Kruskal's Algorithm:

```
1  A = ∅
2  for each vertex v ∈ 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) ∈ G.E, taken in nondecreasing order by weight
6      if FIND-SET(u) ≠ FIND-SET(v)
7          A = A ∪ {(u, v)}
8          UNION(u, v)
9  return A
```

Kruskal's algorithm in java is implemented using the above algorithm.

- MstKruskalsFunc() function is used to implement the kruskal's algorithm .
- Unionf class contains parent and rank attributes. And for all the vertices we compute separate components.
- After computing the connected components, we perform the sort of edges using heap sort. (mlogm - complexity).
- Then for all edges we perform two find operations (find(s1)) which is used to eliminate cycles if both the files are not equal then add edge to the output set. (mlogn – complexity).
- Then perform union operation (union (s1, s2) function) of the two connected components. (n – complexity).
- Overallcomplexity O (mlogm + mlogn + n);
    - mlogm – sort
    - mlogn – find
    - n – union

Prims Algorithm:

```
1  for each u ∈ G.V
2      u.key = ∞
3      u.π = NIL
4  r.key = 0
5  Q = G.V
6  while Q ≠ ∅
7      u = EXTRACT-MIN(Q)
8      for each v ∈ G.Adj[u]
9          if v ∈ Q and w(u, v) < v.key
10             v.π = u
11             v.key = w(u, v)
```

Prim's algorithm in java is implemented using the above algorithm.

- Prim () function is used to implement prim algorithm.
- It takes the graph containing vertices and edges along with the starting vertex. Source vertex is given as the source of the first src-> dest edge.
- Then for all the vertices of the graph update the key with infinity and parent to null.

- Update the source vertex key to zero. (start vertex)
- Then put all the vertices of the graph in the queue.
- Then while the que is not empty do the below steps.
- Then extract the minimum labeled vertex from the queue. (nlogn – complexity).
- Then for all vertices adjacent to the minimum labeled vertex compute the update label operations. (mlogn -- complexity).
- Overallcomplexity O (mlogn + nlogn);
    - Nlogn – find minlabeled vertex from the queue.
    - Mlogn – update labeles of the adjacent vertices of the minimum labeled vertex.

Note: Comments are provided explaining each of the code in Mstgraph.java file.