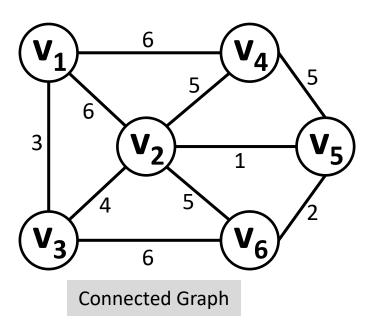
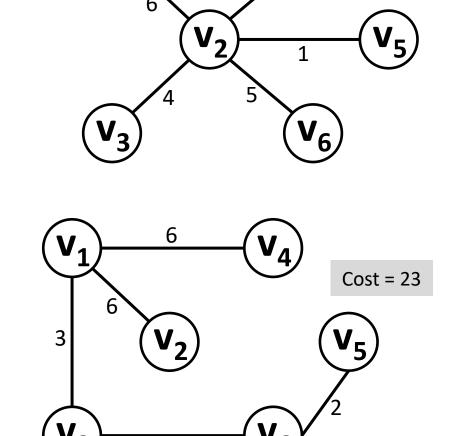
# Graphs

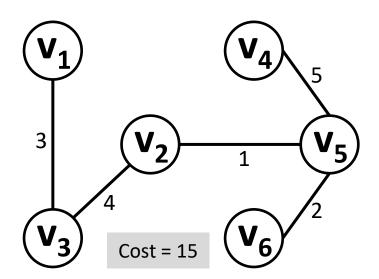
Minimum Spanning Tree





6

Cost = 21



#### Introduction

- Let, G = (V,E) be a connected, undirected graph and w(u,v) be a weight/cost of an edge  $(u,v) \in E$ .
- Then an acyclic subset  $T \subseteq E$ , that connects all of the vertices in V, is called a "spanning tree".
- The problem termed as "minimum spanning tree" or "minimum-weight spanning tree" aims to minimize total weight given as

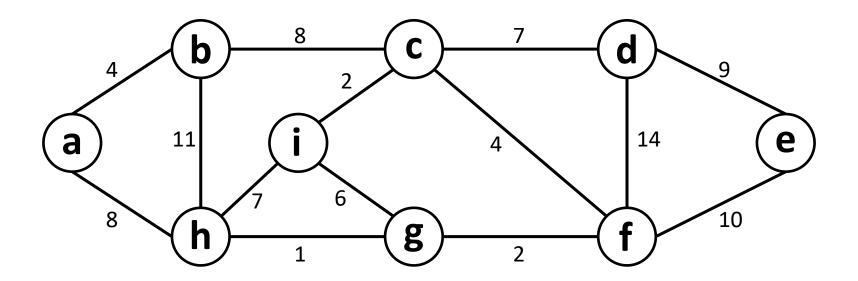
$$w(T) = \sum_{(u,v)\in T} w(u,v)$$

Algorithms to solve the minimum spanning tree problem:

Kruskal's algorithm and

– Prim's algorithm.

# Kruskal's Algorithm

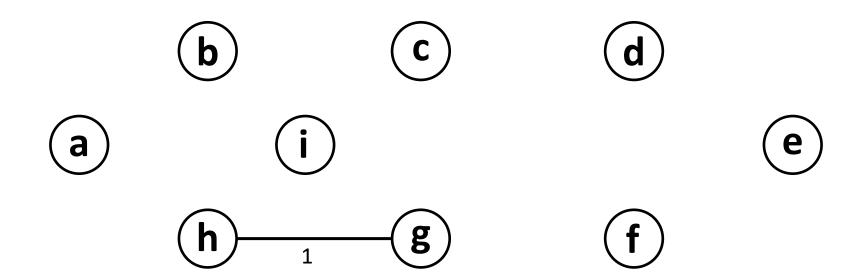


b
c
d
i
g
f
Edge Weight
Edge Weight
Edge Weight

Edge	Weight
(g,h)	1
(f,g)	2
(c,i)	2
(a,b)	4
(c,f)	4

Edge	Weight
(g,i)	6
(c,d)	7
(h,i)	7
(a,h)	8
(b,c)	8

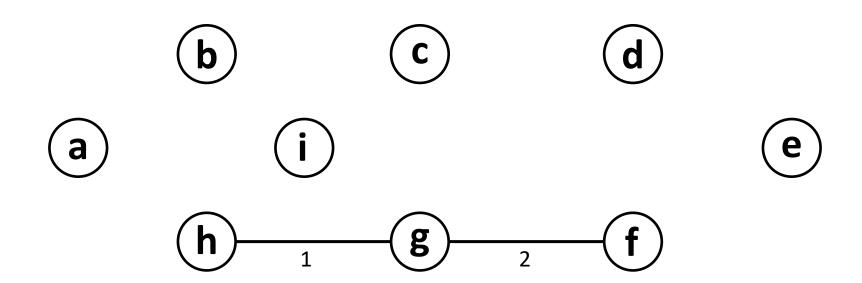
Edge	Weight
(d,e)	9
(e,f)	10
(b,h)	11
(d,f)	14



Edge	Weight
(g,h)	1
(f,g)	2
(c,i)	2
(a,b)	4
(c,f)	4

Edge	Weight
(g,i)	6
(c,d)	7
(h,i)	7
(a,h)	8
(b,c)	8

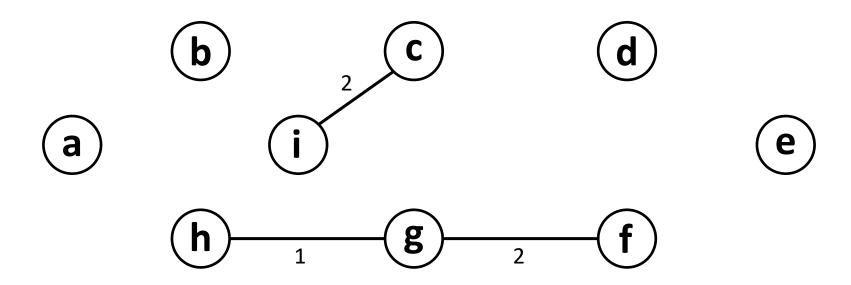
Edge	Weight
(d,e)	9
(e,f)	10
(b,h)	11
(d,f)	14



Edge	Weight
(g,h)	1
(f,g)	2
(c,i)	2
(a,b)	4
(c,f)	4

Edge	Weight
(g,i)	6
(c,d)	7
(h,i)	7
(a,h)	8
(b,c)	8

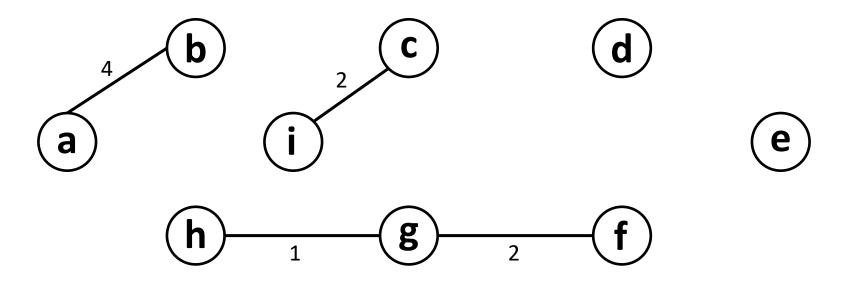
Edge	Weight
(d,e)	9
(e,f)	10
(b,h)	11
(d,f)	14



Edge	Weight
(g,h)	1
(f,g)	2
(c,i)	2
(a,b)	4
(c,f)	4

Edge	Weight
(g,i)	6
(c,d)	7
(h,i)	7
(a,h)	8
(b,c)	8

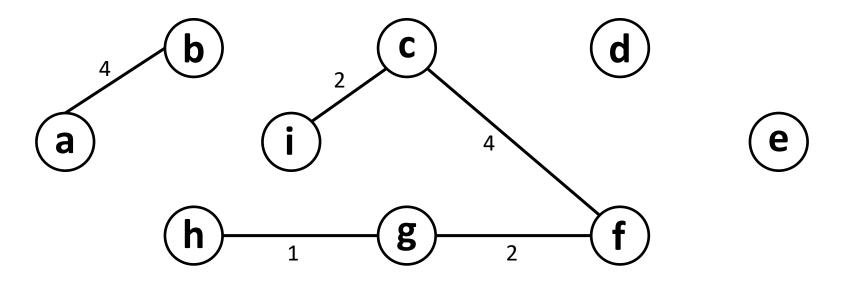
Edge	Weight
(d,e)	9
(e,f)	10
(b,h)	11
(d,f)	14



Edge	Weight
(g,h)	1
(f,g)	2
(c,i)	2
(a,b)	4
(c,f)	4

Edge	Weight
(g,i)	6
(c,d)	7
(h,i)	7
(a,h)	8
(b,c)	8

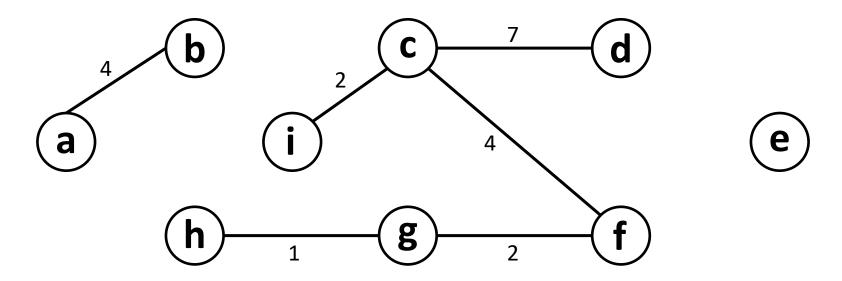
Edge	Weight
(d,e)	9
(e,f)	10
(b,h)	11
(d,f)	14



Edge	Weight
(g,h)	1
(f,g)	2
(c,i)	2
(a,b)	4
(c,f)	4

Edge	Weight
(g,i)	6
(c,d)	7
(h,i)	7
(a,h)	8
(b,c)	8

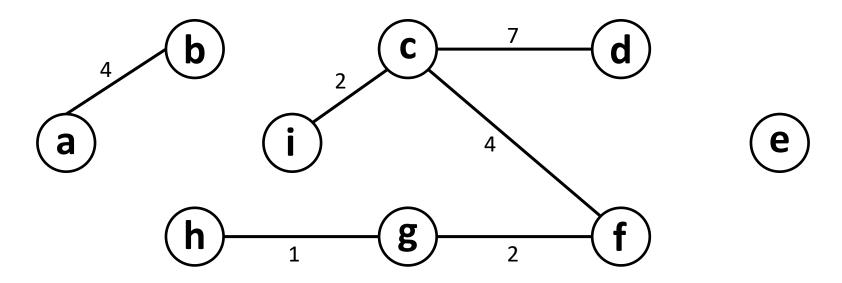
Edge	Weight
(d,e)	9
(e,f)	10
(b,h)	11
(d,f)	14



Edge	Weight
(g,h)	1
(f,g)	2
(c,i)	2
(a,b)	4
(c,f)	4

Edge	Weight
(g,i)	6
(c,d)	7
(h,i)	7
(a,h)	8
(b,c)	8

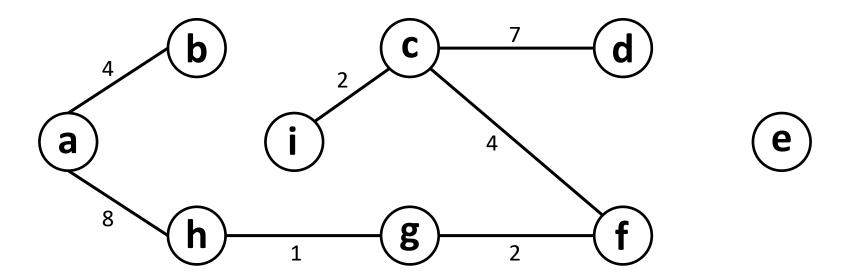
Edge	Weight
(d,e)	9
(e,f)	10
(b,h)	11
(d,f)	14



Edge	Weight
(g,h)	1
(f,g)	2
(c,i)	2
(a,b)	4
(c,f)	4

Edge	Weight
(g,i)	6
(c,d)	7
(h,i)	7
(a,h)	8
(b,c)	8

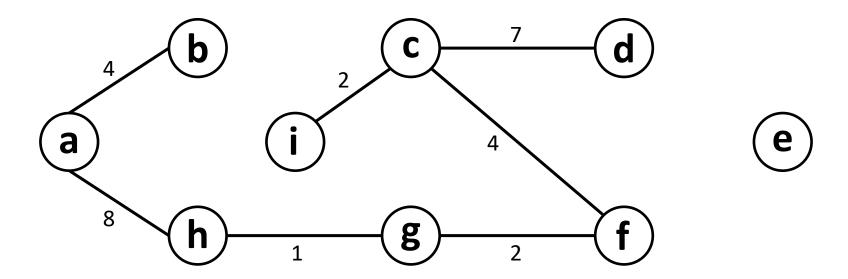
Edge	Weight
(d,e)	9
(e,f)	10
(b,h)	11
(d,f)	14



Edge	Weight
(g,h)	1
(f,g)	2
(c,i)	2
(a,b)	4
(c,f)	4

Edge	Weight
(g,i)	6
(c,d)	7
(h,i)	7
(a,h)	8
(b,c)	8

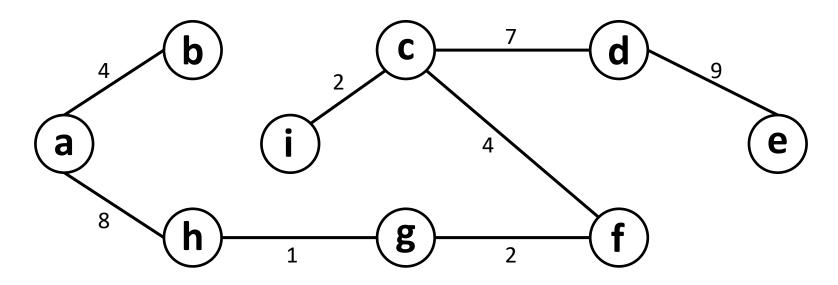
Edge	Weight
(d,e)	9
(e,f)	10
(b,h)	11
(d,f)	14



Edge	Weight
(g,h)	1
(f,g)	2
(c,i)	2
(a,b)	4
(c,f)	4

Edge	Weight
(g,i)	6
(c,d)	7
(h,i)	7
(a,h)	8
(b,c)	8

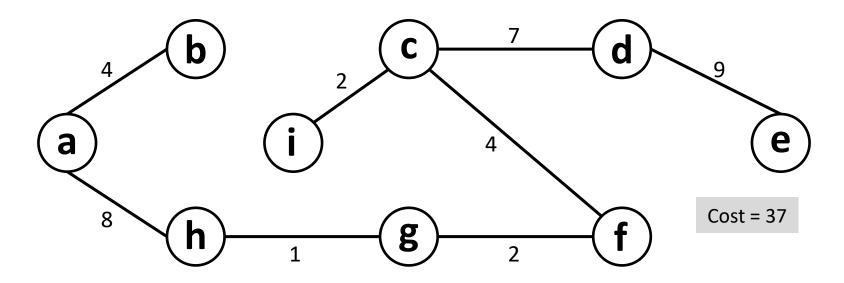
Edge	Weight
(d,e)	9
(e,f)	10
(b,h)	11
(d,f)	14



Edge	Weight
(g,h)	1
(f,g)	2
(c,i)	2
(a,b)	4
(c,f)	4

Edge	Weight
(g,i)	6
(c,d)	7
(h,i)	7
(a,h)	8
(b,c)	8

Edge	Weight
(d,e)	9
(e,f)	10
(b,h)	11
(d,f)	14



Edge	Weight
(g,h)	1
(f,g)	2
(c,i)	2
(a,b)	4
(c,f)	4

Edge	Weight
(g,i)	6
(c,d)	7
(h,i)	7
(a,h)	8
(b,c)	8

Edge	Weight
(d,e)	9
(e,f)	10
(b,h)	11
(d,f)	14

### Implementation MAKE-SET(x)

```
MAKE-SET(x)
1 \quad x.p = x
2 \quad x.rank = 0
```

```
MST-KRUSKAL(G, w)

1 A = \emptyset

2 for each vertex v \in G.V

3 MAKE-SET(v)

4 sort the edges of G.E into nondecreasing order by weight w

5 for each edge (u, v) \in G.E, taken in nondecreasing order by weight

6 if FIND-SET(u) \neq FIND-SET(v)

7 A = A \cup \{(u, v)\}

UNION(u, v)

9 return A
```

```
FIND-SET(x)

1 if x \neq x.p

2 x.p = \text{FIND-SET}(x.p)

3 return x.p
```

```
UNION(x, y)
1 LINK(FIND-SET(x), FIND-SET(y))
```

```
LINK(x, y)

1 if x.rank > y.rank

2 y.p = x

3 else x.p = y

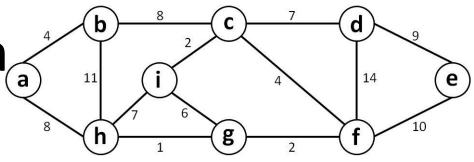
4 if x.rank == y.rank

5 y.rank = y.rank + 1
```

Example - Execution

Vertex	р	rank
а	a	0
b	b	0
С	С	0
d	d	0
е	е	0
f	f	0
g	g	0
h	h	0
i	i	0

$$A = \{\}$$



```
MST-KRUSKAL(G, w)

1 A = \emptyset

2 for each vertex v \in G.V

3 MAKE-SET(v)

4 sort the edges of G.E into nondecreasing order by weight w

5 for each edge (u, v) \in G.E, taken in nondecreasing order by weight

6 if FIND-SET(u) \neq FIND-SET(v)

7 A = A \cup \{(u, v)\}

UNION(u, v)

9 return A
```

Edge	Weight
(g,h)	1
(f,g)	2
(c,i)	2
(a,b)	4
(c,f)	4

Edge	Weight
(g,i)	6
(c,d)	7
(h,i)	7
(a,h)	8
(b,c)	8

Edge	Weight
(d,e)	9
(e,f)	10
(b,h)	11
(d,f)	14

Vertex	р	rank
а	a	0
b	b	0
С	С	0
d	d	0
е	е	0
f	f	0
g	g	0
h	h	0
i	i	0

$$A = \{\}$$

```
FIND-SET(x)

1 if x \neq x.p

2 x.p = \text{FIND-SET}(x.p)

3 return x.p
```

```
LINK(x, y)

1 if x.rank > y.rank

2 y.p = x

3 else x.p = y

4 if x.rank == y.rank

5 y.rank = y.rank + 1
```

```
MST-KRUSKAL(G, w)

1 A = \emptyset

2 for each vertex v \in G.V

3 MAKE-SET(v)

4 sort the edges of G.E into nondecreasing order by weight w

5 for each edge (u, v) \in G.E, taken in nondecreasing order by weight

6 if FIND-SET(u) \neq FIND-SET(v)

7 A = A \cup \{(u, v)\}

UNION(u, v)

9 return A
```

Edge	Weight
(g,h)	1
(f,g)	2
(c,i)	2
(a,b)	4
(c,f)	4

Edge	Weight
(g,i)	6
(c,d)	7
(h,i)	7
(a,h)	8
(b,c)	8

Edge	Weight
(d,e)	9
(e,f)	10
(b,h)	11
(d,f)	14

Vertex	р	rank
а	a	0
b	b	0
С	С	0
d	d	0
е	е	0
f	f	0
g	h	0
h	h	1
i	i	0

$$A = \{(g,h)\}$$

```
FIND-SET(x)

1 if x \neq x.p

2 x.p = \text{FIND-SET}(x.p)

3 return x.p
```

```
LINK(x, y)

1 if x.rank > y.rank

2 y.p = x

3 else x.p = y

4 if x.rank == y.rank

5 y.rank = y.rank + 1
```

```
MST-KRUSKAL(G, w)

1 A = \emptyset

2 for each vertex v \in G.V

3 MAKE-SET(v)

4 sort the edges of G.E into nondecreasing order by weight w

5 for each edge (u, v) \in G.E, taken in nondecreasing order by weight

6 if FIND-SET(u) \neq FIND-SET(v)

7 A = A \cup \{(u, v)\}

UNION(u, v)

9 return A
```

Edge	Weight
(g,h)	1
(f,g)	2
(c,i)	2
(a,b)	4
(c,f)	4

	Edge	Weight
	(g,i)	6
	(c,d)	7
Ī	(h,i)	7
	(a,h)	8
	(b,c)	8

Edge	Weight
(d,e)	9
(e,f)	10
(b,h)	11
(d,f)	14

Vertex	р	rank
а	а	0
b	b	0
С	С	0
d	d	0
е	е	0
f	f	0
g	h	0
h	h	1
i	i	0

$$A = \{(g,h)\}$$

```
FIND-SET(x)

1 if x \neq x.p

2 x.p = \text{FIND-SET}(x.p)

3 return x.p
```

```
LINK(x, y)

1 if x.rank > y.rank

2 y.p = x

3 else x.p = y

4 if x.rank == y.rank

5 y.rank = y.rank + 1
```

```
MST-KRUSKAL(G, w)

1 A = \emptyset

2 for each vertex v \in G.V

3 MAKE-SET(v)

4 sort the edges of G.E into nondecreasing order by weight w

5 for each edge (u, v) \in G.E, taken in nondecreasing order by weight

6 if FIND-SET(u) \neq FIND-SET(v)

7 A = A \cup \{(u, v)\}

UNION(u, v)

9 return A
```

Edge	Weight
(g,h)	1
(f,g)	2
(c,i)	2
(a,b)	4
(c,f)	4

Edge	Weight
(g,i)	6
(c,d)	7
(h,i)	7
(a,h)	8
(b,c)	8

Edge	Weight
(d,e)	9
(e,f)	10
(b,h)	11
(d,f)	14

Vertex	р	rank
а	a	0
b	b	0
С	С	0
d	d	0
е	е	0
f	h	0
g	h	0
h	h	1
i	i	0

$$A = \{(g,h), (f,g)\}$$

```
FIND-SET(x)

1 if x \neq x.p

2 x.p = \text{FIND-SET}(x.p)

3 return x.p
```

```
LINK(x, y)

1 if x.rank > y.rank

2 y.p = x

3 else x.p = y

4 if x.rank == y.rank

5 y.rank = y.rank + 1
```

```
MST-KRUSKAL(G, w)

1 A = \emptyset

2 for each vertex v \in G.V

3 MAKE-SET(v)

4 sort the edges of G.E into nondecreasing order by weight w

5 for each edge (u, v) \in G.E, taken in nondecreasing order by weight

6 if FIND-SET(u) \neq FIND-SET(v)

7 A = A \cup \{(u, v)\}

UNION(u, v)

9 return A
```

Edge	Weight
(g,h)	1
(f,g)	2
(c,i)	2
(a,b)	4
(c,f)	4

Edge	Weight
(g,i)	6
(c,d)	7
(h,i)	7
(a,h)	8
(b,c)	8

Edge	Weight
(d,e)	9
(e,f)	10
(b,h)	11
(d,f)	14

Vertex	р	rank
а	а	0
b	b	0
С	С	0
d	d	0
е	е	0
f	h	0
g	h	0
h	h	1
i	i	0

$$A = \{(g,h), (f,g)\}$$

```
FIND-SET(x)

1 if x \neq x.p

2 x.p = \text{FIND-SET}(x.p)

3 return x.p
```

```
LINK(x, y)

1 if x.rank > y.rank

2 y.p = x

3 else x.p = y

4 if x.rank == y.rank

5 y.rank = y.rank + 1
```

```
MST-KRUSKAL(G, w)

1 A = \emptyset

2 for each vertex v \in G.V

3 MAKE-SET(v)

4 sort the edges of G.E into nondecreasing order by weight w

5 for each edge (u, v) \in G.E, taken in nondecreasing order by weight

6 if FIND-SET(u) \neq FIND-SET(v)

7 A = A \cup \{(u, v)\}

UNION(u, v)

9 return A
```

Edge	Weight
(g,h)	1
(f,g)	2
(c,i)	2
(a,b)	4
(c,f)	4

Edge	Weight
(g,i)	6
(c,d)	7
(h,i)	7
(a,h)	8
(b,c)	8

Edge	Weight
(d,e)	9
(e,f)	10
(b,h)	11
(d,f)	14

Vertex	р	rank
а	a	0
b	b	0
С	i	0
d	d	0
е	е	0
f	h	0
g	h	0
h	h	1
i	i	1

$$A = \{(g,h), (f,g), (c,i)\}$$

```
FIND-SET(x)

1 if x \neq x.p

2 x.p = \text{FIND-SET}(x.p)

3 return x.p
```

```
LINK(x, y)

1 if x.rank > y.rank

2 y.p = x

3 else x.p = y

4 if x.rank == y.rank

5 y.rank = y.rank + 1
```

```
MST-KRUSKAL(G, w)

1 A = \emptyset

2 for each vertex v \in G.V

3 MAKE-SET(v)

4 sort the edges of G.E into nondecreasing order by weight w

5 for each edge (u, v) \in G.E, taken in nondecreasing order by weight

6 if FIND-SET(u) \neq FIND-SET(v)

7 A = A \cup \{(u, v)\}

UNION(u, v)

9 return A
```

Edge	Weight
(g,h)	1
(f,g)	2
(c,i)	2
(a,b)	4
(c,f)	4

Edge	Weight
(g,i)	6
(c,d)	7
(h,i)	7
(a,h)	8
(b,c)	8

Edge	Weight
(d,e)	9
(e,f)	10
(b,h)	11
(d,f)	14

Vertex	р	rank
а	a	0
b	b	0
С	i	0
d	d	0
е	е	0
f	h	0
g	h	0
h	h	1
i	i	1

$$A = \{(g,h), (f,g), (c,i)\}$$

```
FIND-SET(x)

1 if x \neq x.p

2 x.p = \text{FIND-SET}(x.p)

3 return x.p
```

```
LINK(x, y)

1 if x.rank > y.rank

2 y.p = x

3 else x.p = y

4 if x.rank == y.rank

5 y.rank = y.rank + 1
```

```
MST-KRUSKAL(G, w)

1 A = \emptyset

2 for each vertex v \in G.V

3 MAKE-SET(v)

4 sort the edges of G.E into nondecreasing order by weight w

5 for each edge (u, v) \in G.E, taken in nondecreasing order by weight

6 if FIND-SET(u) \neq FIND-SET(v)

7 A = A \cup \{(u, v)\}

UNION(u, v)

9 return A
```

Edge	Weight
(g,h)	1
(f,g)	2
(c,i)	2
(a,b)	4
(c,f)	4

Edge	Weight
(g,i)	6
(c,d)	7
(h,i)	7
(a,h)	8
(b,c)	8

Edge	Weight
(d,e)	9
(e,f)	10
(b,h)	11
(d,f)	14

Vertex	р	rank
а	b	0
b	b	1
С	i	0
d	d	0
е	е	0
f	h	0
g	h	0
h	h	1
i	i	1

```
A = \{(g,h), (f,g), (c,i), (a,b)\}
```

```
FIND-SET(x)

1 if x \neq x.p

2 x.p = \text{FIND-SET}(x.p)

3 return x.p
```

```
LINK(x, y)

1 if x.rank > y.rank

2 y.p = x

3 else x.p = y

4 if x.rank == y.rank

5 y.rank = y.rank + 1
```

```
MST-KRUSKAL(G, w)

1 A = \emptyset

2 for each vertex v \in G.V

3 MAKE-SET(v)

4 sort the edges of G.E into nondecreasing order by weight w

5 for each edge (u, v) \in G.E, taken in nondecreasing order by weight

6 if FIND-SET(u) \neq FIND-SET(v)

7 A = A \cup \{(u, v)\}

UNION(u, v)

9 return A
```

Edge	Weight
(g,h)	1
(f,g)	2
(c,i)	2
(a,b)	4
(c,f)	4

Edge	Weight
(g,i)	6
(c,d)	7
(h,i)	7
(a,h)	8
(b,c)	8

Edge	Weight
(d,e)	9
(e,f)	10
(b,h)	11
(d,f)	14

Vertex	р	rank
а	b	0
b	b	1
С	i	0
d	d	0
е	е	0
f	h	0
g	h	0
h	h	1
i	i	1

```
A = \{(g,h), (f,g), (c,i), (a,b)\}
```

```
FIND-SET(x)

1 if x \neq x.p

2 x.p = \text{FIND-SET}(x.p)

3 return x.p
```

```
LINK(x, y)

1 if x.rank > y.rank

2 y.p = x

3 else x.p = y

4 if x.rank == y.rank

5 y.rank = y.rank + 1
```

```
MST-KRUSKAL(G, w)

1 A = \emptyset

2 for each vertex v \in G.V

3 MAKE-SET(v)

4 sort the edges of G.E into nondecreasing order by weight w

5 for each edge (u, v) \in G.E, taken in nondecreasing order by weight

6 if FIND-SET(u) \neq FIND-SET(v)

7 A = A \cup \{(u, v)\}

UNION(u, v)

9 return A
```

Edge	Weight
(g,h)	1
(f,g)	2
(c,i)	2
(a,b)	4
(c,f)	4

Edge	Weight
(g,i)	6
(c,d)	7
(h,i)	7
(a,h)	8
(b,c)	8

Edge	Weight
(d,e)	9
(e,f)	10
(b,h)	11
(d,f)	14

Vertex	р	rank
а	b	0
b	b	1
С	i	0
d	d	0
е	е	0
f	h	0
g	h	0
h	h	2
i	h	1

```
A = \{(g,h), (f,g), (c,i), (a,b), (c,f)\}
```

```
FIND-SET(x)

1 if x \neq x.p

2 x.p = \text{FIND-SET}(x.p)

3 return x.p
```

```
LINK(x, y)

1 if x.rank > y.rank

2 y.p = x

3 else x.p = y

4 if x.rank == y.rank

5 y.rank = y.rank + 1
```

```
MST-KRUSKAL(G, w)

1 A = \emptyset

2 for each vertex v \in G.V

1 LINK(FIND-SET(x), FIND-SET(y))

4 sort the edges of G.E into nondecreasing order by weight w

5 for each edge (u, v) \in G.E, taken in nondecreasing order by weight

6 if FIND-SET(u) \neq FIND-SET(v)

7 A = A \cup \{(u, v)\}

UNION(u, v)

9 return A
```

Edge	Weight
(g,h)	1
(f,g)	2
(c,i)	2
(a,b)	4
(c,f)	4

Edge	Weight
(g,i)	6
(c,d)	7
(h,i)	7
(a,h)	8
(b,c)	8

Edge	Weight
(d,e)	9
(e,f)	10
(b,h)	11
(d,f)	14

Vertex	р	rank
а	b	0
b	b	1
С	i	0
d	d	0
е	е	0
f	h	0
g	h	0
h	h	2
i	h	1

```
A = \{(g,h), (f,g), (c,i), (a,b), (c,f)\}
```

```
FIND-SET(x)

1 if x \neq x.p

2 x.p = \text{FIND-SET}(x.p)

3 return x.p
```

```
LINK(x, y)

1 if x.rank > y.rank

2 y.p = x

3 else x.p = y

4 if x.rank == y.rank

5 y.rank = y.rank + 1
```

```
MST-KRUSKAL(G, w)

1 A = \emptyset

2 for each vertex v \in G.V

1 LINK(FIND-SET(x), FIND-SET(y))

4 sort the edges of G.E into nondecreasing order by weight w

5 for each edge (u, v) \in G.E, taken in nondecreasing order by weight

6 if FIND-SET(u) \neq FIND-SET(v)

7 A = A \cup \{(u, v)\}

UNION(u, v)

9 return A
```

Edge	Weight
(g,h)	1
(f,g)	2
(c,i)	2
(a,b)	4
(c,f)	4

Ec	lge	Weight
<b>(</b> g	g,i)	6
(c	,d)	7
(h	ı,i)	7
(a	,h)	8
(b	,c)	8

Edge	Weight
(d,e)	9
(e,f)	10
(b,h)	11
(d,f)	14

р	rank
b	0
b	1
i	0
d	0
е	0
h	0
h	0
h	2
h	1
	b i d e h h

```
A = \{(g,h), (f,g), (c,i), (a,b), (c,f)\}
```

```
FIND-SET(x)

1 if x \neq x.p

2 x.p = \text{FIND-SET}(x.p)

3 return x.p
```

```
LINK(x, y)

1 if x.rank > y.rank

2 y.p = x

3 else x.p = y

4 if x.rank == y.rank

5 y.rank = y.rank + 1
```

```
MST-KRUSKAL(G, w)

1 A = \emptyset

2 for each vertex v \in G.V

3 MAKE-SET(v)

4 sort the edges of G.E into nondecreasing order by weight w

5 for each edge (u, v) \in G.E, taken in nondecreasing order by weight

6 if FIND-SET(u) \neq FIND-SET(v)

7 A = A \cup \{(u, v)\}

UNION(u, v)

9 return A
```

Edge	Weight
(g,h)	1
(f,g)	2
(c,i)	2
(a,b)	4
(c,f)	4

Edge	Weight
(g,i)	6
(c,d)	7
(h,i)	7
(a,h)	8
(b,c)	8

Edge	Weight
(d,e)	9
(e,f)	10
(b,h)	11
(d,f)	14

Vertex	р	rank
а	b	0
b	b	1
С	h	0
d	h	0
е	е	0
f	h	0
g	h	0
h	h	2
i	h	1

```
A = \{(g,h), (f,g), (c,i), (a,b), (c,f), (c,d)\}
```

```
FIND-SET(x)

1 if x \neq x.p

2 x.p = \text{FIND-SET}(x.p)

3 return x.p
```

```
LINK(x, y)

1 if x.rank > y.rank

2 y.p = x

3 else x.p = y

4 if x.rank == y.rank

5 y.rank = y.rank + 1
```

```
MST-KRUSKAL(G, w)

1 A = \emptyset

2 for each vertex v \in G.V

3 MAKE-SET(v)

4 sort the edges of G.E into nondecreasing order by weight w

5 for each edge (u, v) \in G.E, taken in nondecreasing order by weight

6 if FIND-SET(u) \neq FIND-SET(v)

7 A = A \cup \{(u, v)\}

UNION(u, v)

9 return A
```

Edge	Weight
(g,h)	1
(f,g)	2
(c,i)	2
(a,b)	4
(c,f)	4

Edge	Weight
(g,i)	6
(c,d)	7
(h,i)	7
(a,h)	8
(b,c)	8

Edge	Weight
(d,e)	9
(e,f)	10
(b,h)	11
(d,f)	14

Vertex	р	rank
а	b	0
b	b	1
С	h	0
d	h	0
е	е	0
f	h	0
g	h	0
h	h	2
i	h	1

```
A = \{(g,h), (f,g), (c,i), (a,b), (c,f), (c,d)\}
```

```
FIND-SET(x)

1 if x \neq x.p

2 x.p = \text{FIND-SET}(x.p)

3 return x.p
```

```
LINK(x, y)

1 if x.rank > y.rank

2 y.p = x

3 else x.p = y

4 if x.rank == y.rank

5 y.rank = y.rank + 1
```

```
MST-KRUSKAL(G, w)

1 A = \emptyset

2 for each vertex v \in G.V

3 MAKE-SET(v)

4 sort the edges of G.E into nondecreasing order by weight w

5 for each edge (u, v) \in G.E, taken in nondecreasing order by weight

6 if FIND-SET(u) \neq FIND-SET(v)

7 A = A \cup \{(u, v)\}

UNION(u, v)

9 return A
```

Edge	Weight
(g,h)	1
(f,g)	2
(c,i)	2
(a,b)	4
(c,f)	4

Edge	Weight
(g,i)	6
(c,d)	7
(h,i)	7
(a,h)	8
(b,c)	8

Edge	Weight
(d,e)	9
(e,f)	10
(b,h)	11
(d,f)	14

Vertex	р	rank
а	b	0
b	b	1
С	h	0
d	h	0
е	е	0
f	h	0
g	h	0
h	h	2
i	h	1

```
A = \{(g,h), (f,g), (c,i), (a,b), (c,f), (c,d)\}
```

```
FIND-SET(x)

1 if x \neq x.p

2 x.p = \text{FIND-SET}(x.p)

3 return x.p
```

```
LINK(x, y)

1 if x.rank > y.rank

2 y.p = x

3 else x.p = y

4 if x.rank == y.rank

5 y.rank = y.rank + 1
```

```
MST-KRUSKAL(G, w)

1 A = \emptyset

2 for each vertex v \in G.V

1 LINK(FIND-SET(x), FIND-SET(y))

4 sort the edges of G.E into nondecreasing order by weight w

5 for each edge (u, v) \in G.E, taken in nondecreasing order by weight

6 if FIND-SET(u) \neq FIND-SET(v)

7 A = A \cup \{(u, v)\}

UNION(u, v)

9 return A
```

Edge	Weight
(g,h)	1
(f,g)	2
(c,i)	2
(a,b)	4
(c,f)	4

Edge	Weight
(g,i)	6
(c,d)	7
(h,i)	7
(a,h)	8
(b,c)	8

Edge	Weight
(d,e)	9
(e,f)	10
(b,h)	11
(d,f)	14

р	rank
b	0
h	1
h	0
h	0
е	0
h	0
h	0
h	2
h	1
	b h h h h h

```
A = {(g,h), (f,g), (c,i),
(a,b), (c,f), (c,d),
(a,h)}
```

```
FIND-SET(x)

1 if x \neq x.p

2 x.p = \text{FIND-SET}(x.p)

3 return x.p
```

```
LINK(x, y)

1 if x.rank > y.rank

2 y.p = x

3 else x.p = y

4 if x.rank == y.rank

5 y.rank = y.rank + 1
```

```
MST-KRUSKAL(G, w)

1 A = \emptyset

2 for each vertex v \in G.V

1 LINK(FIND-SET(x), FIND-SET(y))

4 sort the edges of G.E into nondecreasing order by weight w

5 for each edge (u, v) \in G.E, taken in nondecreasing order by weight

6 if FIND-SET(u) \neq FIND-SET(v)

7 A = A \cup \{(u, v)\}

UNION(u, v)

9 return A
```

<b>F.I.</b>	307.1.1.1
Edge	Weight
(g,h)	1
(f,g)	2
(c,i)	2
(a,b)	4
(c,f)	4

Edge	Weight
(g,i)	6
(c,d)	7
(h,i)	7
(a,h)	8
(b,c)	8

Edge	Weight
(d,e)	9
(e,f)	10
(b,h)	11
(d,f)	14

Vertex	р	rank
а	b	0
b	h	1
С	h	0
d	h	0
е	е	0
f	h	0
g	h	0
h	h	2
i	h	1

```
A = {(g,h), (f,g), (c,i),
(a,b), (c,f), (c,d),
(a,h)}
```

```
FIND-SET(x)

1 if x \neq x.p

2 x.p = \text{FIND-SET}(x.p)

3 return x.p
```

```
LINK(x, y)

1 if x.rank > y.rank

2 y.p = x

3 else x.p = y

4 if x.rank == y.rank

5 y.rank = y.rank + 1
```

```
MST-KRUSKAL(G, w)

1 A = \emptyset

2 for each vertex v \in G.V

3 MAKE-SET(v)

4 sort the edges of G.E into nondecreasing order by weight w

5 for each edge (u, v) \in G.E, taken in nondecreasing order by weight

6 if FIND-SET(u) \neq FIND-SET(v)

7 A = A \cup \{(u, v)\}

UNION(u, v)

9 return A
```

Edge	Weight
(g,h)	1
(f,g)	2
(c,i)	2
(a,b)	4
(c,f)	4

Edge	Weight
(g,i)	6
(c,d)	7
(h,i)	7
(a,h)	8
(b,c)	8

Edge	Weight
(d,e)	9
(e,f)	10
(b,h)	11
(d,f)	14

Vertex	р	rank
а	b	0
b	h	1
С	h	0
d	h	0
е	е	0
f	h	0
g	h	0
h	h	2
i	h	1

```
A = {(g,h), (f,g), (c,i),
(a,b), (c,f), (c,d),
(a,h)}
```

```
FIND-SET(x)

1 if x \neq x.p

2 x.p = \text{FIND-SET}(x.p)

3 return x.p
```

```
LINK(x, y)

1 if x.rank > y.rank

2 y.p = x

3 else x.p = y

4 if x.rank == y.rank

5 y.rank = y.rank + 1
```

```
MST-KRUSKAL(G, w)

1 A = \emptyset

2 for each vertex v \in G.V

1 LINK(FIND-SET(x), FIND-SET(y))

4 sort the edges of G.E into nondecreasing order by weight w

5 for each edge (u, v) \in G.E, taken in nondecreasing order by weight

6 if FIND-SET(u) \neq FIND-SET(v)

7 A = A \cup \{(u, v)\}

UNION(u, v)

9 return A
```

Edge	Weight
(g,h)	1
(f,g)	2
(c,i)	2
(a,b)	4
(c,f)	4

Edge	Weight
(g,i)	6
(c,d)	7
(h,i)	7
(a,h)	8
(b,c)	8

Edge	Weight
(d,e)	9
(e,f)	10
(b,h)	11
(d,f)	14

Vertex	р	rank
а	b	0
b	h	1
С	h	0
d	h	0
е	h	0
f	h	0
g	h	0
h	h	2
i	h	1

```
A = {(g,h), (f,g), (c,i),
(a,b), (c,f), (c,d),
(a,h), (d,e)}
```

```
FIND-SET(x)

1 if x \neq x.p

2 x.p = \text{FIND-SET}(x.p)

3 return x.p
```

```
LINK(x, y)

1 if x.rank > y.rank

2 y.p = x

3 else x.p = y

4 if x.rank == y.rank

5 y.rank = y.rank + 1
```

```
MST-KRUSKAL(G, w)

1 A = \emptyset

2 for each vertex v \in G.V

1 LINK(FIND-SET(x), FIND-SET(y))

4 sort the edges of G.E into nondecreasing order by weight w

5 for each edge (u, v) \in G.E, taken in nondecreasing order by weight

6 if FIND-SET(u) \neq FIND-SET(v)

7 A = A \cup \{(u, v)\}

UNION(u, v)

9 return A
```

Edge	Weight
(g,h)	1
(f,g)	2
(c,i)	2
(a,b)	4
(c,f)	4

Edge	Weight
(g,i)	6
(c,d)	7
(h,i)	7
(a,h)	8
(b,c)	8

Edge	Weight
(d,e)	9
(e,f)	10
(b,h)	11
(d,f)	14

р	rank
b	0
h	1
h	0
h	0
h	0
h	0
h	0
h	2
h	1
	b h h h h h

```
A = {(g,h), (f,g), (c,i),
(a,b), (c,f), (c,d),
(a,h), (d,e)}
```

```
FIND-SET(x)

1 if x \neq x.p

2 x.p = \text{FIND-SET}(x.p)

3 return x.p
```

```
LINK(x, y)

1 if x.rank > y.rank

2 y.p = x

3 else x.p = y

4 if x.rank == y.rank

5 y.rank = y.rank + 1
```

```
MST-KRUSKAL(G, w)

1 A = \emptyset

2 for each vertex v \in G.V

1 LINK(FIND-SET(x), FIND-SET(y))

4 sort the edges of G.E into nondecreasing order by weight w

5 for each edge (u, v) \in G.E, taken in nondecreasing order by weight

6 if FIND-SET(u) \neq FIND-SET(v)

7 A = A \cup \{(u, v)\}

UNION(u, v)

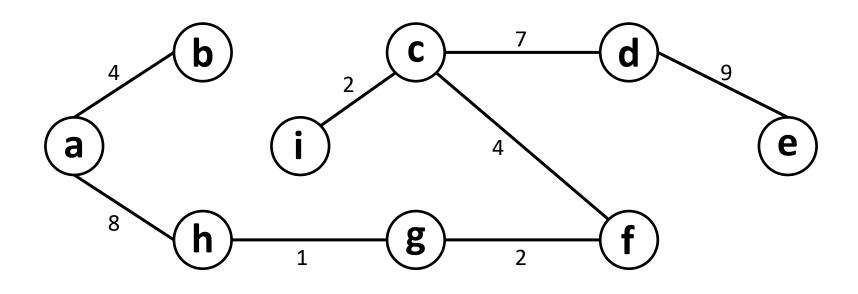
9 return A
```

Edge	Weight
(g,h)	1
(f,g)	2
(c,i)	2
(a,b)	4
(c,f)	4

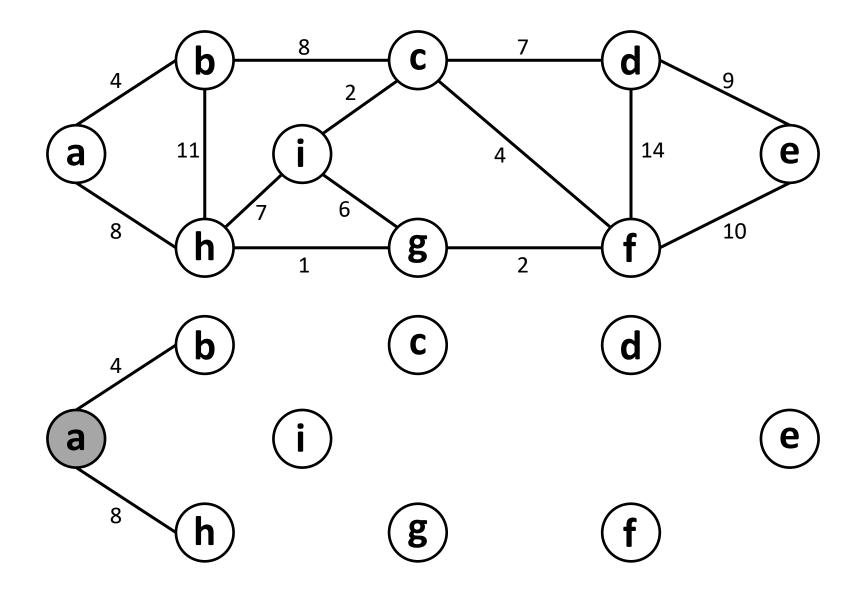
Edge	Weight
(g,i)	6
(c,d)	7
(h,i)	7
(a,h)	8
(b,c)	8

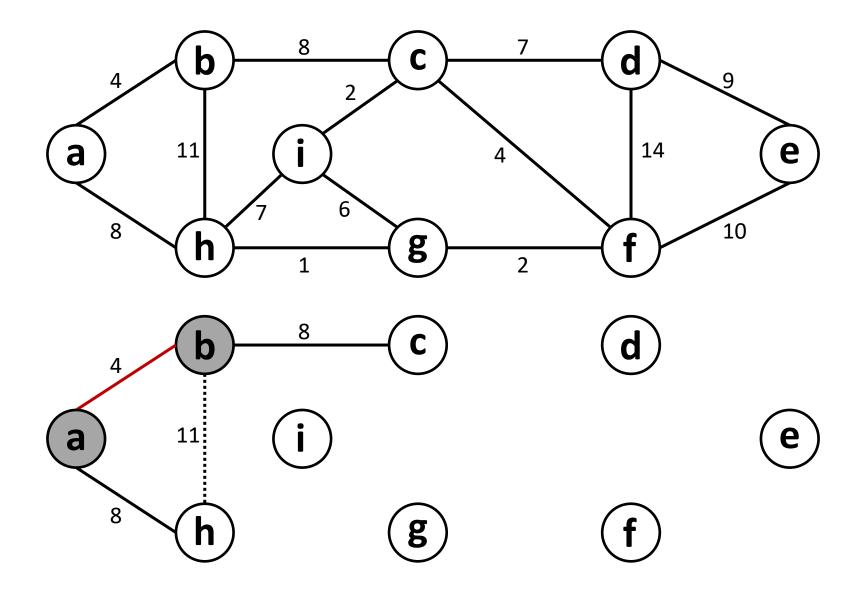
Edge	Weight
(d,e)	9
(e,f)	10
(b,h)	11
(d,f)	14

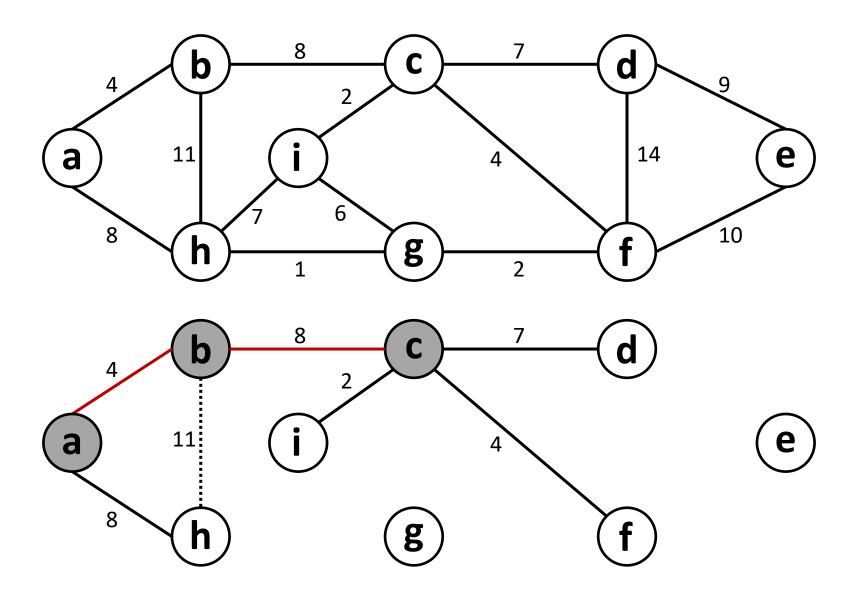
 $A = \{(g,h), (f,g), (c,i), (a,b), (c,f), (c,d), (a,h), (d,e)\}$ 

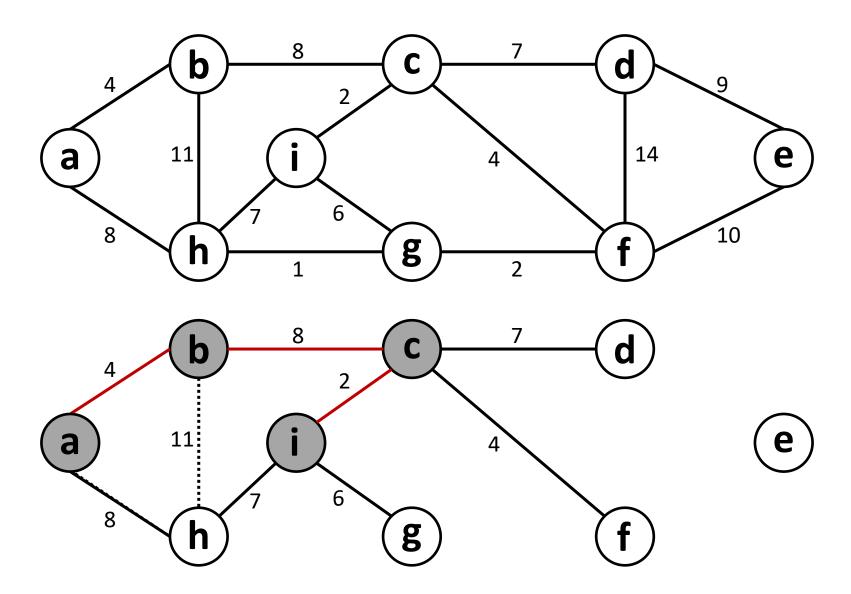


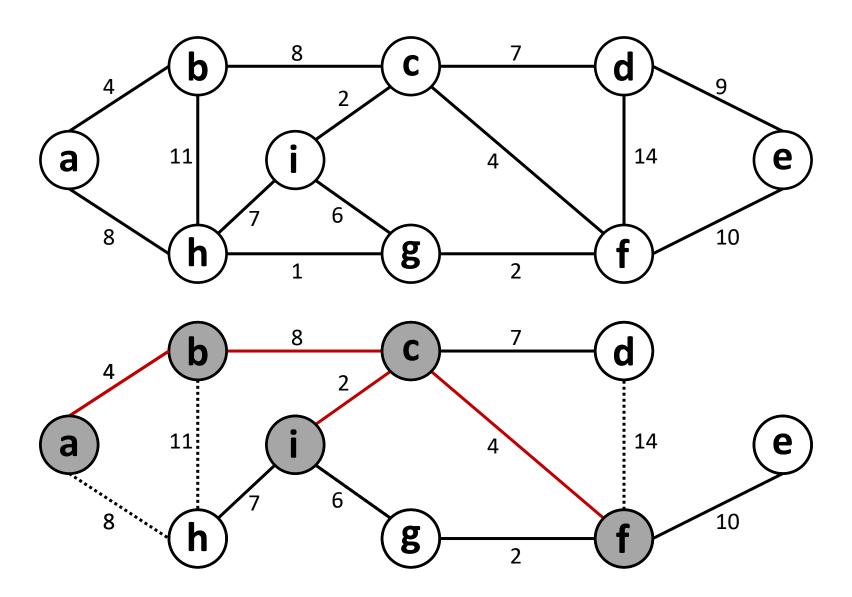
Cost = 37

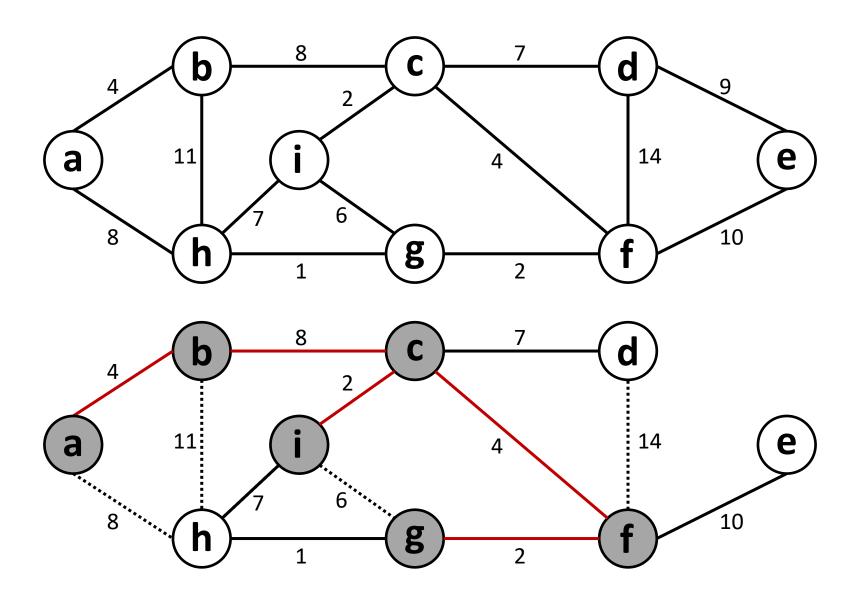


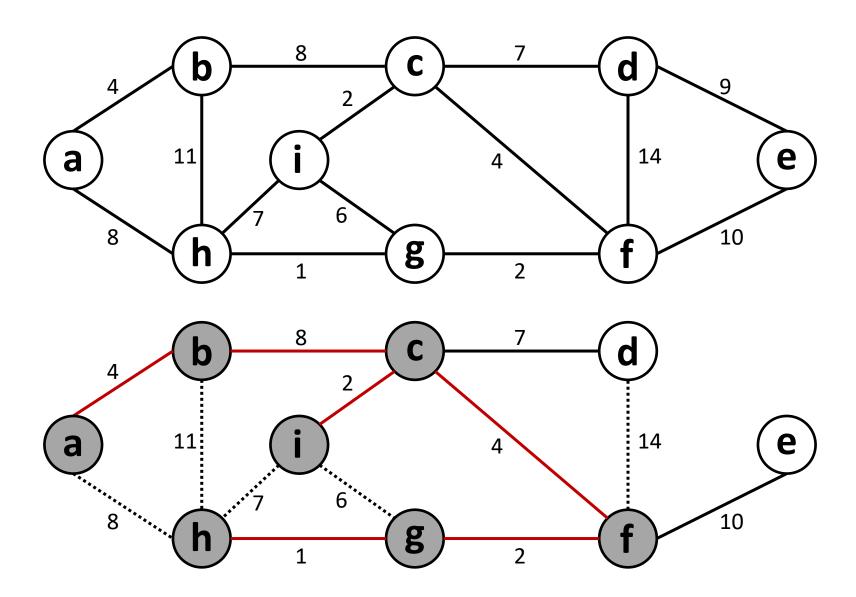


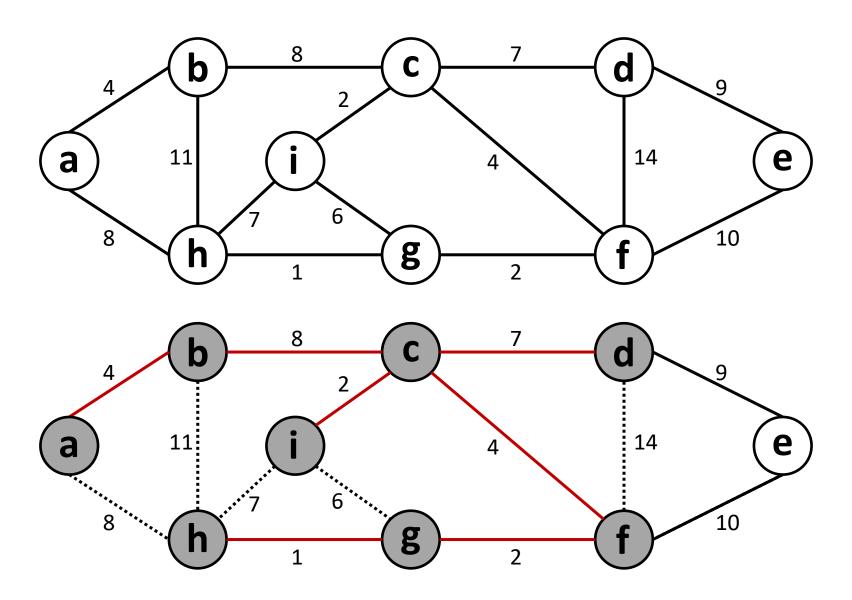


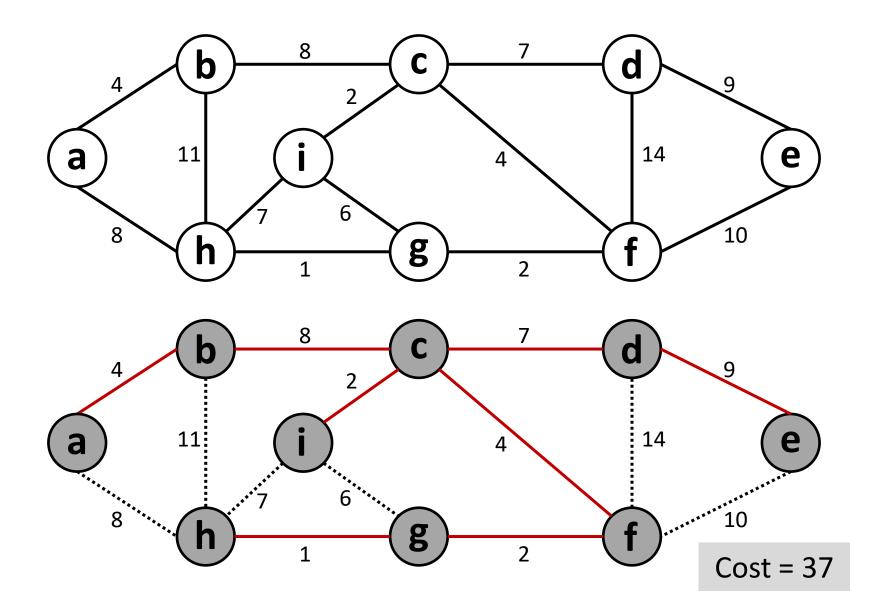












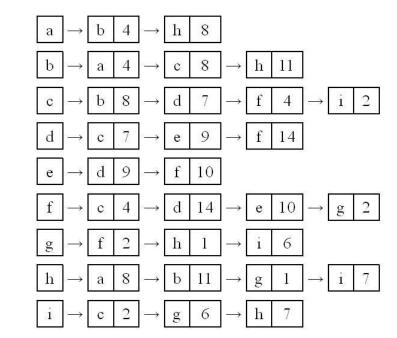
## Implementation

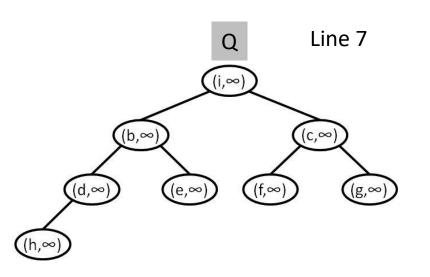
```
11
MST-PRIM(G, w, r)
                                                                   9
                                                                             14
     for each u \in G.V
                                                                  10
          u.key = \infty
                                                              d
                                                                  14 \rightarrow e
                                                                             10
         u.\pi = NIL
    r.key = 0
                                                              h
     Q = G.V
                                                              b
                                                                  11
     while Q \neq \emptyset
          u = \text{EXTRACT-MIN}(Q)
                                                            \rightarrow \mid g
          for each v \in G.Adj[u]
               if v \in Q and w(u, v) < v.key
                    \nu.\pi = u
                                                 b
                    v.key = w(u, v)
                                                11
                                                                               14
                                         a
 A = \{(v, v.\pi) : v \in V - \{r\} - Q\}
                                                        1
```

### **Example - Execution**

Vertex	π	key
а	NIL	0
b	NIL	8
С	NIL	8
d	NIL	8
е	NIL	8
f	NIL	8
g	NIL	8
h	NIL	8
i	NIL	8

```
MST-PRIM(G, w, r)
     for each u \in G.V
         u.key = \infty
         u.\pi = NIL
    r.key = 0
     Q = G.V
    while Q \neq \emptyset
         u = \text{EXTRACT-MIN}(Q)
 8
         for each v \in G.Adj[u]
              if v \in Q and w(u, v) < v.key
10
                   \nu.\pi = u
                   v.key = w(u, v)
```





MST-PRIM(G, w, r)

r.key = 0

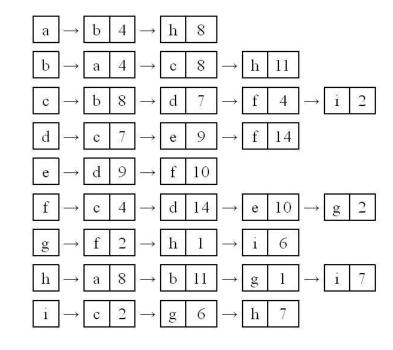
Q = G.V

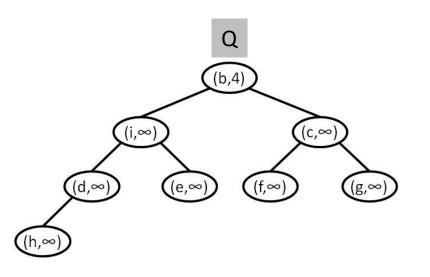
10

while  $Q \neq \emptyset$ 

for each  $u \in G.V$ 

$egin{array}{c cccc} oldsymbol{Vertex} & \pi & oldsymbol{key} \\ \hline a & oldsymbol{NIL} & O \end{array}$					
	Vertex	π	key		
	а	NIL	0		
	b	а	4		
	С	NIL	8		
	d	NIL	8		
$\operatorname{IM}(G, w, r)$	е	NIL	8		
each $u \in G.V$	f	NIL	8		
$u.key = \infty$	g	NIL	8		
$u.\pi = NIL$	h	NIL	8		
y = 0 = $G.V$	i	NIL	8		
$\begin{array}{c} = 0.7 \\ \text{le } Q \neq \emptyset \end{array}$					
u = EXTRACT-MI	N(Q)				
<b>for</b> each $v \in G.Adj$					
if $v \in Q$ and $u$		v.ke	v		
$v.\pi = u$	(,-)				
v.key = v	v(u, v)				
	(,,,,,				





MST-PRIM(G, w, r)

r.key = 0

Q = G.V

10

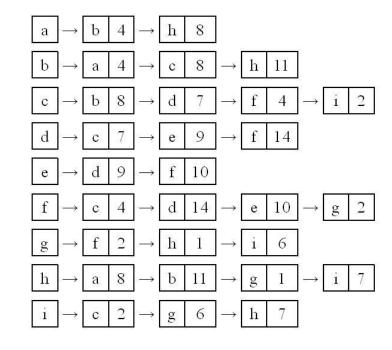
while  $Q \neq \emptyset$ 

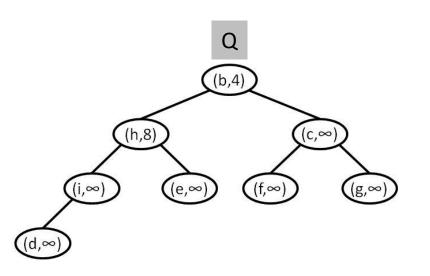
for each  $u \in G.V$ 

 $\nu.\pi = u$ 

v.key = w(u, v)

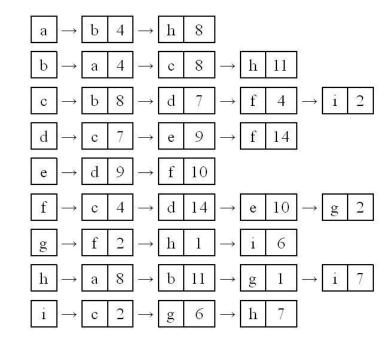
	Vertex	π	key	
	VEILEX	, n	ne y	
	a	NIL	0	
	b	а	4	
	С	NIL	8	
	d	NIL	8	
$\operatorname{IM}(G, w, r)$	е	NIL	8	
each $u \in G.V$	f	NIL	8	
$u.key = \infty$	g	NIL	8	
$u.\pi = NIL$	h	а	8	
y = 0 $= G.V$	i	NIL	8	
= 0. v le Q ≠ Ø				
u = EXTRACT-MI	N(O)			
for each $v \in G.Adj[u]$				
if $v \in Q$ and $w(u, v) < v.key$				

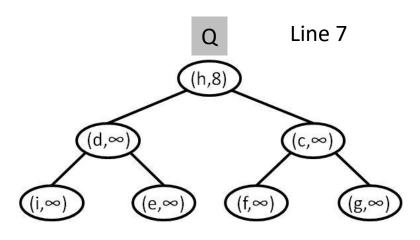




_ 0 4 _ ∞ _ ∞
_ ∞
_ ∞
_ ∞
1
_ ∞
_ ∞
8
_ ∞

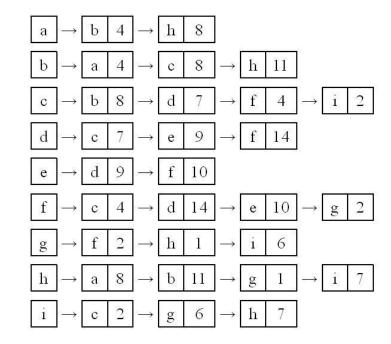
```
MST-PRIM(G, w, r)
     for each u \in G.V
         u.key = \infty
         u.\pi = NIL
    r.key = 0
     Q = G.V
    while Q \neq \emptyset
         u = \text{EXTRACT-MIN}(Q)
         for each v \in G.Adj[u]
              if v \in Q and w(u, v) < v.key
10
                   \nu.\pi = u
                   v.key = w(u, v)
```

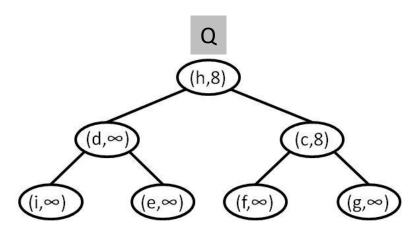




Vertex	π	key
а	NIL	0
b	а	4
С	b	8
d	NIL	∞
е	NIL	∞
f	NIL	∞
g	NIL	∞
h	а	8
i	NIL	∞

		b	a	4		
		С	b	8		
		d	NIL	8		
MS	T-PRIM $(G, w, r)$	е	NIL	8		
1	for each $u \in G.V$	f	NIL	8		
2	$u.key = \infty$	g	NIL	8		
3	$u.\pi = NIL$	h	а	8		
4	$r.key = 0$ i NIL $\infty$					
5	Q = G.V		<u> </u>			
6	while $Q \neq \emptyset$					
7	u = EXTRACT-MI	N(Q)				
8	<b>for</b> each $v \in G$ . Adj	i[u]				
9	if $v \in Q$ and $w(u, v) < v.key$					
0	$v.\pi = u$					
11	v.key = v	v(u, v)				

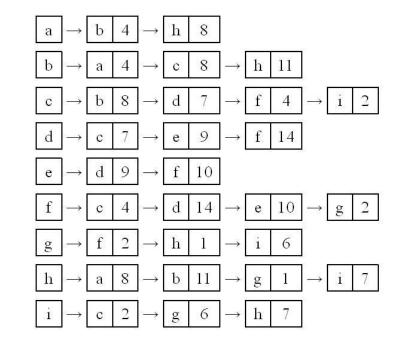


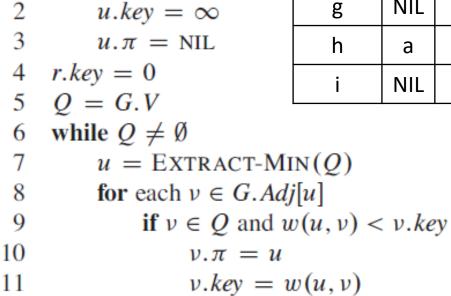


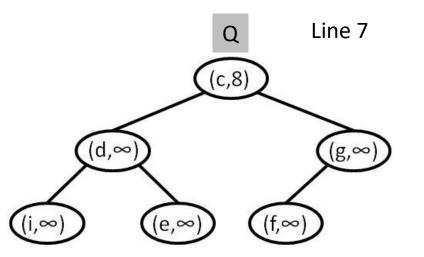
MST-PRIM(G, w, r)

for each  $u \in G.V$ 

Vertex	π	key			
а	NIL	0			
b	а	4			
С	b	8			
d	NIL	8			
е	NIL	8			
f	NIL	8			
g	NIL	8			
h	а	8			
i	NIL	8			
N(Q) $[u]$					

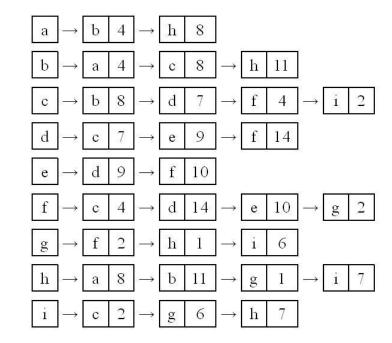


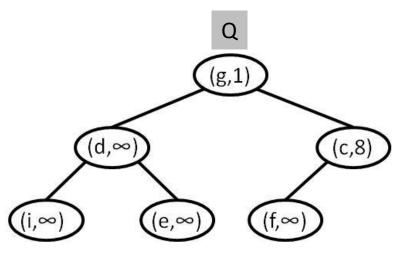




Vertex	π	key
а	NIL	0
b	а	4
С	b	8
d	NIL	8
е	NIL	8
f	NIL	8
g	h	1
h	а	8
i	NIL	8

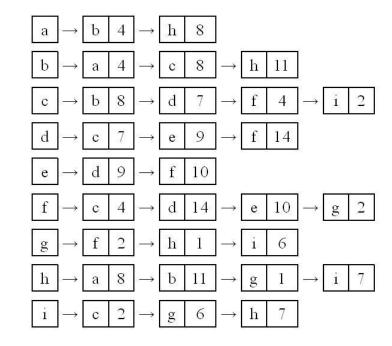
		d	NIL		
MST-PRIM(G, w, r)		е	NIL		
1	for each $u \in G.V$	f	NIL		
2	$u.key = \infty$	g	h		
3	$u.\pi = NIL$	h	а		
4	r.key = 0	i	NIL		
	Q = G.V	•		ı	
6	while $Q \neq \emptyset$				
7	u = EXTRACT-MIN(Q)				
8	for each $v \in G.Adj[u]$				
9					
10	$v.\pi = u$				
11	v.key = w(u, v)				

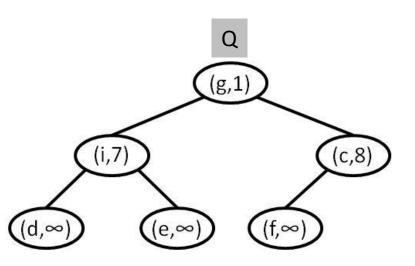




Vertex	π	key
а	NIL	0
b	а	4
С	b	8
d	NIL	8
е	NIL	8
f	NIL	8
g	h	1
h	а	8
i	h	7

		С	b		
		d	NIL		
MS	T-PRIM $(G, w, r)$	е	NIL		
1	for each $u \in G.V$	f	NIL		
2	$u.key = \infty$	g	h		
3	$u.\pi = NIL$	h	а		
4 5	r.key = 0	i	h		
5	Q = G.V				
6	6 while $Q \neq \emptyset$				
7	u = EXTRACT-MIN(Q)				
8	\ <del>~</del> /				
9	9 <b>if</b> $v \in Q$ and $w(u, v) < v.key$				
10	$v.\pi = u$				
11	1 $v.key = w(u, v)$				





MST-PRIM(G, w, r)

r.key = 0

Q = G.V

10

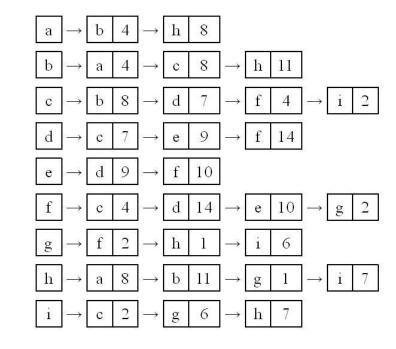
while  $Q \neq \emptyset$ 

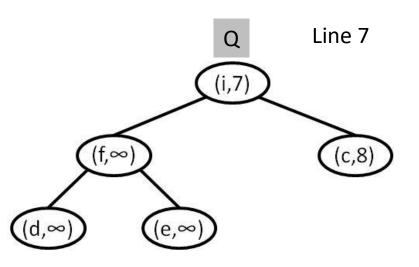
for each  $u \in G.V$ 

 $\nu.\pi = u$ 

v.key = w(u, v)

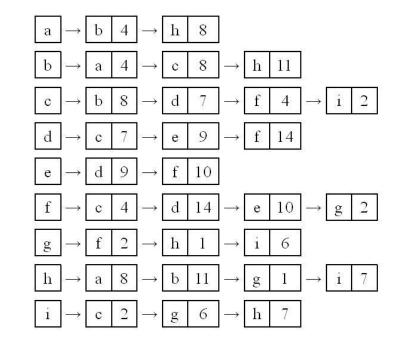
	Vortov		kov		
	Vertex	π	key		
	а	NIL	0		
	b	а	4		
	С	b	8		
	d	NIL	8		
$\operatorname{IM}(G, w, r)$	е	NIL	8		
each $u \in G.V$	f	NIL	8		
$u.key = \infty$	g	h	1		
$u.\pi = NIL$	h	а	8		
y = 0	i	h	7		
$= G.V$ $le Q \neq \emptyset$					
u = EXTRACT-MIN(Q)					
for each $v \in G.Adj[u]$					
if $v \in Q$ and $w(u, v) < v.key$					

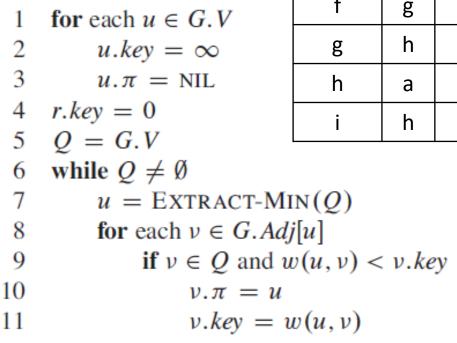


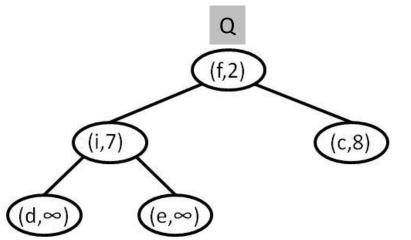


MST-PRIM(G, w, r)

Vertex	π	key
а	NIL	0
b	а	4
С	b	8
d	NIL	8
е	NIL	8
f	g	2
g	h	1
h	а	8
i	h	7

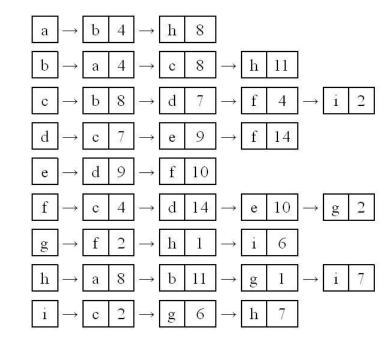


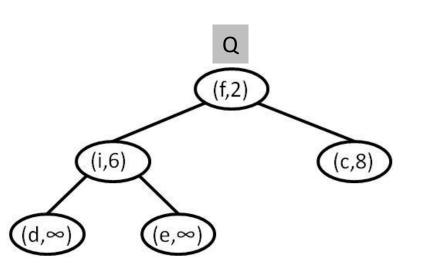




Vertex	π	key
а	NIL	0
b	а	4
С	b	8
d	NIL	~
е	NIL	8
f	g	2
g	h	1
h	а	8
i	g	6

		С	b	
		d	NIL	
MS	T-PRIM $(G, w, r)$	е	NIL	
1	for each $u \in G.V$	f	യ	
2	$u.key = \infty$	g	h	
3	$u.\pi = NIL$	h	а	
4	r.key = 0 $Q = G.V$	i	g	
5	Q = G.V	•	0	
	while $Q \neq \emptyset$			
7	u = EXTRACT-MI	N(Q)		
8	<b>for</b> each $v \in G.Adj$	[u]		
9	if $v \in Q$ and $u$	v(u,v) <	v.ke	y
10	$v.\pi = u$			
11	v.key = v	v(u,v)		





MST-PRIM(G, w, r)

r.key = 0

Q = G.V

10

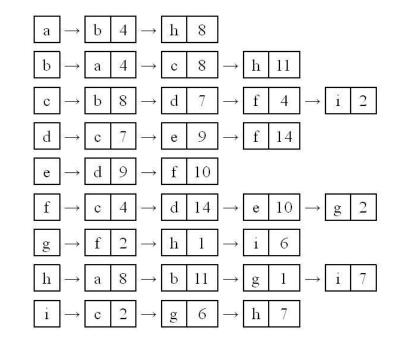
while  $Q \neq \emptyset$ 

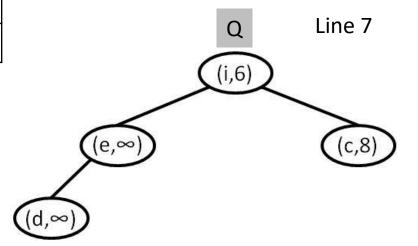
for each  $u \in G.V$ 

 $\nu.\pi = u$ 

v.key = w(u, v)

	Vertex	π	key	
	а	NIL	0	
	b	а	4	
	С	b	8	
	d	NIL	8	
$\operatorname{IM}(G, w, r)$	е	NIL	8	
each $u \in G.V$	f	g	2	
$u.key = \infty$	g	h	1	
$u.\pi = NIL$	h	а	8	
y = 0	i	g	6	
$= G.V$ $le Q \neq \emptyset$				
u = EXTRACT-MIN(Q)				
for each $v \in G.Adj[u]$				
if $v \in Q$ and $u$		v.ke	y	





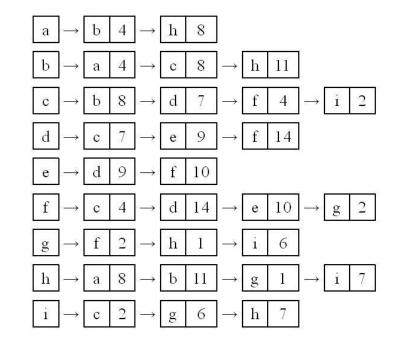
MST-PRIM(G, w, r)

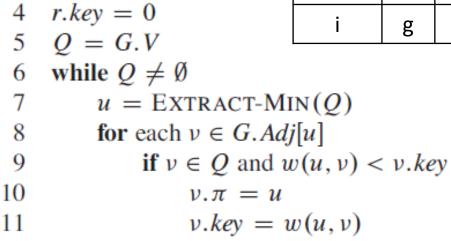
for each  $u \in G.V$ 

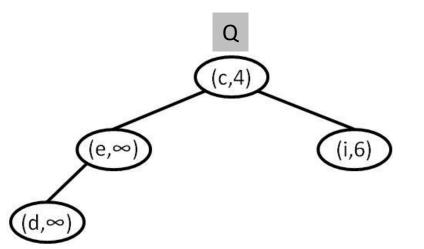
 $u.key = \infty$ 

 $u.\pi = NIL$ 

Vertex	π	key
а	NIL	0
b	а	4
С	f	4
d	NIL	8
е	NIL	8
f	g	2
g	h	1
h	а	8
i	g	6







MST-PRIM(G, w, r)

r.key = 0

Q = G.V

10

while  $Q \neq \emptyset$ 

for each  $u \in G.V$ 

 $u.key = \infty$ 

 $u.\pi = NIL$ 

u = EXTRACT-MIN(Q)

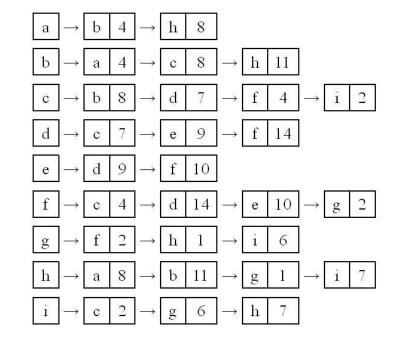
 $\nu.\pi = u$ 

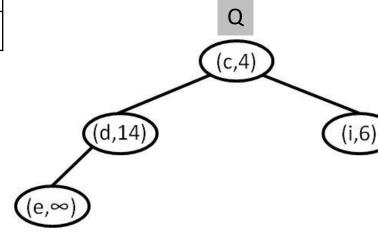
if  $v \in Q$  and w(u, v) < v.key

v.key = w(u, v)

for each  $v \in G.Adj[u]$ 

Vertex	π	key
а	NIL	0
b	а	4
С	f	4
d	f	14
е	NIL	8
f	g	2
g	h	1
h	а	8
i	g	6





MST-PRIM(G, w, r)

r.key = 0

Q = G.V

10

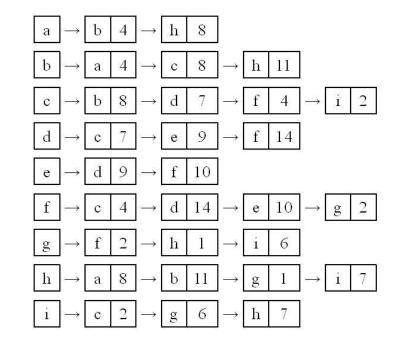
while  $Q \neq \emptyset$ 

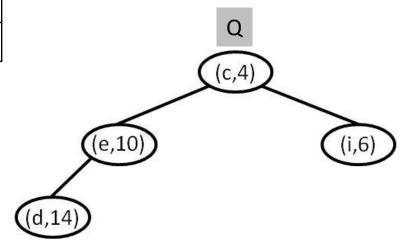
for each  $u \in G.V$ 

 $\nu.\pi = u$ 

v.key = w(u, v)

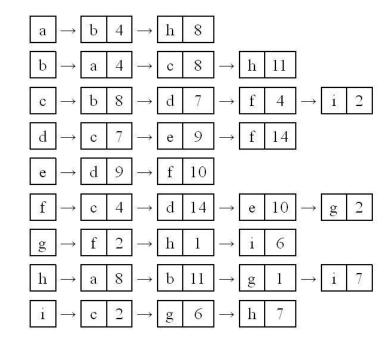
	Vertex	π	key	
	а	NIL	0	
	b	а	4	
	С	f	4	
	d	f	14	
$\operatorname{IM}(G, w, r)$	е	f	10	
each $u \in G.V$	f	g	2	
$u.key = \infty$	g	h	1	
$u.\pi = NIL$	h	а	8	
y = 0	i	g	6	
$= G.V$ $le Q \neq \emptyset$				
u = EXTRACT-MIN(Q)				
for each $v \in G.Adj[u]$				
if $v \in Q$ and $u$	v(u,v) <	v.ke	y	
	-		-	

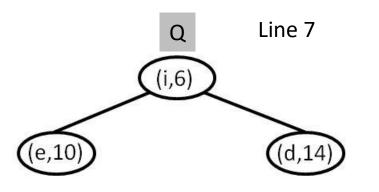




Vertex	π	key
а	NIL	0
b	а	4
С	f	4
d	f	14
е	f	10
f	g	2
g	h	1
h	а	8
i	g	6

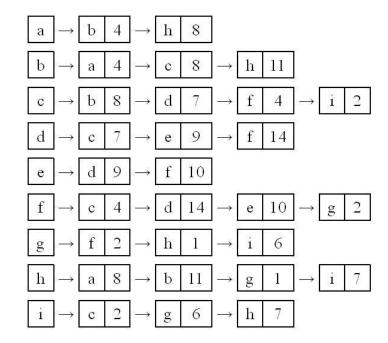
		d	f
MS	T-PRIM $(G, w, r)$	е	f
1	for each $u \in G.V$	f	g
2	$u.key = \infty$	g	h
3	$u.\pi = NIL$	h	а
4	r.key = 0	i	g
5	Q = G.V	•	8
6	while $Q \neq \emptyset$		
7	u = EXTRACT-MI	N(Q)	
8	for each $v \in G$ . Adj	[u]	
9	if $v \in Q$ and $u$	v(u,v) <	v.ke
10	$v.\pi = u$		
11	v.key = u	v(u,v)	

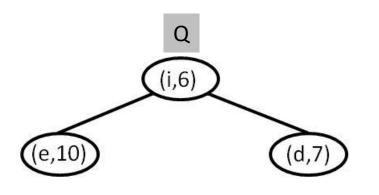




Vertex	π	key
а	NIL	0
b	а	4
С	f	4
d	С	7
е	f	10
f	g	2
g	h	1
h	а	8
i	g	6

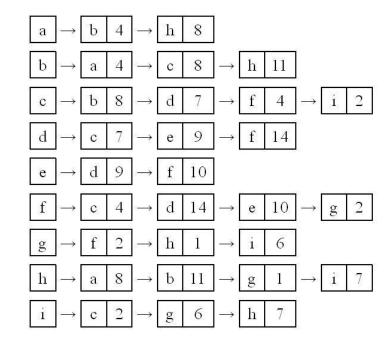
		5	
MS	T-PRIM $(G, w, r)$	е	f
1	for each $u \in G.V$	f	g
2	$u.key = \infty$	g	h
3	$u.\pi = NIL$	h	а
	r.key = 0	i	g
	Q = G.V	•	ъ
6	while $Q \neq \emptyset$		
7	u = EXTRACT-MI	N(Q)	
8	for each $v \in G.Adj$	[u]	
9	if $v \in Q$ and $u$	v(u,v) <	v.ke
10	$v.\pi = u$		
11	v.key = u	v(u,v)	

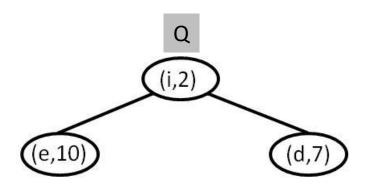




Vertex	π	key
а	NIL	0
b	а	4
С	f	4
d	С	7
е	f	10
f	g	2
g	h	1
h	а	8
i	С	2

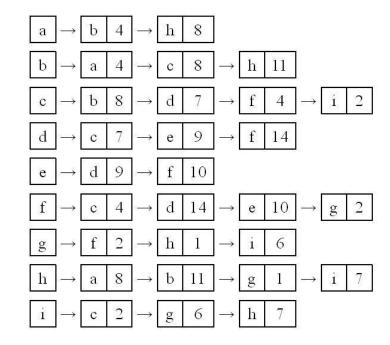
```
MST-PRIM(G, w, r)
     for each u \in G.V
         u.key = \infty
         u.\pi = NIL
    r.key = 0
     Q = G.V
    while Q \neq \emptyset
         u = \text{EXTRACT-MIN}(Q)
         for each v \in G.Adj[u]
 8
              if v \in Q and w(u, v) < v.key
10
                   \nu.\pi = u
                   v.key = w(u, v)
```

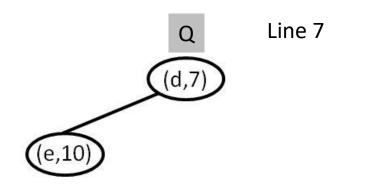




Vertex	π	key
а	NIL	0
b	а	4
С	f	4
d	С	7
е	f	10
f	g	2
g	h	1
h	а	8
i	С	2

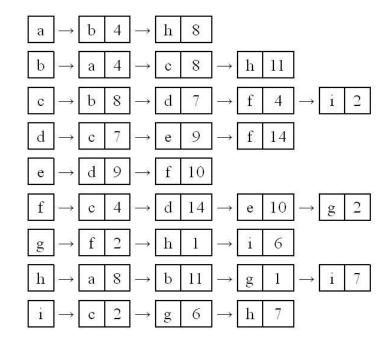
		a	С		
MS	T-PRIM $(G, w, r)$	е	f		
1	for each $u \in G.V$	f	g		
2	$u.key = \infty$	g	h		
3	$u.\pi = NIL$	h	а		
4	r.key = 0 $Q = G.V$	i	С		
	Q = G.V	•			
6 while $Q \neq \emptyset$					
7 $u = \text{EXTRACT-MIN}(Q)$					
8	<b>for</b> each $v \in G.Adj$	[u]			
9 <b>if</b> $v \in Q$ and $w(u, v) < v.ke$					
10	$v.\pi = u$				
11	v.key = v	v(u,v)			

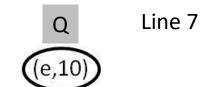




Vertex	π	key
а	NIL	0
b	а	4
С	f	4
d	С	7
е	f	10
f	g	2
g	h	1
h	а	8
i	С	2

MS	T-PRIM $(G, w, r)$	е	†		
1	for each $u \in G.V$	f	g		
2	$u.key = \infty$	g	h		
3	$u.\pi = NIL$	h	а		
4	r.key = 0	i	С		
5	Q = G.V	1	C		
6	6 while $Q \neq \emptyset$				
7	u = EXTRACT-MIN(Q)				
8	8 <b>for</b> each $v \in G.Adj[u]$				
9 <b>if</b> $v \in Q$ and $w(u, v) < v.key$					
10	$v.\pi = u$				
11	v.key = w(u, v)				



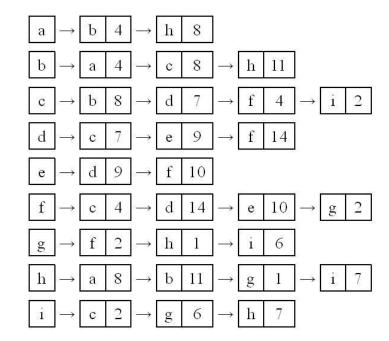


MST-PRIM(G, w, r)

for each  $u \in G V$ 

Vertex	π	key
а	NIL	0
b	а	4
С	f	4
d	С	7
е	d	9
f	g	2
g	h	1
h	а	8
i	С	2

	<b>TOT</b> Cacif $u \in G$ . $v$			
2	$u.key = \infty$	g	h	
3	$u.\pi = NIL$	h	а	
4	r.key = 0	i	С	
5	Q = G.V	•		
6	6 while $Q \neq \emptyset$			
7	u = EXTRACT-MIN(Q)			
8	8 <b>for</b> each $v \in G.Adj[u]$			
9	9 <b>if</b> $v \in Q$ and $w(u, v) < v.key$			
10	$0 \qquad \qquad \nu.\pi = u$			
11	v.key = w(u, v)			



Q (e,9)

```
A = \{(b,a), (c,f), (d,c), \}
(e,d), (f,g), (g,h),
(h,a), (i,c)}
```

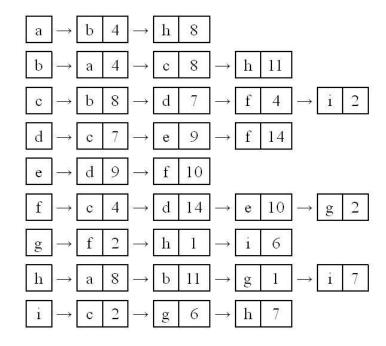
Vertex	π	key	
а	NIL	0	
b	а	4	
С	f	4	
d	С	7	
е	d	9	
f	g	2	
g	h	1	
h	а	8	
i	С	2	

```
MST-PRIM(G, w, r)
     for each u \in G.V
         u.key = \infty
 3
         u.\pi = NIL
    r.key = 0
     Q = G.V
     while Q \neq \emptyset
```

8

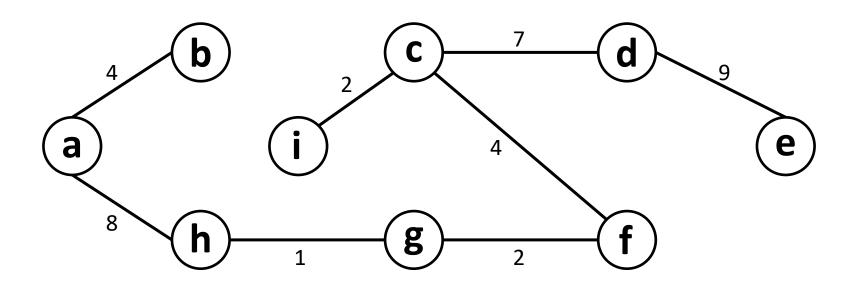
10

= 0.7
le $Q \neq \emptyset$
u = EXTRACT-MIN(Q)
for each $v \in G.Adj[u]$
if $v \in Q$ and $w(u, v) < v.key$
$\nu.\pi = u$
v.key = w(u, v)



Line 7 Q

 $A = \{(b,a), (c,f), (d,c), (e,d), (f,g), (g,h), (h,a), (i,c)\}$ 



Cost = 37