Exercise Help Sheet.

Exercise 1
This is a free for all, draw whatever they want.

Exercise 2

```
var enter_update_basic = (function () {

   return {
       create_update_canvas: function (placement, data, options) {
           var svg = d3.select(placement).append("svg")
               .attr("width", options.width)
               .attr("height", options.height)
               .append("g");

           enter_update_basic.update_canvas(placement, data)
       },

       update_canvas: function (placement,data) {
           var svg = d3.select(placement + " svg g");
           var rect = svg.selectAll("rect")
               .data(data);

           rect.enter().append("rect")
               .style("fill", "#fff")
               .attr("height", 20)
               .attr("width", 0)
               .transition()
               .attr("width", 25);

           rect.attr("x", function (d) {
               return d.x;
           })
           .attr("y", function (d) {
               return d.y;
           });

           rect.exit().attr("width", 25).transition()
               .attr("width", 0)
               .remove();
       }
   }
})();

enter_update_basic.create_update_canvas(placement,
[{x:100, y:10}, {x:50, y:50}], {'width':100, 'height': 100});
```

Exercise 3
Add Scales

```
var enter_update_basic = (function () {

   return {
       create_update_canvas: function (placement, data, options) {
           var svg = d3.select(placement).append("svg")
               .attr("width", options.width)
               .attr("height", options.height)
               .append("g");
           enter_update_basic.update_canvas(placement, data)
       },

       update_canvas: function (placement,data) {
           var svg = d3.select(placement + " svg g");
           var rect = svg.selectAll("rect")
               .data(data);

           var xScale = d3.scale.linear()
               .domain(d3.extent(data, function (d) {
                   return d.x;
               }))
               .range([0, width - margin.left - margin.right]);

            var yScale = d3.scale.linear()
               .domain(d3.extent(data, function (d) {
                   return d.y;
               }))
               .range([height - margin.top - margin.bottom, 0]);

           rect.enter().append("rect")
               .style("fill", "#fff")
               .attr("height", 20)
               .attr("width", 0)
               .transition()
               .attr("width", 25);

           rect.attr("x", function (d) {
               return xScale(d.x);
           })
           .attr("y", function (d) {
               return yScale(d.y);
           });

           rect.exit().attr("width", 25).transition()
               .attr("width", 0)
               .remove();
       }
   }
})();
```

Exercise 4
Add Axes

```
var enter_update_basic = (function () {

   return {
       create_update_canvas: function (placement, data, options) {
           var svg = d3.select(placement).append("svg")
               .attr("width", options.width)
               .attr("height", options.height)
               .append("g");
           enter_update_basic.update_canvas(placement, data)
       },

       update_canvas: function (placement,data) {
           var svg = d3.select(placement + " svg g");
           var rect = svg.selectAll("rect")
               .data(data);

           var xScale = d3.scale.linear()
               .domain(d3.extent(data, function (d) {
                   return d.x;
               }))
               .range([0, width - margin.left - margin.right]);

            var yScale = d3.scale.linear()
               .domain(d3.extent(data, function (d) {
                   return d.y;
               }))
               .range([height - margin.top - margin.bottom, 0]);

           var xAxis = d3.svg.axis()
               .scale(xScale)
               .orient("bottom")
               .tickPadding(4);

          var yAxis = d3.svg.axis()
               .scale(yScale)
               .orient("left")
               .tickPadding(10);

           rect.enter().append("rect")
               .style("fill", "#fff")
               .attr("height", 20)
               .attr("width", 0)
               .transition()
               .attr("width", 25);

           rect.attr("x", function (d) {
               return xScale(d.x);
            })
            .attr("y", function (d) {
                return yScale(d.y);
            });

            rect.exit().attr("width", 25).transition()
                .attr("width", 0)
                .remove();

            svg.append("g")
                .attr("class", "x axis")
                .attr("transform", "translate(0," + yScale.range()[0] + ")")
                .call(xAxis);

            svg.append("g")
                .attr("class", "y axis")
                .call(yAxis);

        }
    }
})();
```

Exercise 5
Loading data from a JSON File.

```
var chocolates = (function () {

   return {

      loadAndDisplayData: function(placement, url, width, height) {

            d3.select(placement).html("");

            d3.json(url, function (data) {
                  data = data.chocolates;

                  var svg = d3.select(placement)
                        .append("svg").attr("width", width)
                        .attr("height", height).append("g")
                        .attr("transform",
                        "translate(" + margins.left + "," + margins.top + ")");

                  var xScale = d3.scale.linear()
                        .domain(d3.extent(data, function (d) {
                              return d.price;
                        }))
                        .range([0, width - margins.left - margins.right]);

                  var yScale = d3.scale.linear()
                        .domain(d3.extent(data, function (d) {
                              return d.rating;
                       }))
                       .range([height - margins.top - margins.bottom, 0]);

                  // create a default colour scale for our nodes.
                  var colors = d3.scale.category10();

                  var xAxis = d3.svg.axis()
                        .scale(xScale).orient("bottom").tickPadding(2);
                  var yAxis = d3.svg.axis().scale(yScale)
                        .orient("left").tickPadding(2);

                  svg.append("g")
                        .attr("class", "x axis")
                        .attr("transform", "translate(0," + yScale.range()[0] + ")")
                        .call(xAxis);

                  svg.append("g").attr("class", "y axis").call(yAxis);

                  svg.append("text")
                        .attr("fill", "#414241").attr("text-anchor", "end")
                        .attr("x", width / 2)
```

```
                            .attr("y", height - 35).text("Price in pence (£)");

                    var chocolate = svg.selectAll("g.node")
                            .data(data, function (d) { return d.name; });

                     // add your group item. This will be the container for
                     // the circle item and the text label.
                     var chocolateEnter = chocolate.enter().append("g")
                            .attr("class", "node")
                            .attr('transform', function (d) {
                                    return "translate(" + xScale(d.price) + "," +
                                            yScale(d.rating) + ")";
                     });

                     // add your circle to represent the record
                     chocolateEnter.append("circle")
                            .attr("r", 5).attr("class", "dot")
                            .style("fill", function (d) {
                                    return colors(d.manufacturer);
                     });

                    // add your label
                    chocolateEnter.append("text")
                            .style("text-anchor", "middle").attr("dy", -10)
                            .text(function (d) {
                                    return d.name;
                            });
        });
    }})()
```

Exercise 6

```
var chocolates = (function () {

   return {

      loadAndDisplayData: function(placement, url, width, height) {

            d3.select(placement).html("");

            d3.json(url, function (data) {
                  data = data.chocolates;

                  var svg = d3.select(placement)
                        .append("svg").attr("width", width)
                        .attr("height", height).append("g")
                        .attr("transform",
                        "translate(" + margins.left + "," + margins.top + ")");

                  var xScale = d3.scale.linear()
                        .domain(d3.extent(data, function (d) {
                              return d.price;
                        }))
                        .range([0, width - margins.left - margins.right]);

                  var yScale = d3.scale.linear()
                        .domain(d3.extent(data, function (d) {
                              return d.rating;
                       }))
                       .range([height - margins.top - margins.bottom, 0]);

                  // create a default colour scale for our nodes.
                  var colors = d3.scale.category10();

                  var xAxis = d3.svg.axis()
                        .scale(xScale).orient("bottom").tickPadding(2);
                  var yAxis = d3.svg.axis().scale(yScale)
                        .orient("left").tickPadding(2);

                  svg.append("g")
                        .attr("class", "x axis")
                        .attr("transform",
                              "translate(0," + yScale.range()[0] + ")")
                        .call(xAxis);

                  svg.append("g").attr("class", "y axis").call(yAxis);

                  svg.append("text")
                        .attr("fill", "#414241").attr("text-anchor", "end")
                        .attr("x", width / 2)
                        .attr("y", height - 35).text("Price in pence (£)");
```

```
                    var chocolate = svg.selectAll("g.node")
                          .data(data, function (d) { return d.name; });

                     // add your group item. This will be the container for
                     // the circle item and the text label.
                     var chocolateEnter = chocolate.enter().append("g")
                          .attr("class", "node")
                          .attr('transform', function (d) {
                                return "translate(" + xScale(d.price) + "," +
                                      yScale(d.rating) + ")";
                     });

                     // add your circle to represent the record
                     chocolateEnter.append("circle")
                          .attr("r", 5).attr("class", "dot")
                          .style("fill", function (d) {
                                return colors(d.manufacturer);
                     });

                    // add your label
                    chocolateEnter.append("text")
                          .style("text-anchor", "middle").attr("dy", -10)
                          .text(function (d) {
                                return d.name;
                          });

                    chocolateEnter.on("mouseover", function (d) {
                          d3.select(this).style("stroke-width", "1px")
                                .style("stroke", "white");
                    }).on("mouseout", function (d) {
                          d3.select(this).style("stroke", "none");
                    }).on("click", function (d) {
                          alert("Hi, you clicked on " + d.name);
                    })

  });
}})()
```

Exercise 7

Add zooming.

```
var chocolates = (function () {

   return {

      loadAndDisplayData: function(placement, url, width, height) {

            d3.select(placement).html("");

            d3.json(url, function (data) {
                  data = data.chocolates;

                  var svg = d3.select(placement)
                        .append("svg").attr("width", width)
                        .attr("height", height).append("g")
                        .attr("transform",
                        "translate(" + margins.left + "," + margins.top + ")");

                  var xScale = d3.scale.linear()
                        .domain(d3.extent(data, function (d) {
                              return d.price;
                        }))
                        .range([0, width - margins.left - margins.right]);

                  var yScale = d3.scale.linear()
                        .domain(d3.extent(data, function (d) {
                              return d.rating;
                       }))
                       .range([height - margins.top - margins.bottom, 0]);

                  var zoom = d3.behavior.zoom()
                     .x(xScale).y(yScale)
                     .scaleExtent([1, 5])
                     .on("zoom", function() {
                        d3.selectAll("g.x.axis").call(xAxis);
                        d3.selectAll("g.y.axis").call(yAxis);
                        svg.selectAll("g.node")
                        .attr("transform", function (d) {
                              return "translate(" + x(d.price) + "," + y(d.rating) +
                        ")scale(" + d3.event.scale + ")"
                     })
                  });

                  svg.call(zoom);

                  // add the rectangle so that zooming is captured across the plot.
                  svg.append('rect')
                     .attr('width', width)
                     .attr('height', height)
                     .attr('fill', 'rgba(1,1,1,0)');
```

```
// create a default colour scale for our nodes.
var colors = d3.scale.category10();

var xAxis = d3.svg.axis()
      .scale(xScale).orient("bottom").tickPadding(2);
var yAxis = d3.svg.axis().scale(yScale)
      .orient("left").tickPadding(2);

svg.append("g")
      .attr("class", "x axis")
      .attr("transform",
            "translate(0," + yScale.range()[0] + ")")
      .call(xAxis);

svg.append("g").attr("class", "y axis").call(yAxis);

svg.append("text")
      .attr("fill", "#414241").attr("text-anchor", "end")
      .attr("x", width / 2)
      .attr("y", height - 35).text("Price in pence (£)");

var chocolate = svg.selectAll("g.node")
      .data(data, function (d) { return d.name; });

 // add your group item. This will be the container for
 // the circle item and the text label.
 var chocolateEnter = chocolate.enter().append("g")
      .attr("class", "node")
      .attr('transform', function (d) {
            return "translate(" + xScale(d.price) + "," +
                  yScale(d.rating) + ")";
 });

 // add your circle to represent the record
 chocolateEnter.append("circle")
      .attr("r", 5).attr("class", "dot")
      .style("fill", function (d) {
            return colors(d.manufacturer);
 });

// add your label
chocolateEnter.append("text")
      .style("text-anchor", "middle").attr("dy", -10)
      .text(function (d) {
            return d.name;
      });

chocolateEnter.on("mouseover", function (d) {
      d3.select(this).style("stroke-width", "1px")
```

```
                               .style("stroke", "white");
                  }).on("mouseout", function (d) {
                        d3.select(this).style("stroke", "none");
                  }).on("click", function (d) {
                        alert("Hi, you clicked on " + d.name);
                  })
   });
}})()
```

Exercise 8

Add brushing.

```
var chocolates = (function () {

   return {

      loadAndDisplayData: function(placement, url, width, height) {

            d3.select(placement).html("");

            d3.json(url, function (data) {
                  data = data.chocolates;

                  var svg = d3.select(placement)
                        .append("svg").attr("width", width)
                        .attr("height", height).append("g")
                        .attr("transform",
                        "translate(" + margins.left + "," + margins.top + ")");

                  var xScale = d3.scale.linear()
                        .domain(d3.extent(data, function (d) {
                              return d.price;
                        }))
                        .range([0, width - margins.left - margins.right]);

                  var yScale = d3.scale.linear()
                        .domain(d3.extent(data, function (d) {
                              return d.rating;
                       }))
                       .range([height - margins.top - margins.bottom, 0]);

                  var zoom = d3.behavior.zoom()
                     .x(xScale).y(yScale)
                     .scaleExtent([1, 5])
                     .on("zoom", function() {
                        d3.selectAll("g.x.axis").call(xAxis);
                        d3.selectAll("g.y.axis").call(yAxis);
                        svg.selectAll("g.node")
                        .attr("transform", function (d) {
                              return "translate(" + x(d.price) + "," + y(d.rating) +
                        ")scale(" + d3.event.scale + ")"
                      })
                  });

                  svg.call(zoom);

                  // add the rectangle so that zooming is captured across the plot.
                  svg.append('rect')
                     .attr('width', width)
                     .attr('height', height)
                     .attr('fill', 'rgba(1,1,1,0)');
```

```
// create a default colour scale for our nodes.
var colors = d3.scale.category10();

var xAxis = d3.svg.axis()
      .scale(xScale).orient("bottom").tickPadding(2);
var yAxis = d3.svg.axis().scale(yScale)
      .orient("left").tickPadding(2);

svg.append("g")
      .attr("class", "x axis")
      .attr("transform",
            "translate(0," + yScale.range()[0] + ")")
      .call(xAxis);

svg.append("g").attr("class", "y axis").call(yAxis);

svg.append("text")
      .attr("fill", "#414241").attr("text-anchor", "end")
      .attr("x", width / 2)
      .attr("y", height - 35).text("Price in pence (£)");

var chocolate = svg.selectAll("g.node")
      .data(data, function (d) { return d.name; });

 // add your group item. This will be the container for
 // the circle item and the text label.
 var chocolateEnter = chocolate.enter().append("g")
      .attr("class", "node")
      .attr('transform', function (d) {
            return "translate(" + xScale(d.price) + "," +
                  yScale(d.rating) + ")";
 });

 // add your circle to represent the record
 chocolateEnter.append("circle")
      .attr("r", 5).attr("class", "dot")
      .style("fill", function (d) {
            return colors(d.manufacturer);
 });

// add your label
chocolateEnter.append("text")
      .style("text-anchor", "middle").attr("dy", -10)
      .text(function (d) {
            return d.name;
      });

chocolateEnter.on("mouseover", function (d) {
      d3.select(this).style("stroke-width", "1px")
```

```
                .style("stroke", "white");
            }).on("mouseout", function (d) {
                d3.select(this).style("stroke", "none");
            }).on("click", function (d) {
                alert("Hi, you clicked on " + d.name);
            });

            brush = d3.svg.brush()
              .x(xScale)
              .y(yScale)
              .on("brushstart", function () {
                console.log("Resetting selected var");
                selected = {};
              })
              .on("brush", function () {

                var extent = brush.extent();
                d3.selectAll("g.chocolatenode").select("circle").style("fill", function (d) {
                  d.selected = (d.x > x(extent[0][0]) && d.x < x(extent[1][0]))
                    && (d.y < y(extent[0][1]) && d.y > y(extent[1][1]));

                  if (d.selected) {
                    selected[d.name] = d;
                  }
                  return d.selected ? "#F15D2F" : colors(d.manufacturer);


                });

              })
              .on("brushend", function () {
                // do nothing
              });

            svg.append("g")
              .attr("class", "brush")
              .call(brush);

    });
})()
```