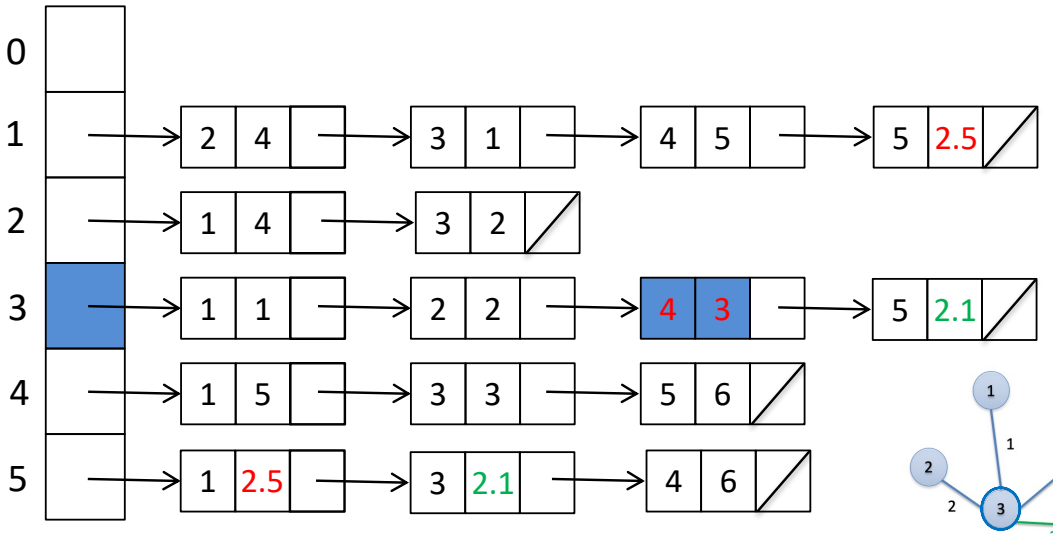
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



	intree	distance	parent
0			
1	T	0	-1
2	F	2	3
3	T	1	1
4	F	5	1
5	F	2.5	1

v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	
	2	2	

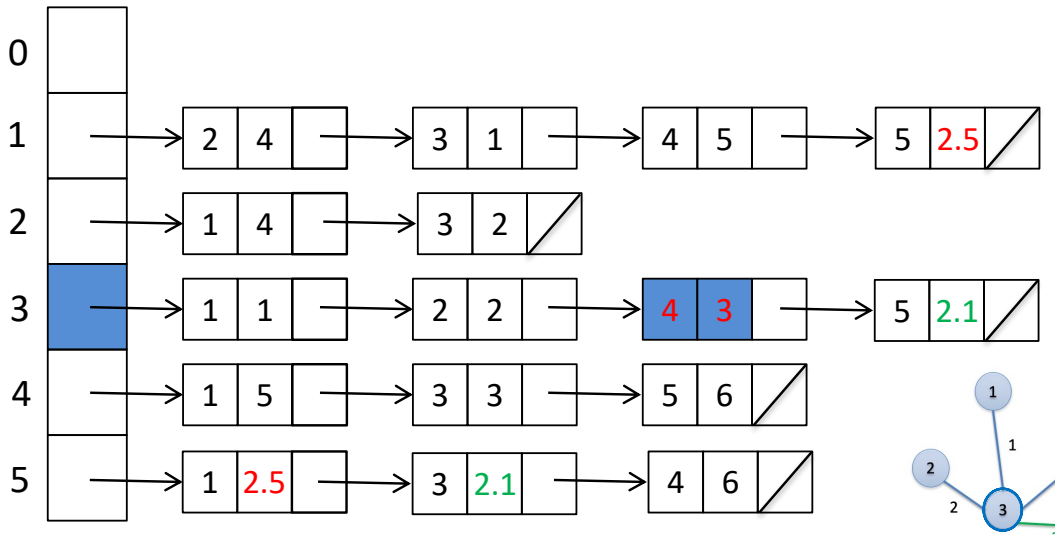
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



	intree	distance	parent
0			
1	T	0	-1
2	F	2	3
3	T	1	1
4	F	5	1
5	F	2.5	1

v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	
	2	2	

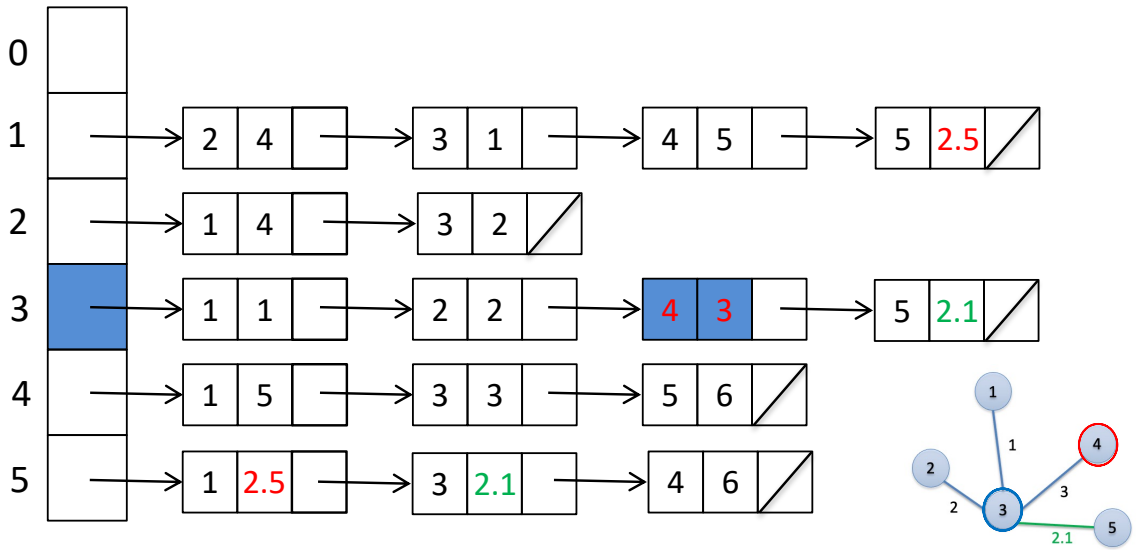
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



	intree	distance	parent
0			
1	T	0	-1
2	F	2	3
3	T	1	1
4	F	5	1
5	F	2.5	1

v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	
	2	2	
	4	3	

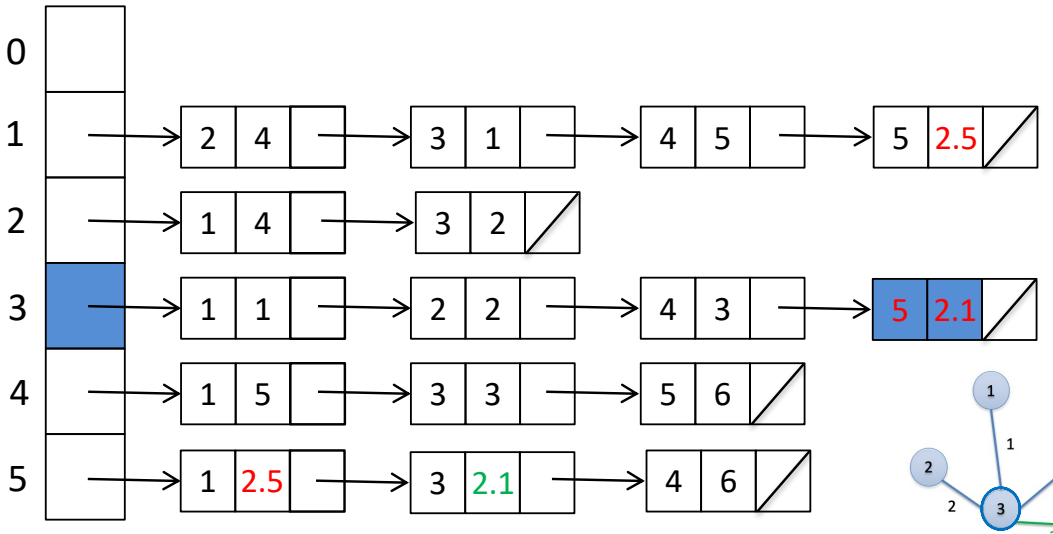
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



	intree	distance	parent
0			
1	T	0	-1
2	F	2	3
3	T	1	1
4	F	3	3
5	F	2.5	1

v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	
	2	2	
	4	3	

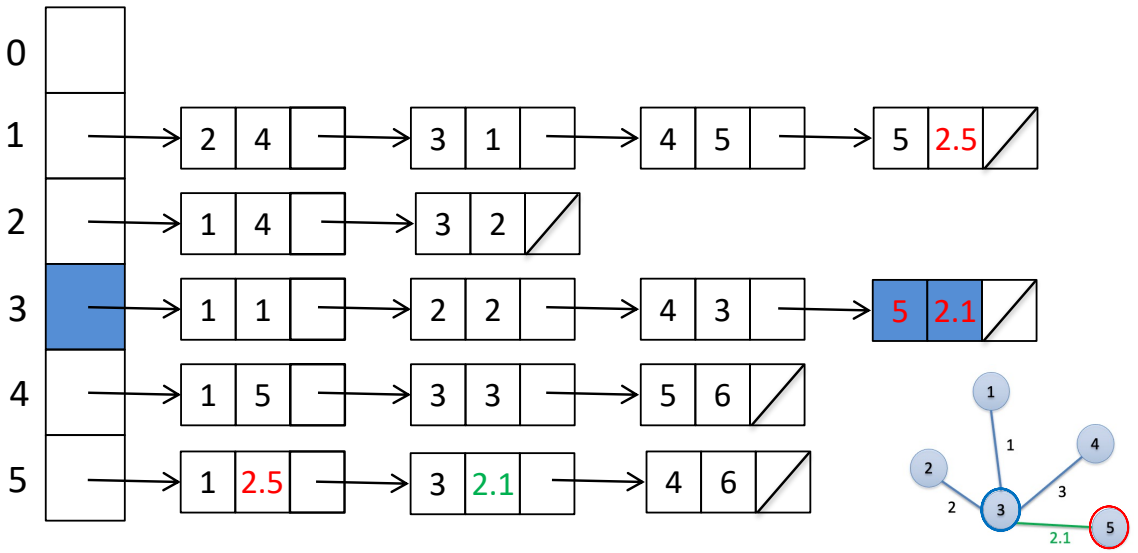
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



	intree	distance	parent
0			
1	T	0	-1
2	F	2	3
3	T	1	1
4	F	5	1
5	F	2.5	1

v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	
	2	2	
	4	3	

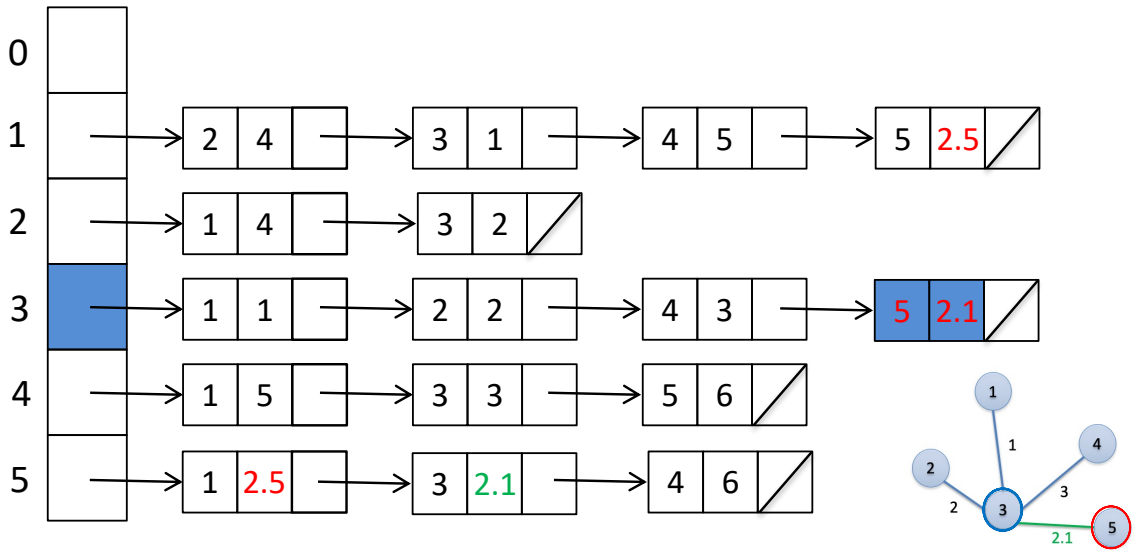
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



	intree	distance	parent
0			
1	T	0	-1
2	F	2	3
3	T	1	1
4	F	5	1
5	F	2.5	1

v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	
	2	2	
	4	3	
	5	2.1	

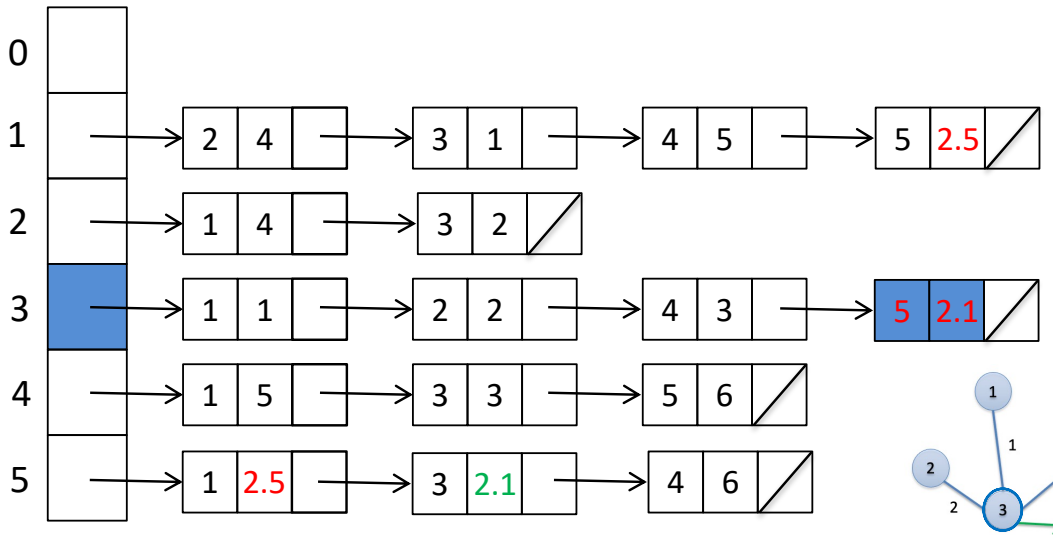
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



	intree	distance	parent
0			
1	T	0	-1
2	F	2	3
3	T	1	1
4	F	3	3
5	F	2.1	3

v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	
	2	2	
	4	3	
	5	2.1	

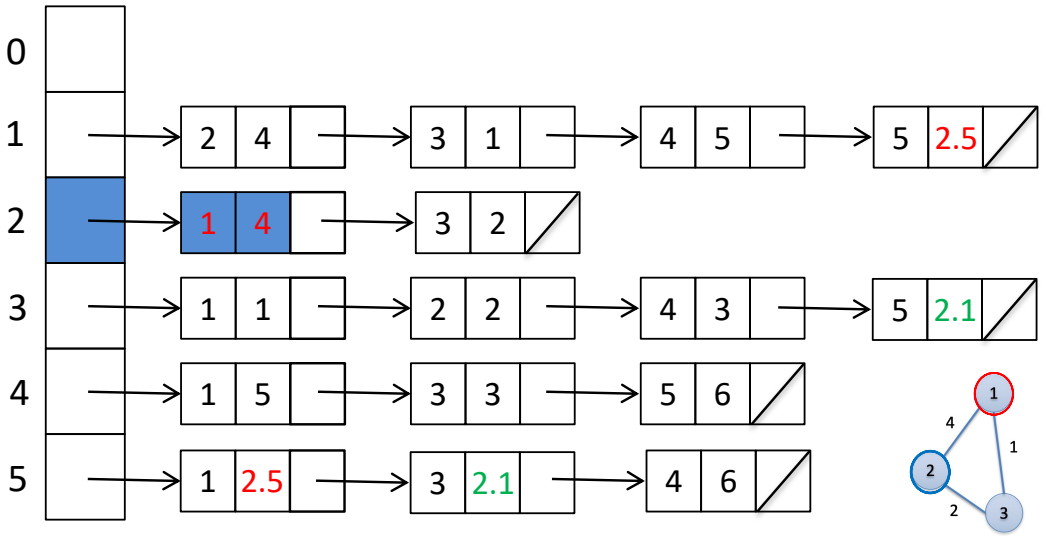
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



	intree	distance	parent
0			
1	T	0	-1
2	F	2	3
3	T	1	1
4	F	3	3
5	F	2.1	3

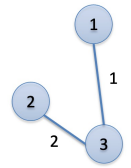
v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	∞
1	2	2	2
2	4	3	
	5	2.1	

```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```

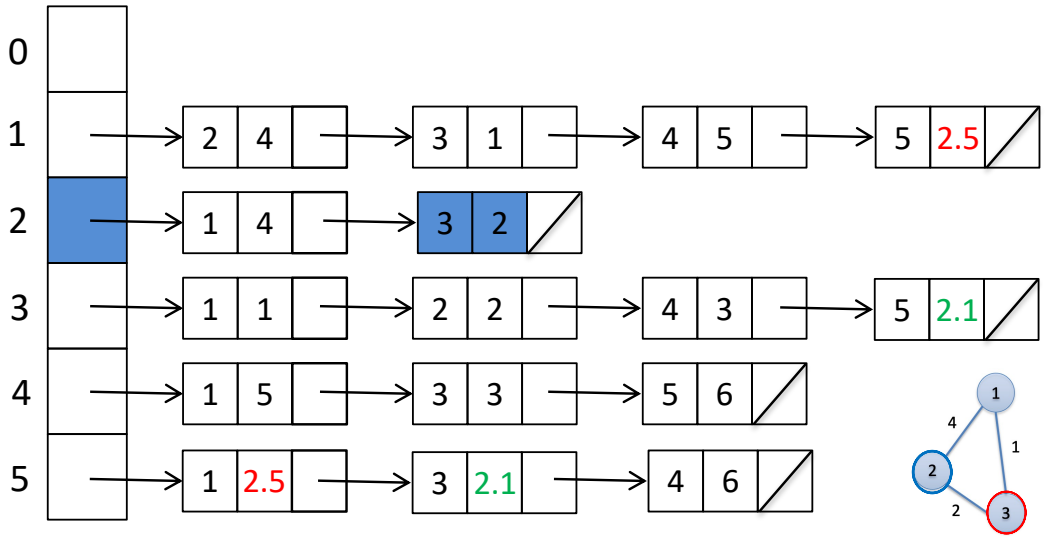


	intree	distance	parent
0			
1	T	0	-1
2	T	2	3
3	T	1	1
4	F	3	3
5	F	2.1	3

v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	∞
1	2	2	2
2	4	3	
	5	2.1	
	1	4	

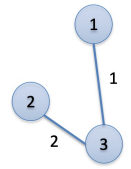


```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```

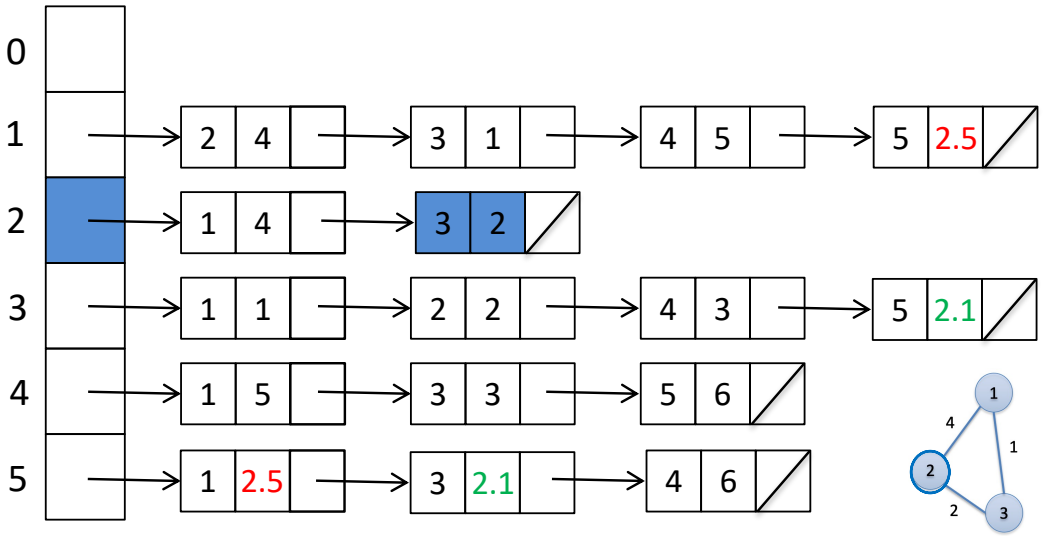


	intree	distance	parent
0			
1	T	0	-1
2	T	2	3
3	T	1	1
4	F	3	3
5	F	2.1	3

v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	∞
1	2	2	2
2	4	3	
	1	4	
	5	2.1	
	3	2	

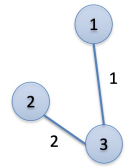


```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```

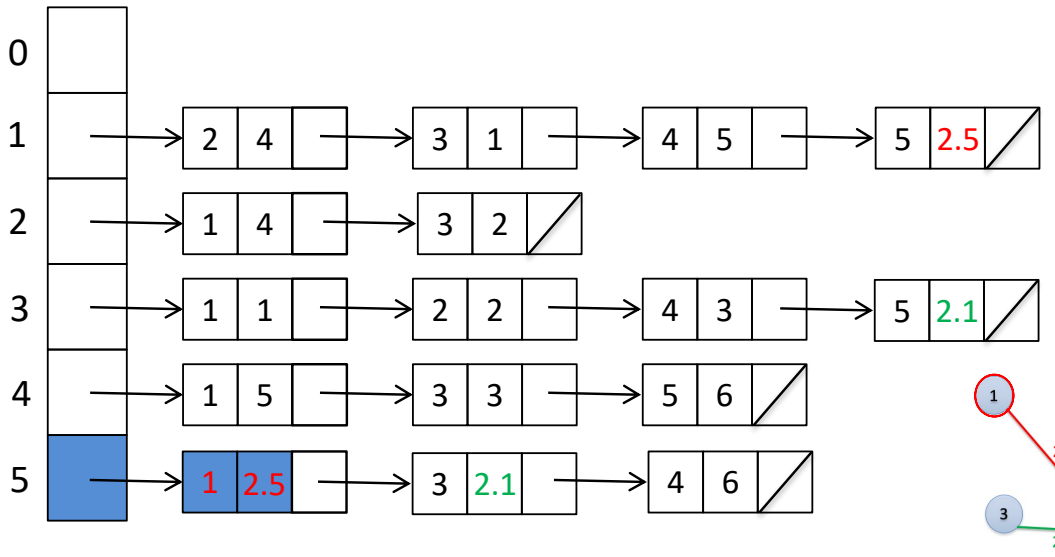


	intree	distance	parent
0			
1	T	0	-1
2	T	2	3
3	T	1	1
4	F	3	3
5	F	2.1	3

v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	∞
1	2	2	2
2	4	3	∞
1	1	4	3
4	5	2.1	2.1
5	3	2	

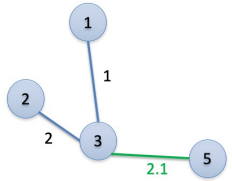


```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```

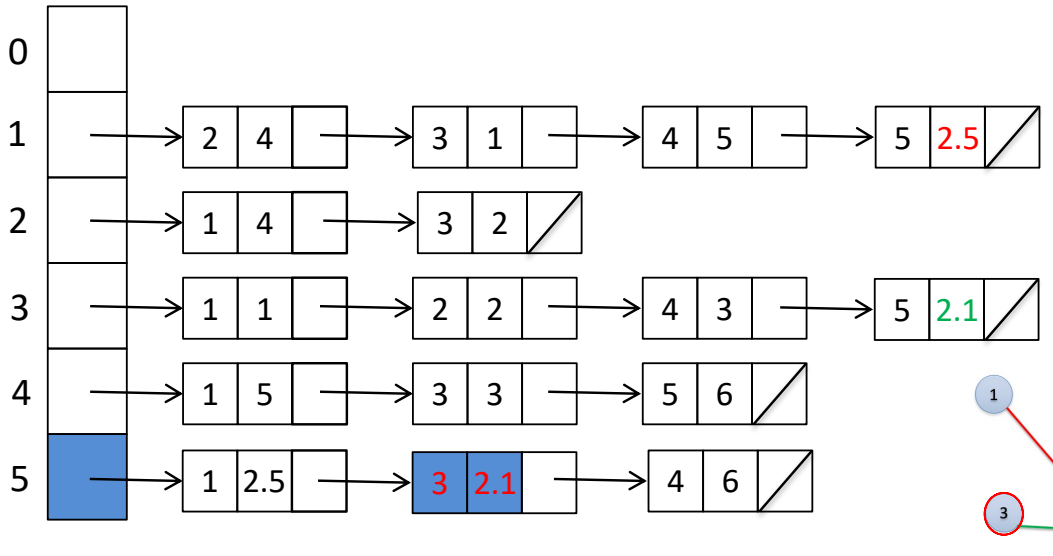


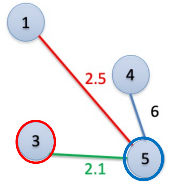
	intree	distance	parent
0			
1	T	0	-1
2	T	2	3
3	T	1	1
4	F	3	3
5	T	2.1	3

v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	∞
1	2	2	2
2	4	3	∞
1	1	4	3
4	5	2.1	2.5
5	3	2	
	1	2.5	



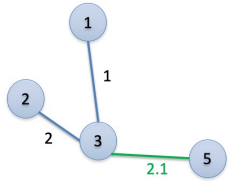
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



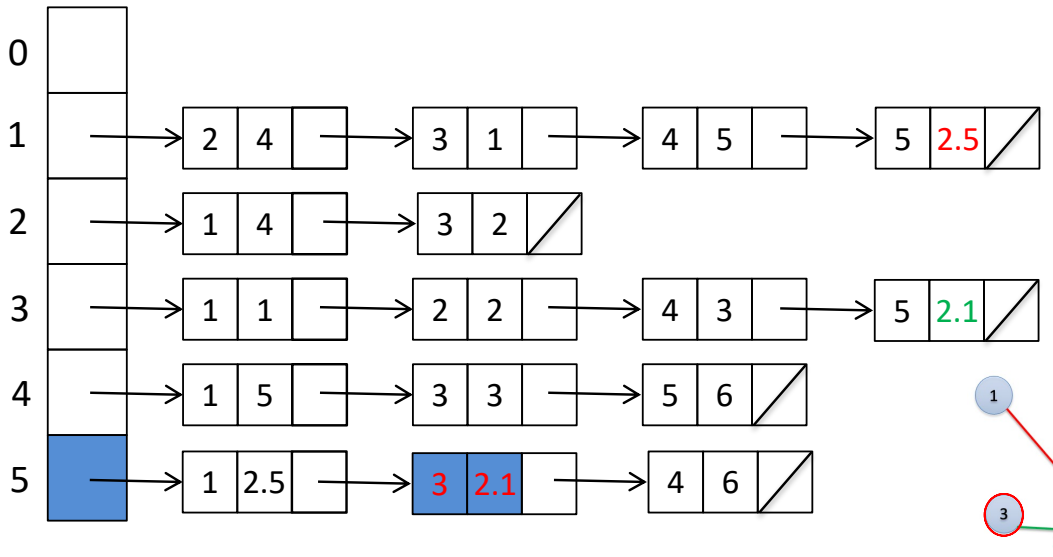


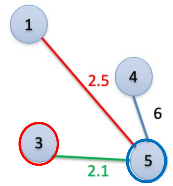
	intree	distance	parent
0			
1	T	0	-1
2	T	2	3
3	T	1	1
4	F	3	3
5	T	2.1	3

v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	∞
1	2	2	2
2	4	3	∞
1	1	4	3
4	5	2.1	2.5
5	3	2	
	1	2.5	



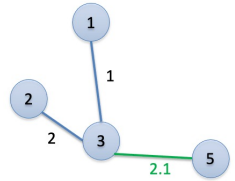
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



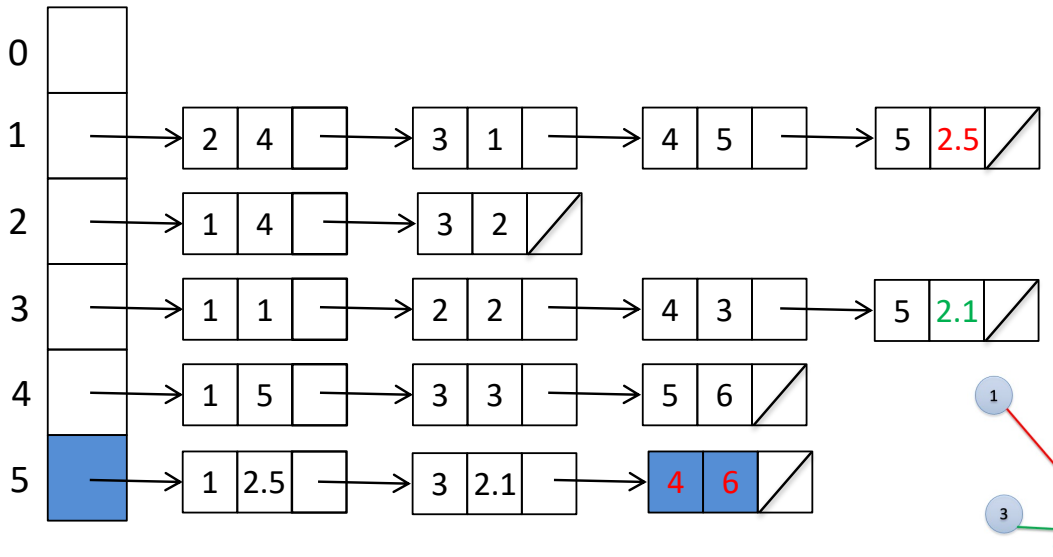


	intree	distance	parent
0			
1	T	0	-1
2	T	2	3
3	T	1	1
4	F	3	3
5	T	2.1	3

v	w	weight	dist
1	2	4	∞
1	3	1	4
2	4	5	1
3	1	1	∞
1	2	2	2
2	4	3	∞
1	1	4	3
4	5	2.1	2.5
5	3	2	
	1	2.5	
3		2.1	

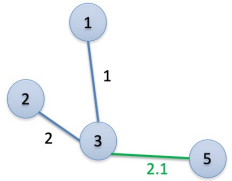


```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```

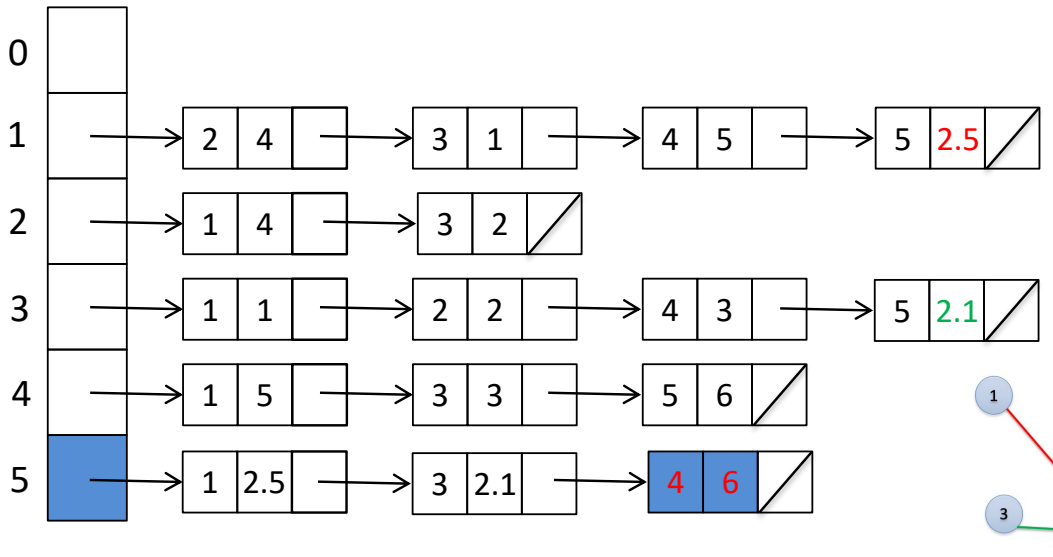


	intree	distance	parent
0			
1	T	0	-1
2	T	2	3
3	T	1	1
4	F	3	3
5	T	2.1	3

v	w	weight	dist
1	...	...	∞
1	3	1	4
2	4	5	1
3	1	1	∞
1	2	2	2
2	4	3	∞
1	1	4	3
4	5	2.1	2.5
5	3	2	
	1	2.5	
	3	2.1	

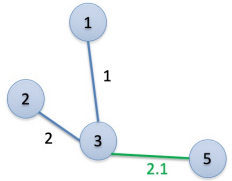


```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```

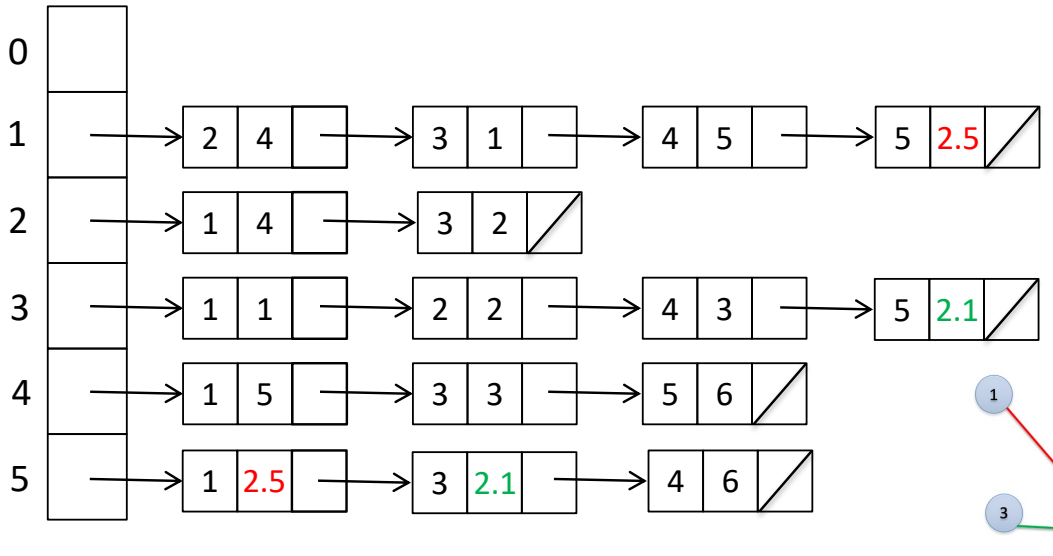


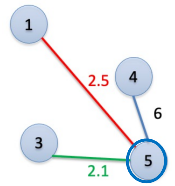
	intree	distance	parent
0			
1	T	0	-1
2	T	2	3
3	T	1	1
4	F	3	3
5	T	2.1	3

v	w	weight	dist
1	...	...	∞
1	4	6	4
2			1
3			∞
1			2
2			∞
1			3
4			2.5
5			



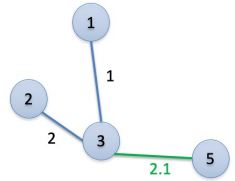
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



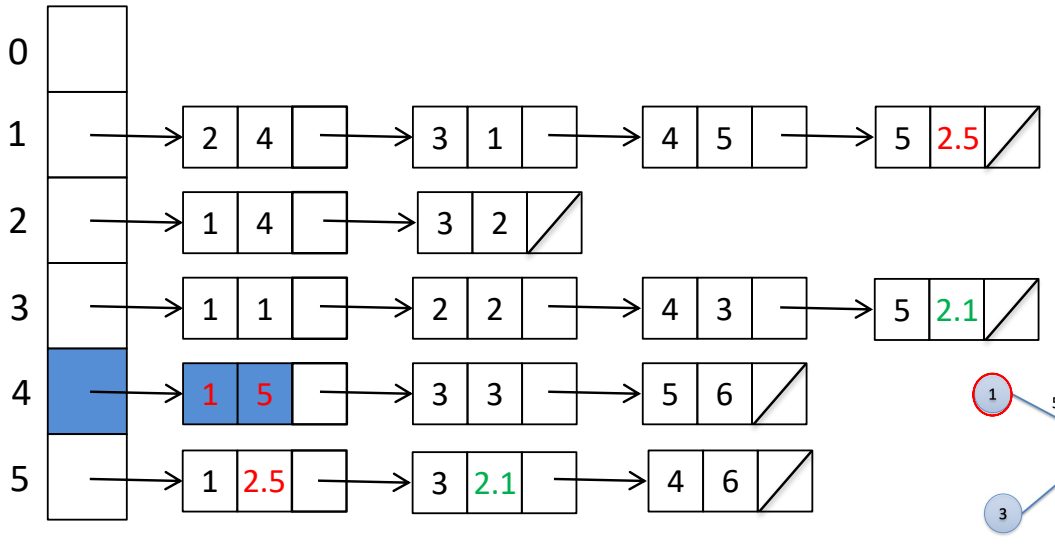


	intree	distance	parent
0			
1	T	0	-1
2	T	2	3
3	T	1	1
4	F	3	3
5	T	2.1	3

v	w	weight	dist
1	...	...	∞
1	4	6	4
2			1
3			∞
1			2
2			∞
1			3
4			2.5
5			∞
1			3
4			

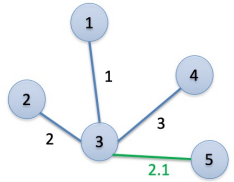


```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```

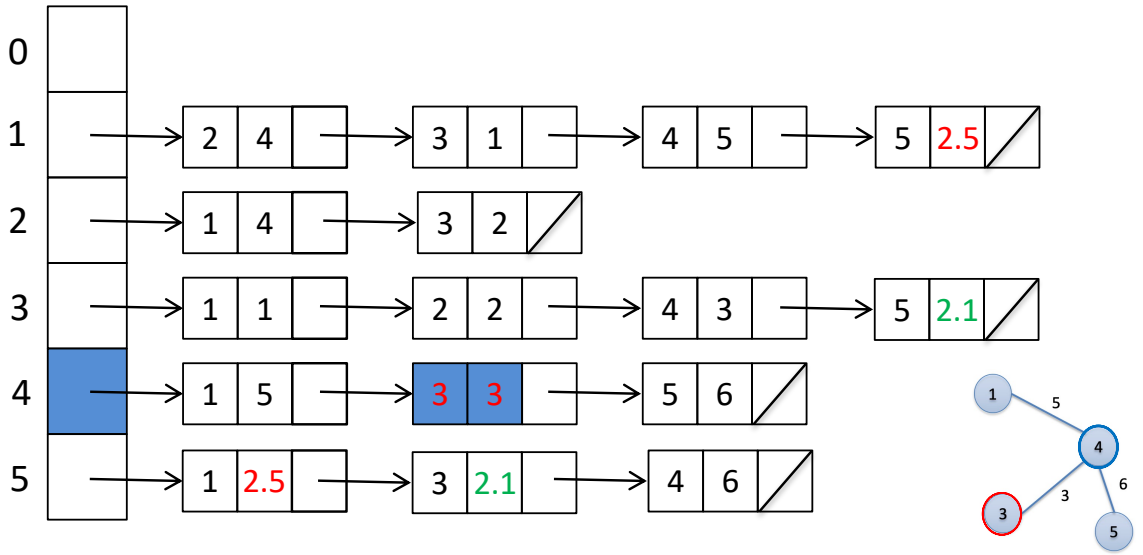


	intree	distance	parent
0			
1	T	0	-1
2	T	2	3
3	T	1	1
4	T	3	3
5	T	2.1	3

v	w	weight	dist
...	...	...	...
5	4	6	
	1	5	

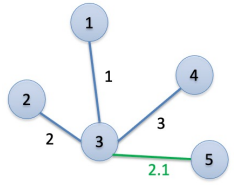


```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```

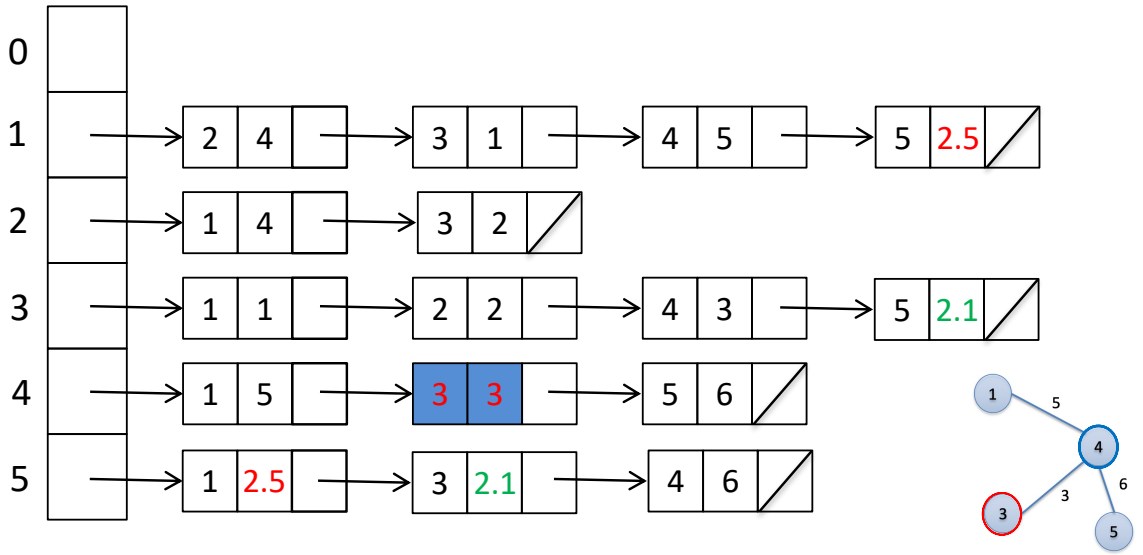


	intree	distance	parent
0			
1	T	0	-1
2	T	2	3
3	T	1	1
4	T	3	3
5	T	2.1	3

v	w	weight	dist
...	...	...	...
5	4	6	
	1	5	

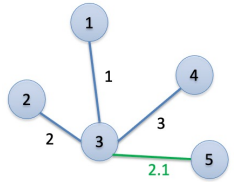


```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```

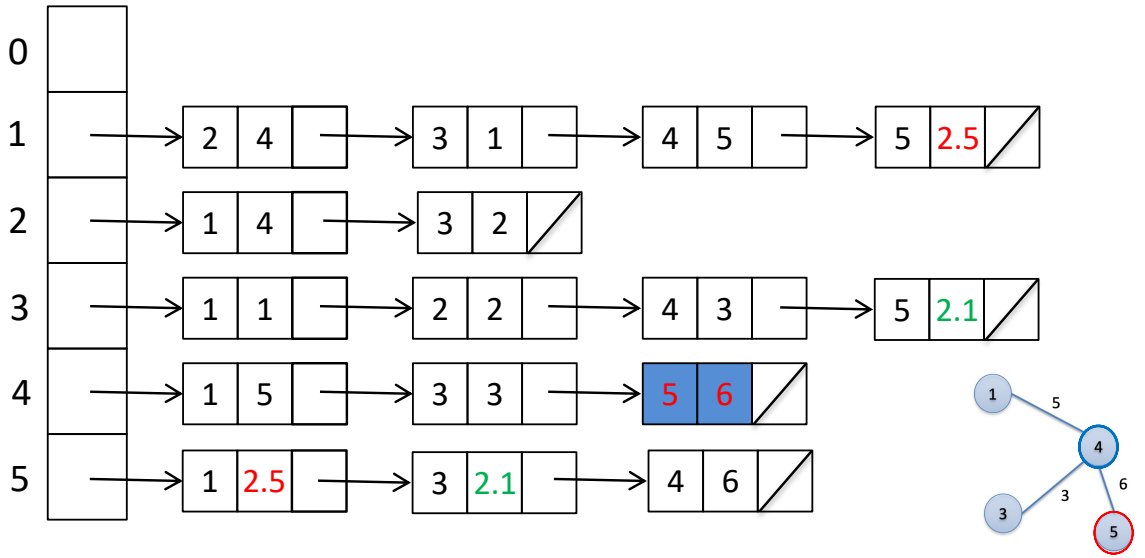


	intree	distance	parent
0			
1	T	0	-1
2	T	2	3
3	T	1	1
4	T	3	3
5	T	2.1	3

v	w	weight	dist
...	...	...	...
5	4	6	
	1	5	
	3	3	

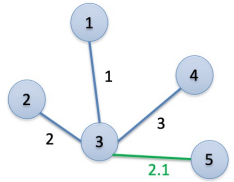


```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```

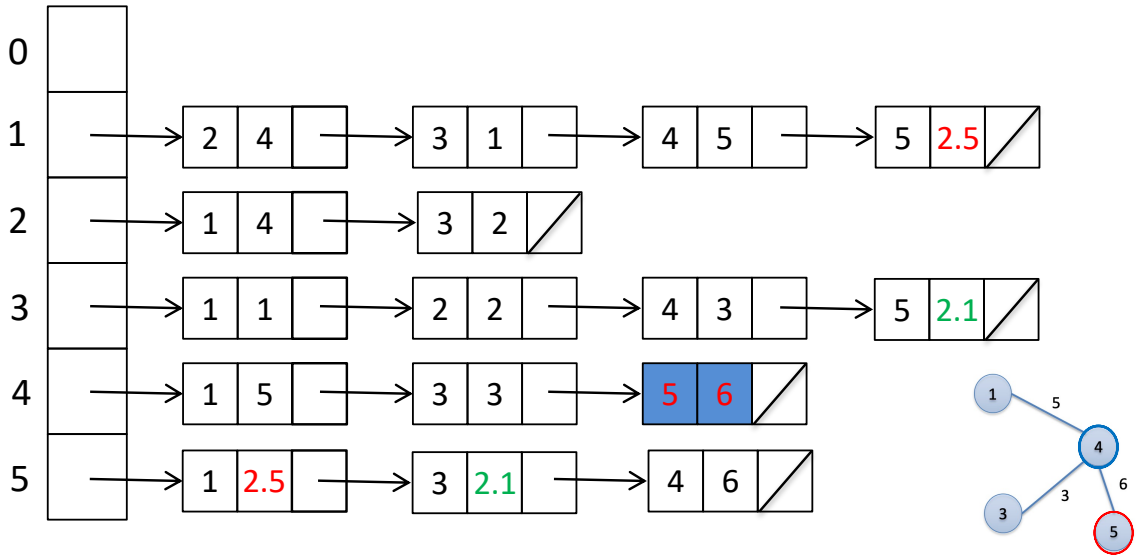


	intree	distance	parent
0			
1	T	0	-1
2	T	2	3
3	T	1	1
4	T	3	3
5	T	2.1	3

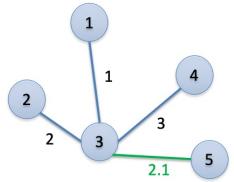
v	w	weight	dist
...	...	...	...
5	4	6	
	1	5	
	3	3	



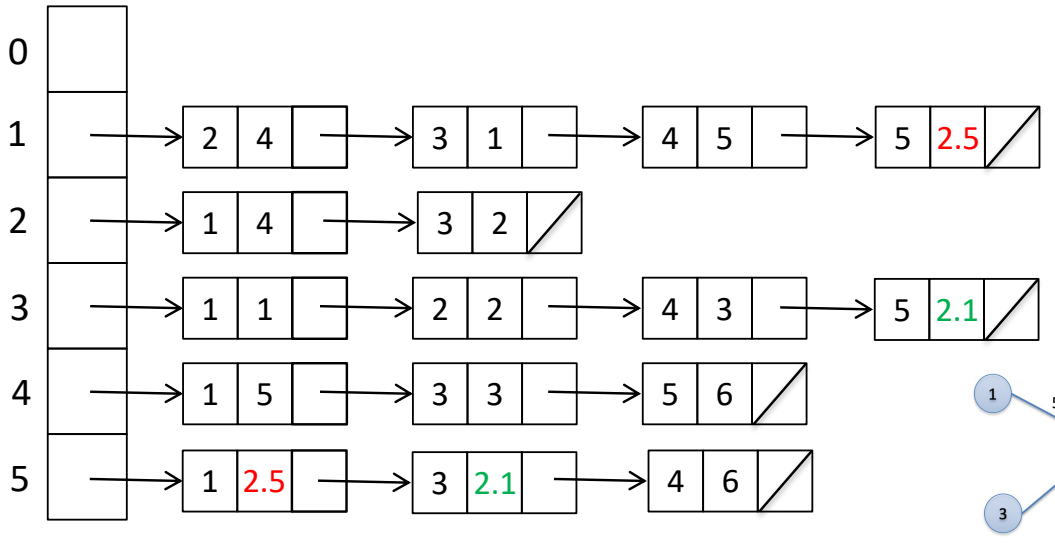
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



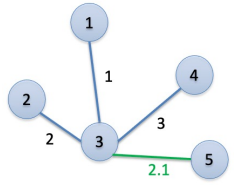
	intree	distance	parent	v	w	weight	dist
0				...	...	...	...
1	T	0	-1	5	4	6	
2	T	2	3		1	5	
3	T	1	1		3	3	
4	T	3	3		5	6	
5	T	2.1	3				



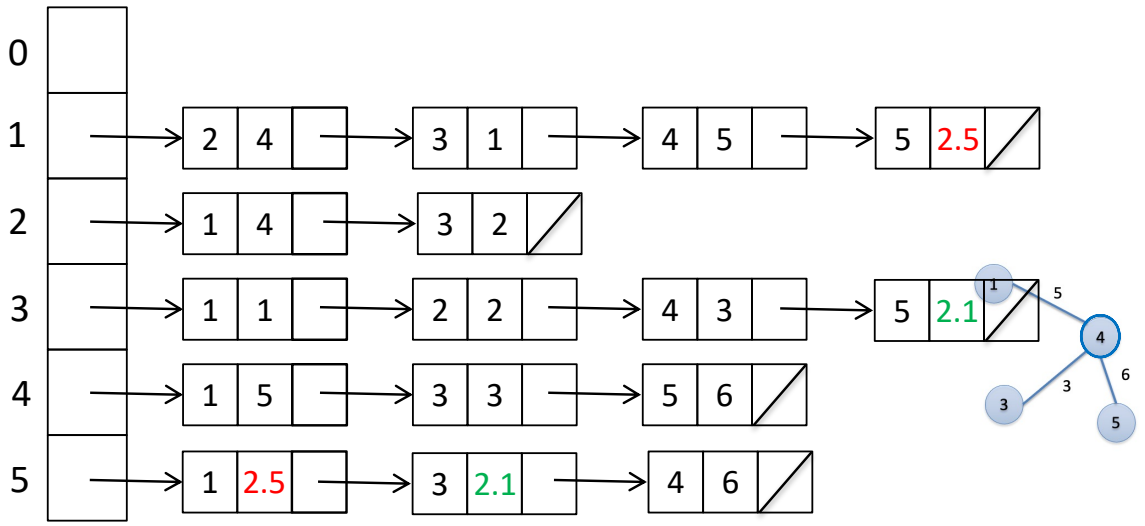
```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



	intree	distance	parent	v	w	weight	dist
0				...	...	...	...
1	T	0	-1	5	4	6	∞
2	T	2	3	1	1	5	
3	T	1	1		3	3	
4	T	3	3		5	6	
5	T	2.1	3				

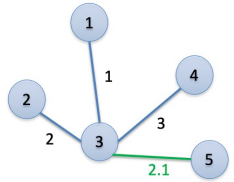


```
for (i=1; i<=g->nvertices; i++) {
   intree[i] = FALSE;
   distance[i] = MAXINT;
   parent[i] = -1;
}

distance[start] = 0;
v = start;

while (intree[v] == FALSE) {

   intree[v] = TRUE;
   p = g->edges[v];

   while (p != NULL) {
      w = p->y;
      weight = p->weight;
      if ((weight < distance[w]) &&
          (intree[w] == FALSE)) {
         distance[w] = weight;
         parent[w] = v;
      }
      p = p->next;
   }

   v = 1;
   dist = MAXINT;
   for (i=1; i<=g->nvertices; i++)
      if ((intree[i] == FALSE) &&
          (distance[i] < dist)) {
         dist = distance[i];
         v = i;
      }
}
```



	intree	distance	parent
0			
1	T	0	-1
2	T	2	3
3	T	1	1
4	T	3	3
5	T	2.1	3

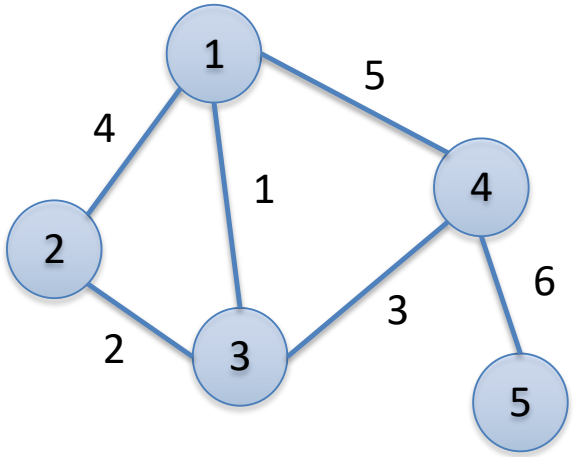
v	w	weight	dist
...	...	...	...
5	4	6	∞
1	1	5	
	3	3	
	5	6	

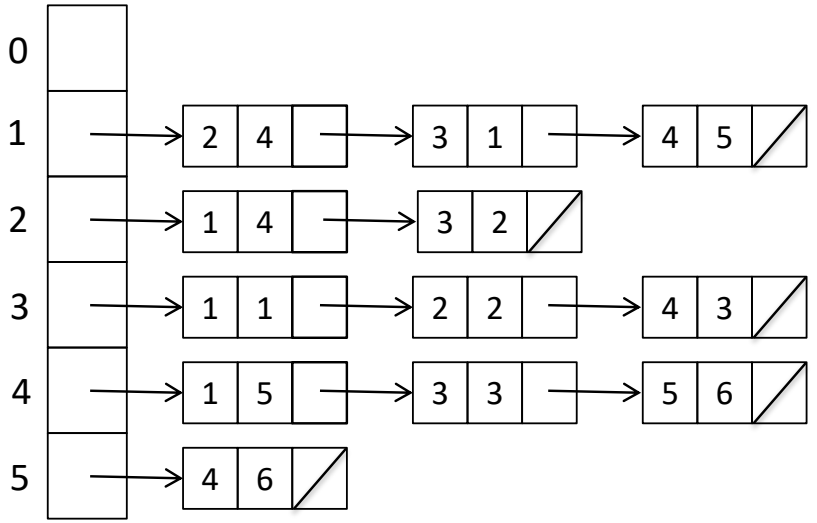


Minimum Spanning Tree

Prim's Algorithm

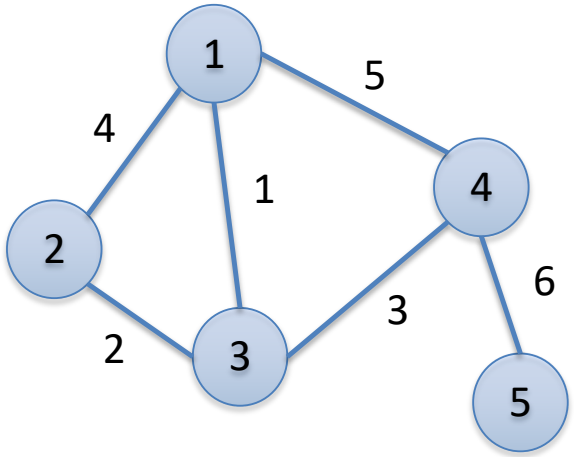
- The minimum spanning tree itself or its cost has to be reconstructed
  - Augment this procedure with statements that print the edges as they are found or totals the weight of all selected edges
  - Alternatively, the tree topology is given by the the `parent` relations and so it can be constructed using the original graph

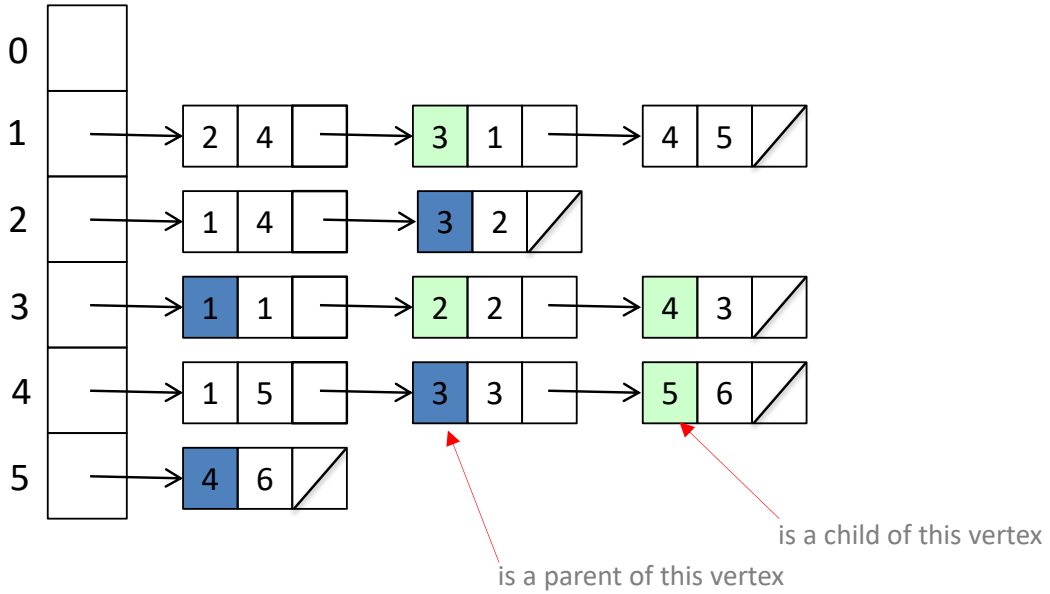




parent

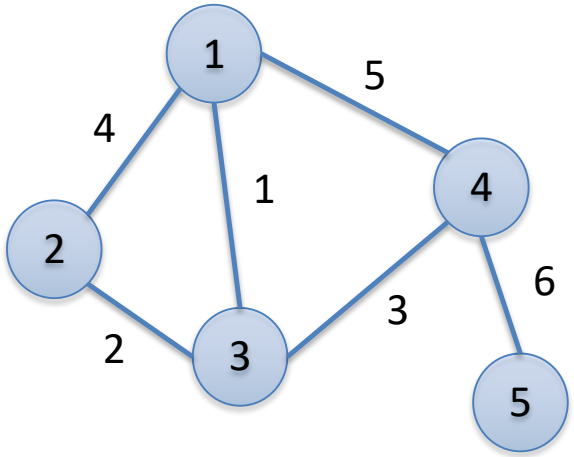
0	
1	-1
2	3
3	1
4	3
5	4

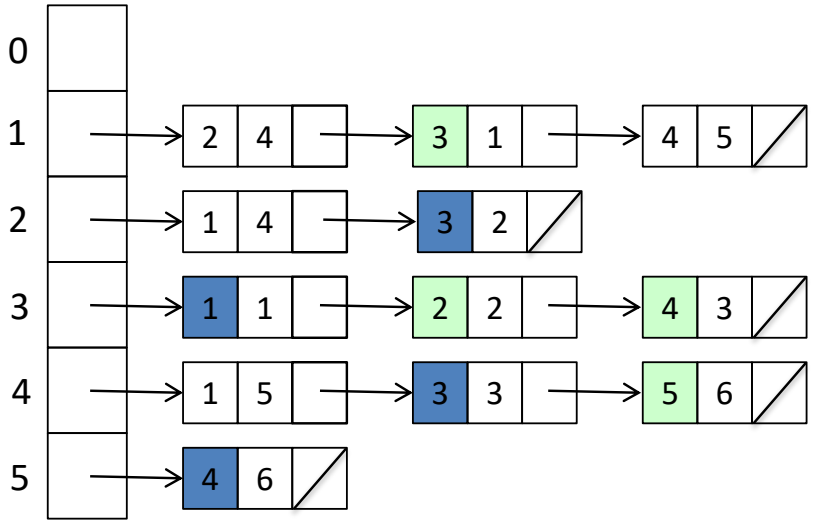




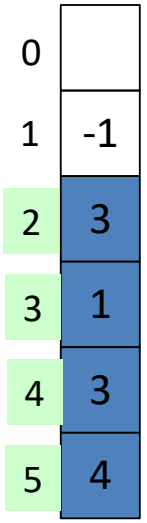
parent

0	
1	-1
2	3
3	1
4	3
5	4


1
2
3
4
5
4
5
1
2
3
6


0
1
2
3
4
5
2 4
3 1
4 5
1 4
3 2
1 1
2 2
4 3
1 5
3 3
5 6
4 6

parent

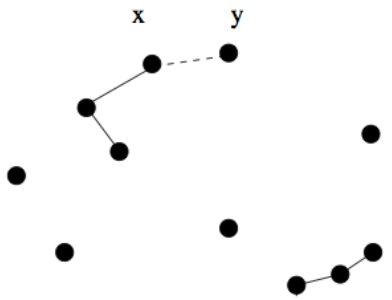

0
1
-1
2
3
3
1
4
3
5
4

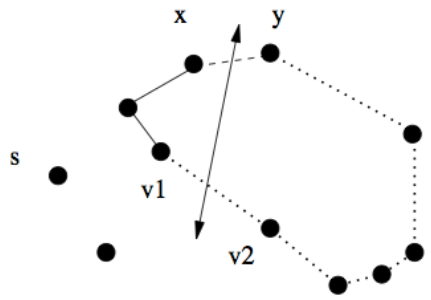
Minimum Spanning Tree

Kruskal's Algorithm

Greedy algorithm:

- Builds up connected components of vertices, culminating in the MST
- Initially each vertex forms its own connected component
- The algorithm repeatedly considers the lightest remaining edge
  - If its two endpoints lie within the same connected component, discard it (because adding it would create a cycle in the tree)
  - Otherwise, insert the edge and merge the two components in to one





Minimum Spanning Tree

Kruskal's Algorithm

```
Kruskal-MST(G):  // note: no start vertex s; cf. Prim's Algorithm

Put the edges in a priority queue ordered by weight
count = 0
While (count < n-1) do
    get next edge (v, w)
    if (component (v) != component (w))
        add to T_Kruskal
        merge component (v) and component (w)
```

Minimum Spanning Tree

Prim's Algorithm vs. Kruskal's Algorithm

- Prim $O(mn)$ … if we don't keep track of lightest edge
- Prim $O(n^2)$ … if we do but use a simple data structure
- Prim $O(m + n \log n)$ … if we use a priority queue data structure
- Kruskal $O(m \log m)$

n vertices, m edges

- Prim's algorithm is faster on dense graphs (depends on n)
- Kruskal's algorithm is faster on sparse graphs (depends on m)