**2)**

Class MAPPER

Method MAP(nodeMap n, tree t)

EMIT(nodeMap n)

For all (node a, node b) ϵ tree t do

EMIT(node b, nodeMap n[node a] + 1)

Class REDUCER

Method REDUCE(node n, distances [d1, d2,…])

EMIT(node n, min(distances [d1, d2,…]))

**3)**

**package** ICE6  
**import** org.apache.spark.{SparkConf, SparkContext}  
**object** ICE6 {  
 **def** main(args: Array[String]) {  
 // administration  
 System.*setProperty*("hadoop.home.dir", "C:\\winutils")  
 **val** config = **new** SparkConf()  
 .setAppName("ICE6")  
 .setMaster("local[\*]")  
 **val** sc = **new** SparkContext(config)  
 **val** tree = sc.textFile("src/main/scala/ICE6/tree.txt").map(line => (line.split(" ")(0), line.split(" ")(1)))  
 **var** nodes = (tree.keys ++ tree.values).distinct().map(x => (x, **if** (x == "S") 0 **else** 9999))  
 **while** (nodes.values.max() == 9999) {  
 nodes = nodes.union(tree.join(nodes).map(x => (x.\_2.\_1,x.\_2.\_2 + 1))).reduceByKey(\_ min \_)  
 }  
 nodes.saveAsTextFile("src/main/scala/ICE6/output")  
 }  
}