[1335] Histogram Class

This Histogram class does more than the simple version we just saw. For instance, it takes a simple List of strings and counts up the number of instances of each string so that the programmer (you) doesn't need to do that. You'll be using this specific class in this week's assignment.

```
class Histogram
{
   private int width;
   private int maxBarWidth;
   private int maxLabelWidth;
   private int minValue;
   private List<KeyValuePair<string, int>> bars;

   public Histogram(List<string> data, int width = 80, int maxLabelWidth = 10, int minValue = 0)
   {
      this.width = width;
      this.maxLabelWidth = maxLabelWidth;
      this.minValue = minValue;
      this.maxBarWidth = width - maxLabelWidth - 2;   // -2 for the space and pipe separator

      // Use a Dictionary to count up the frequency of each string in the list
      var barCounts = new Dictionary<string, int>();
      foreach (string item in data)
      {
         if (barCounts.ContainsKey(item))
            barCounts[item]++;
         else
            barCounts.Add(item, 1);
      }

      // Store strings and frequencies in a (sortable) list of key/value pairs
      this.bars = new List<KeyValuePair<string, int>>(barCounts);
   }

   public void Sort(Comparison<KeyValuePair<string, int>> f)
   {
      bars.Sort(f);
   }

   public override string ToString()
   {
      string s = "";
      string blankLabel = "".PadRight(maxLabelWidth);

      int maxValue = 0;
      foreach (KeyValuePair<string, int> bar in bars)
      {
         if (bar.Value > maxValue)
            maxValue = bar.Value;
      }
```

```
        foreach (KeyValuePair<string, int> bar in bars)
        {
            string key = bar.Key;
            int value = bar.Value;

            if (value >= minValue)
            {
                string label;
                if (key.Length < maxLabelWidth)
                    label = key.PadLeft(maxLabelWidth);
                else
                    label = key.Substring(0, maxLabelWidth);

                int barSize = (int)(((double)value / maxValue) * maxBarWidth);
                string barStars = "".PadRight(barSize, '*');

                s += label + " |" + barStars + " " + value + "\n";
            }
        }

        string axis = blankLabel + " +".PadRight(maxBarWidth + 2, '-') + "\n";    //TODO: Why is +2 is needed?
        s += axis;

        return s;
    }
}
```