Hands On: Building a Degree Histogram

In this activity, we are going to build a Degree Histogram using the graph that we created in the previous activity *Hands-On: Building a Graph*. Make sure to complete that activity before starting this one.

Step 1. Count the number of vertices and edges.

Print the number of links.

Input:

```
metrosGraph.numEdges
```

output:

```
res9: Long = 65
```

Print the number of nodes.

Input:

```
metrosGraph.numVertices
```

output:

```
res10: Long = 93
```

Step 2. Define a min and max function for Spark's reduce method.

Define a min and max function.

Input:

```
def max(a: (VertexId, Int), b: (VertexId, Int)): (VertexId, Int) = {
  if (a._2 > b._2) a else b
}
```

Output:

```
max: (a: (org.apache.spark.graphx.VertexId, Int), b: (org.apache.spark.graphx.VertexId, Int
```

Input:

```
def min(a: (VertexId, Int), b: (VertexId, Int)): (VertexId, Int) = {
  if (a._2 <= b._2) a else b
}
```

Output:

```
min: (a: (org.apache.spark.graphx.VertexId, Int), b: (org.apache.spark.graphx.VertexId, Int
```

Step 3. Compute min and max degrees.

Find which VertexId and the edge count of the vertex with the most out edges. (This can be any vertex because all vertices have one out edge.)

Input:

```
metrosGraph.outDegrees.reduce(max)
```

Output:

```
res11: (org.apache.spark.graphx.VertexId, Int) = (44,1)
```

Print the returned vertex.

Input:

```
metrosGraph.vertices.filter(_._1 == 44).collect()
```

Output:

```
res12: Array[(org.apache.spark.graphx.VertexId, PlaceNode)] = Array((44,Metro(Toronto,60557
```

Find which VertexId and the edge count of the vertex with the most in edges.

Input:

```
metrosGraph.inDegrees.reduce(max)
```

Output:

```
res13: (org.apache.spark.graphx.VertexId, Int) = (108,14)
```

Print the returned vertex.

Input:

```
metrosGraph.vertices.filter(_._1 == 108).collect()
```

Output:

```
res14: Array[(org.apache.spark.graphx.VertexId, PlaceNode)] = Array((108,Country(United Sta
```

Find the number vertexes that have only one out edge.

Input:

```
metrosGraph.outDegrees.filter(_._2 <= 1).count
```

Output:

```
res15: Long = 65
```

Find the maximum and minimum degrees of the connections in the network.

Input:

```
metrosGraph.degrees.reduce(max)
```

Output:

```
res16: (org.apache.spark.graphx.VertexId, Int) = (108,14)
```

Input:

```
metrosGraph.degrees.reduce(min)
```

Output:

```
res17: (org.apache.spark.graphx.VertexId, Int) = (19,1)
```

Step 4. Compute the histogram data of the degree of connectedness.

Print the histogram data of the degrees for countries only.

Input:

```
metrosGraph.degrees.
  filter { case (vid, count) => vid >= 100 }. // Apply filter so only VertexId < 100 (count
  map(t => (t._2,t._1)).
  groupByKey.map(t => (t._1,t._2.size)).
  sortBy(_._1).collect()
```

Output:

```
res18: Array[(Int, Int)] = Array((1,18), (2,4), (3,2), (5,2), (9,1), (14,1))
```

We recommend you to go directly to the next activity *Hands-On: Plot the Degree Histogram*, because that activity will use the histogram you created in this activity.

Go to next item ✓ Completed