Use the `head` command on your three files again. This time, describe at least one potential problem with the data you see. Consider issues with missing values and bad data.

The bus table seems to have the most issues, as just in the first 5 lines, the first row has a placeholder for no phone number (-9999), and the next four rows all do not have actual location data, also using -9999 as placeholders for missing latitude and longitude. Similarly, inspection scores of -1 are given for a multitude of cases, seemingly for different reasons, which makes it ambiguous.

In the cell below, write the name of the restaurant with the lowest inspection scores ever. You can also head to yelp.com and look up the reviews page for this restaurant. Feel free to add anything interesting you want to share.

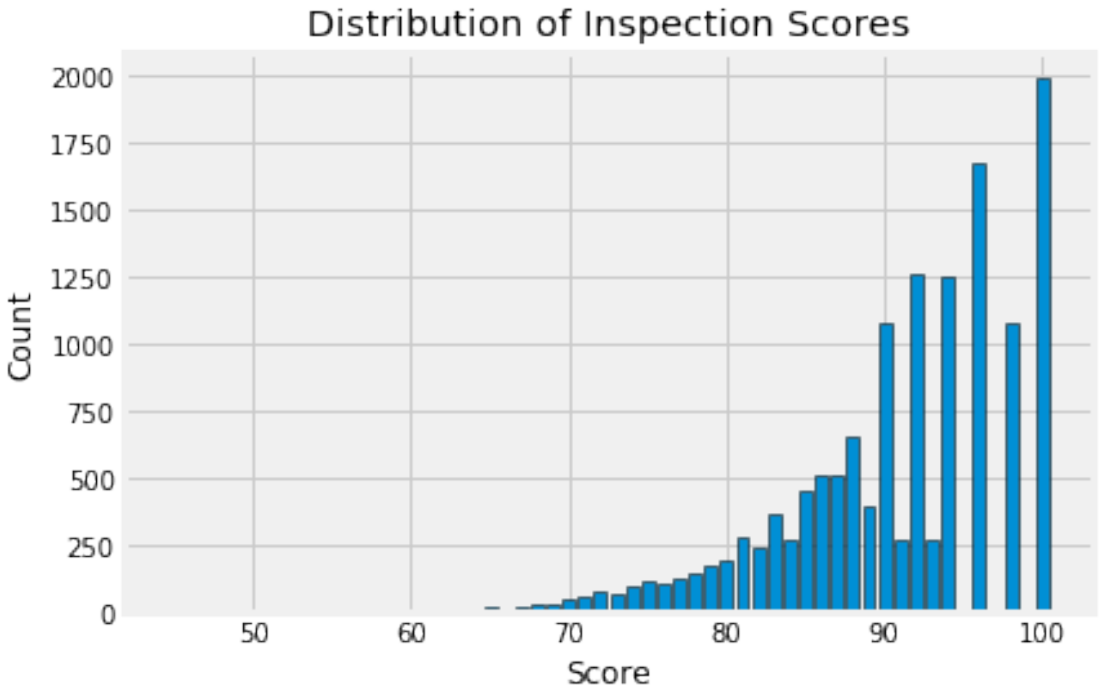
Lowest score on record: Lollipot

Unfortunately (or fortunately), after looking it up on Yelp it appears that Lollipot is closed, likely due to COVID.

0.1 Question 6a

Let's look at the distribution of inspection scores. As we saw before when we called head on this data frame, inspection scores appear to be integer values. The discreteness of this variable means that we can use a barplot to visualize the distribution of the inspection score. Make a bar plot of the counts of the number of inspections receiving each score.

It should look like the image below. It does not need to look exactly the same (e.g., no grid), but make sure that all labels and axes are correct.

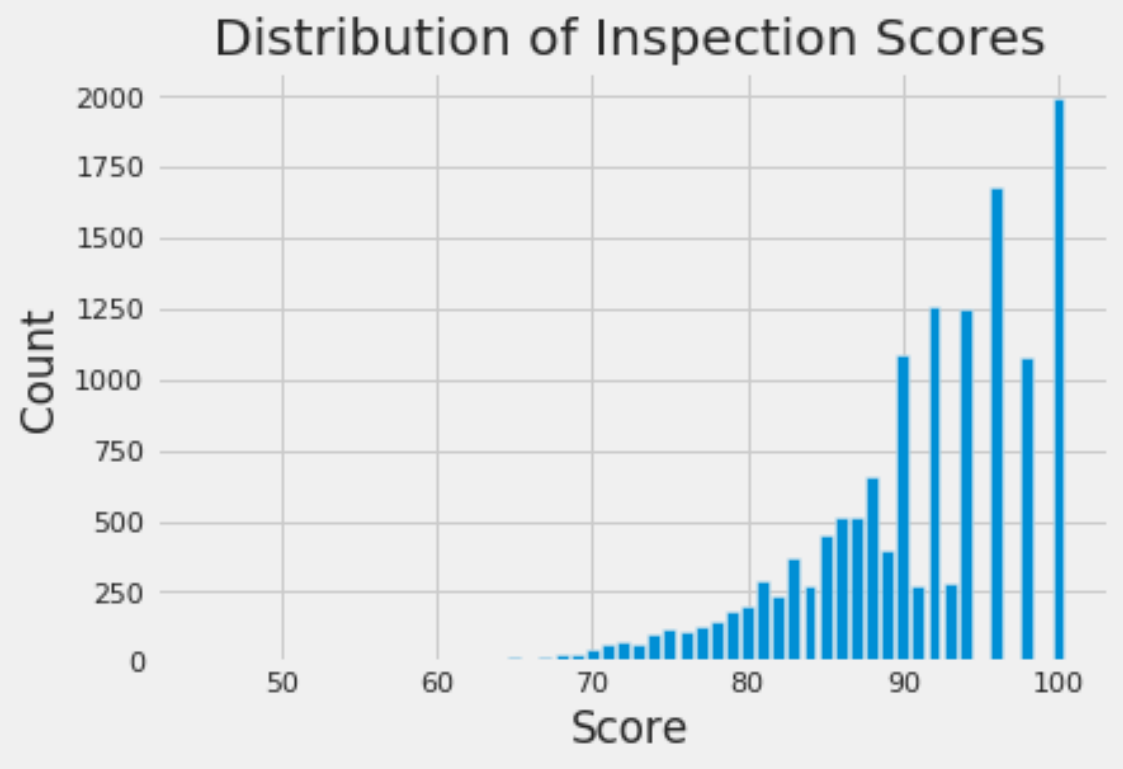


You might find this matplotlib.pyplot tutorial useful. Key syntax that you'll need:

```
plt.bar
plt.xlabel
plt.ylabel
plt.title
```

Note: If you want to use another plotting library for your plots (e.g. plotly, sns) you are welcome to use that library instead so long as it works on DataHub. If you use seaborn sns.countplot(), you may need to manually set what to display on xticks.

In [79]:
```
plt.bar(ins["score"].value_counts().index, ins["score"].value_counts())
plt.xlabel("Score")
plt.ylabel("Count")
plt.title("Distribution of Inspection Scores");
```



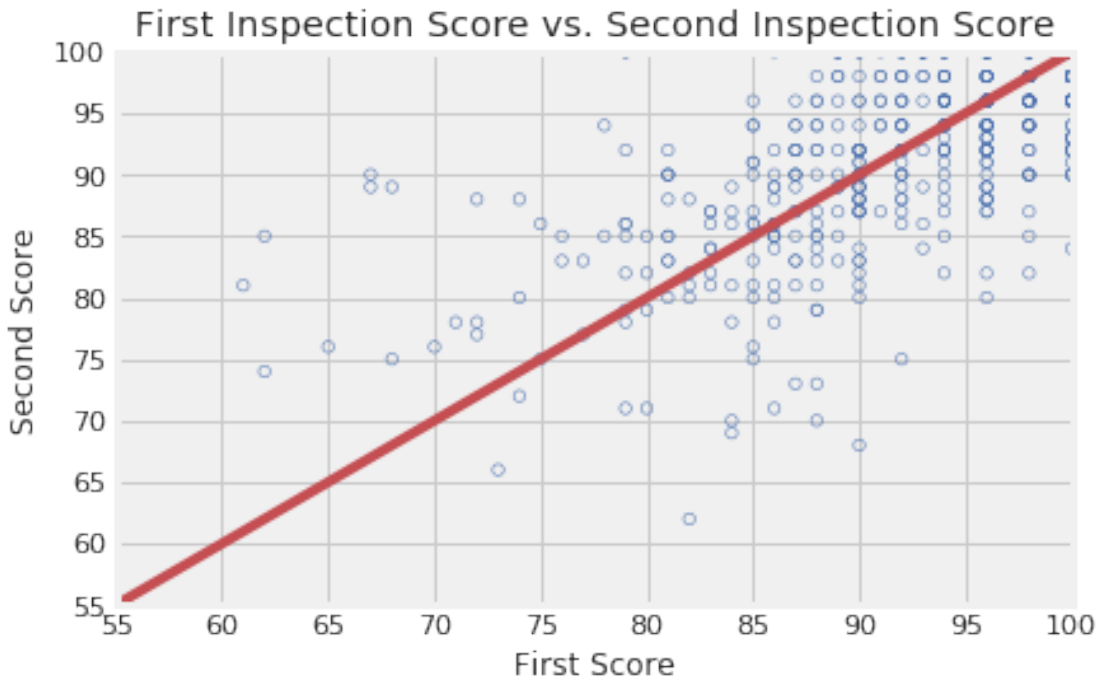
0.1.1 Question 6b

Describe the qualities of the distribution of the inspections scores based on your bar plot. Consider the mode(s), symmetry, tails, gaps, and anomalous values. Are there any unusual features of this distribution? What do your observations imply about the scores?

The distribution is highly left-skewed, so it's not symmetric or normal. Most restaurants received scores at or above 90 points, with the mode at 100, also the maximum value. There are scores that are not assigned at all, which could be reflective of the method of scoring or tabulation.

The scoring suggests that most restaurants in San Francisco are sanitary, given that the inspection score reflects food safety standards.

Now, create your scatter plot in the cell below. It does not need to look exactly the same (e.g., no grid) as the sample below, but make sure that all labels, axes and data itself are correct.



Key pieces of syntax you'll need:

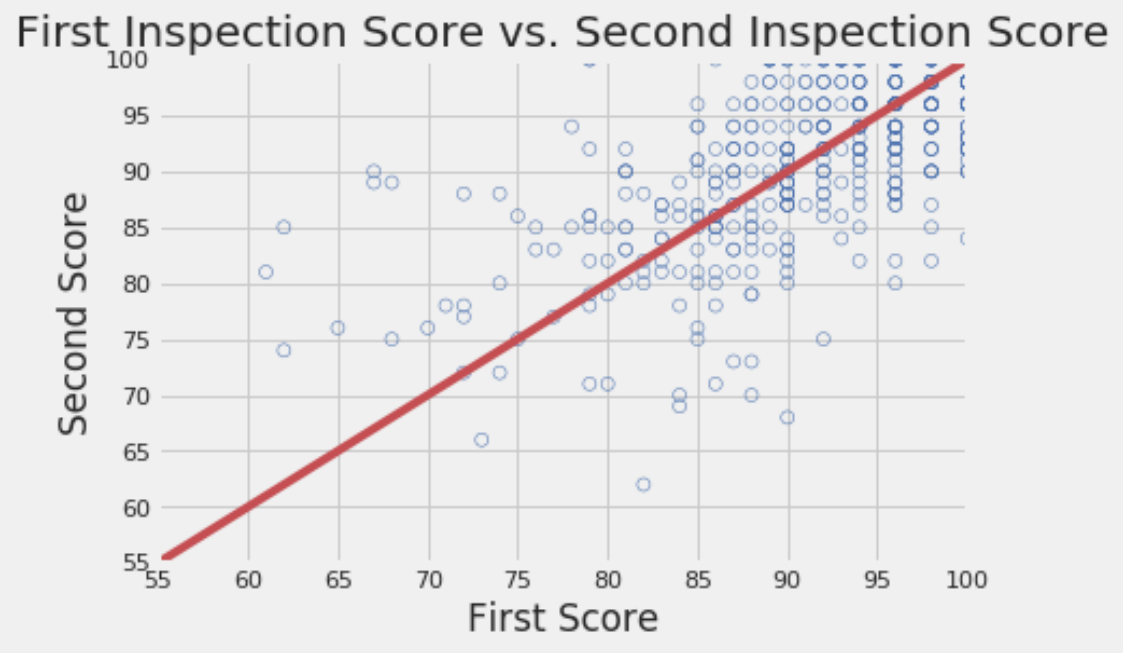
`plt.scatter` plots a set of points. Use `facecolors='none'` and `edgecolors=b` to make circle markers with blue borders.

`plt.plot` for the reference line.

`plt.xlabel`, `plt.ylabel`, `plt.axis`, and `plt.title`.

Hint: You may find it convenient to use the `zip()` function to unzip scores in the list.

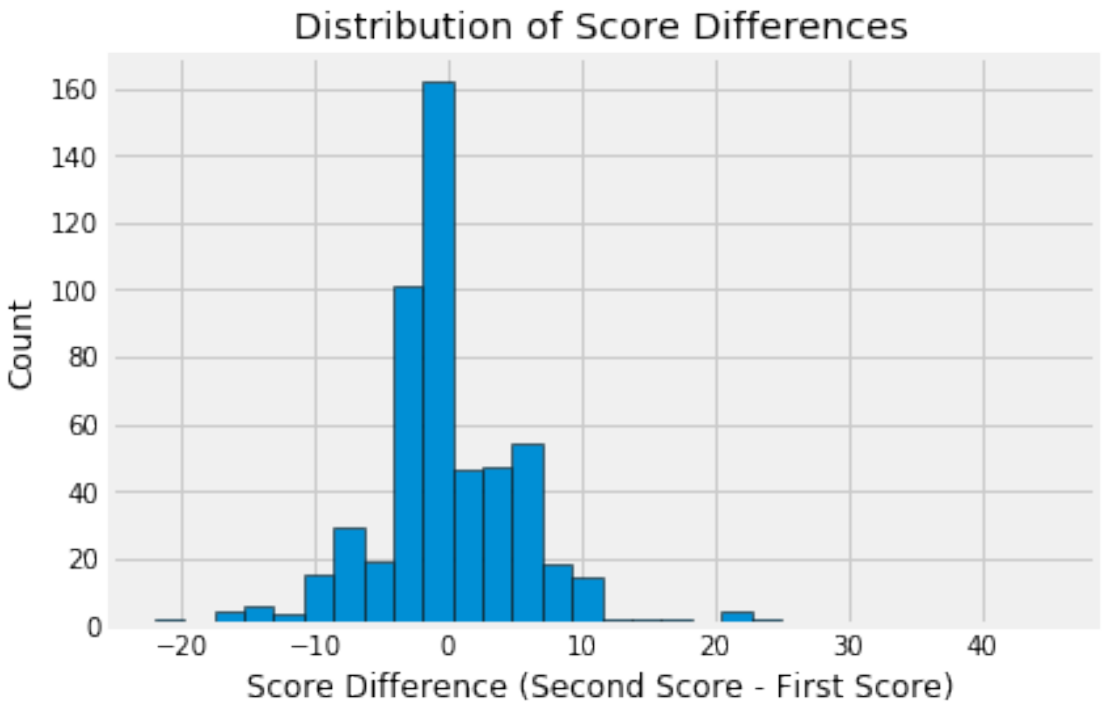
```
In [88]: first, second = zip(*scores_pairs_by_business['score_pair'])
         plt.scatter(first, second, facecolors='none', edgecolors="b")
         plt.xlabel("First Score")
         plt.ylabel("Second Score")
         plt.title("First Inspection Score vs. Second Inspection Score")
         plt.plot([55,100], [55,100], color='r')
         plt.xlim(55, 100)
         plt.ylim(55, 100);
```


First Inspection Score vs. Second Inspection Score
Second Score
First Score
100
95
90
85
80
75
70
65
60
55
55
60
65
70
75
80
85
90
95
100

0.1.2 Question 7d

Another way to compare the scores from the two inspections is to examine the difference in scores. Subtract the first score from the second in `scores_pairs_by_business`. Make a histogram of these differences in the scores. We might expect these differences to be positive, indicating an improvement from the first to the second inspection.

The histogram should look like this:

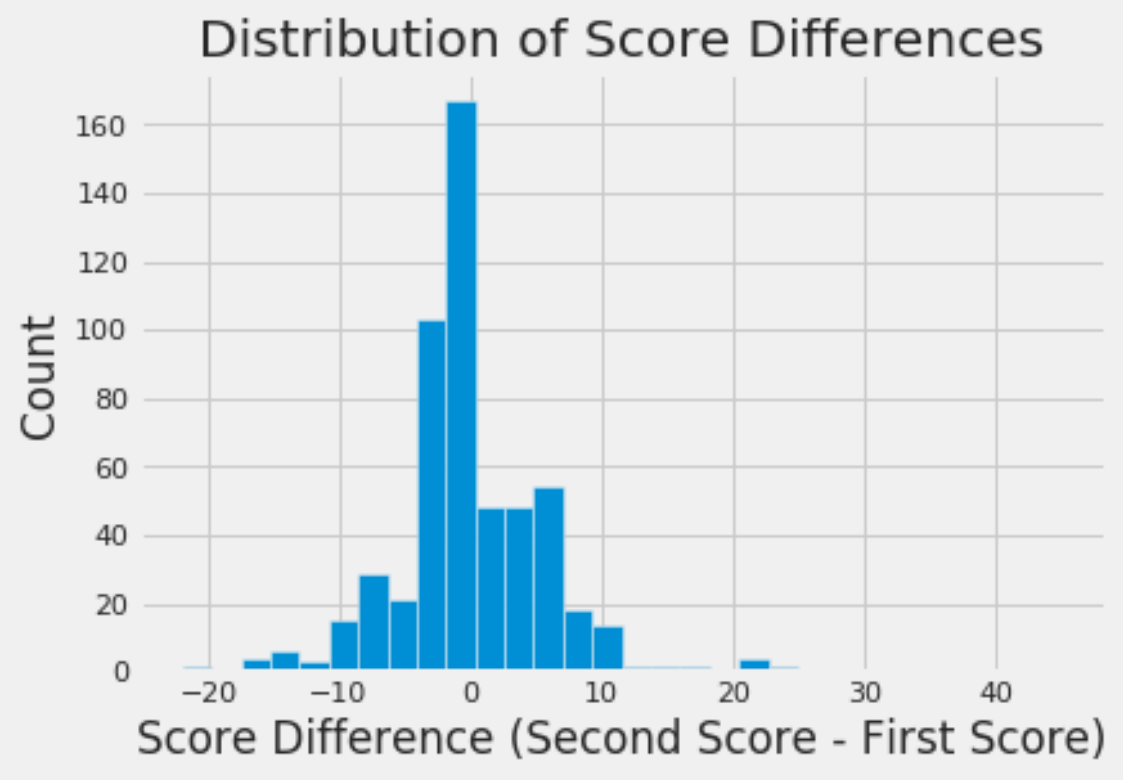


Hint: Use `second_score` and `first_score` created in the scatter plot code above.

Hint: Convert the scores into numpy arrays to make them easier to deal with.

Hint: Use `plt.hist()` Try changing the number of bins when you call `plt.hist()`.

```
In [89]: diff = np.array(second) - np.array(first)
         plt.hist(diff, bins=30)
         plt.xlabel("Score Difference (Second Score - First Score)")
         plt.ylabel("Count")
         plt.title("Distribution of Score Differences");
```


Distribution of Score Differences
Count
160
140
120
100
80
60
40
20
0
−20
−10
0
10
20
30
40
Score Difference (Second Score - First Score)

0.1.3 Question 7e

If restaurants' scores tend to improve from the first to the second inspection, what do you expect to see in the scatter plot that you made in question 7c? What do you oberve from the plot? Are your observations consistent with your expectations?

Hint: What does the slope represent?

The red reference line represents scoring exactly the same on the first and second inspection (no change or improvement). Any values below the line are cases where restaurants scored lower on the second inspection, and values above the line are where restaurants scored higher on the second inspection. If we expect that second inspections tend to have higher scores than first inspections, we would expect to see more values above the red line.

Instead, the graph does not suggest a clear improvement as the scattering is fairly even around the line. This means that our observations are not consistent with the expectation/hypothesis.

0.1.4 Question 7f

If a restaurant's score improves from the first to the second inspection, how would this be reflected in the histogram of the difference in the scores that you made in question 7d? What do you oberve from the plot? Are your observations consistent with your expectations? Explain your observations in the language of Statistics: for instance, the center, the spread, the deviation etc.

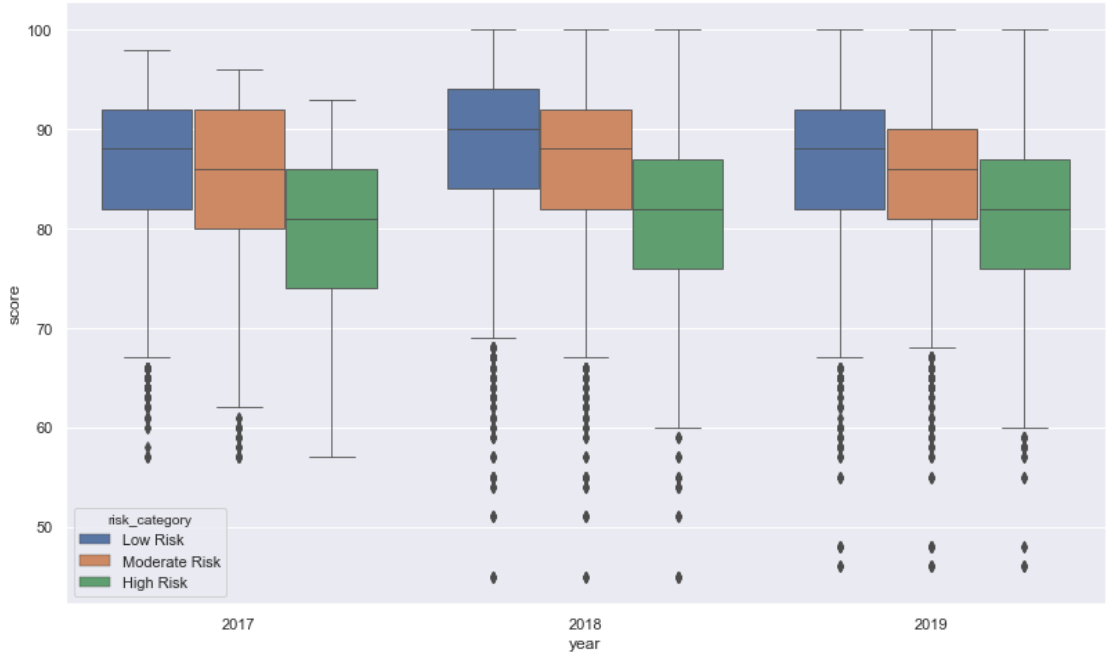
In the histogram representation, if restaurants tended to improve their inspection scores, we would expect the distribution to be left-skewed to some degree. The more strongly restaurants overall improved, the more left-skewed it would appear.

Instead, from the observed histogram the distribution is roughly normal, with the center roughly at 0, and a roughly estimated SD around 5 points. This histogram and these values strongly suggest that most restaurants do not change their inspection scores by much on a second pass.

0.1.5 Question 7g

To wrap up our analysis of the restaurant ratings over time, one final metric we will be looking at is the distribution of restaurant scores over time. Create a side-by-side boxplot that shows the distribution of these scores for each different risk category from 2017 to 2019. Use a figure size of at least 12 by 8.

The boxplot should look similar to the sample below. Make sure the boxes are in the correct order!



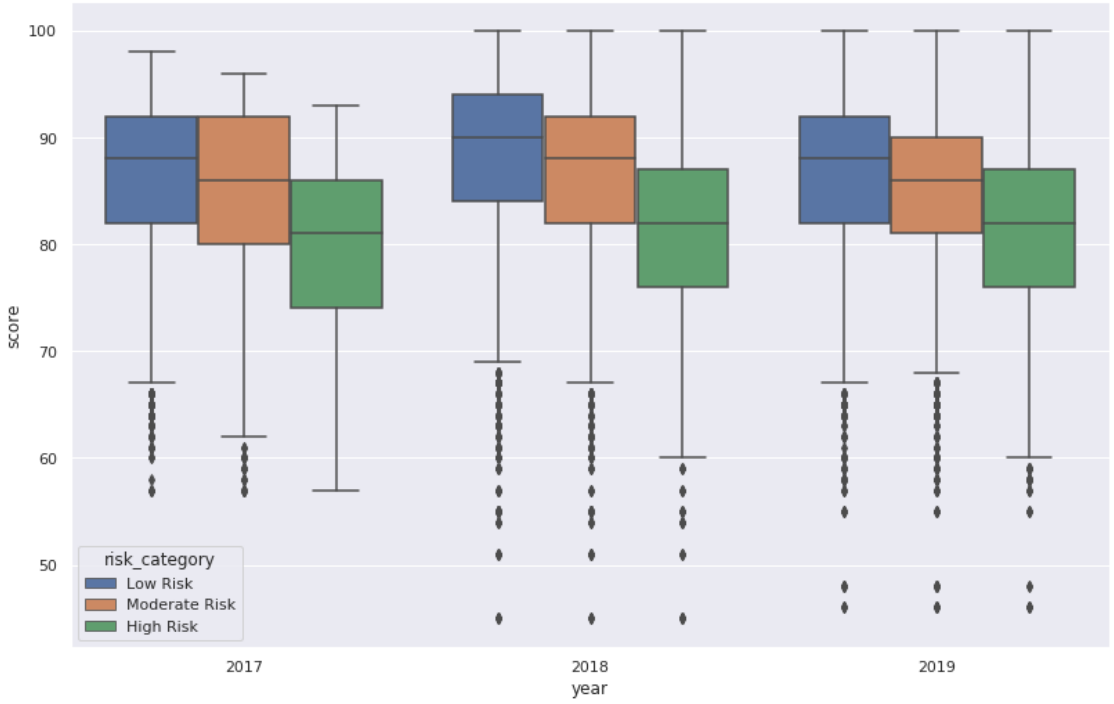
Hint: Use `sns.boxplot()`. Try taking a look at the first several parameters. The documentation is linked here!

Hint: Use `plt.figure()` to adjust the figure size of your plot.

```
In [90]: # Do not modify this line
         sns.set()

         ins_temp = ins[ins["year"] >= 2017]
         ins_vio = pd.merge(ins_temp, ins2vio, how="left", on="iid")
         ins_vio = pd.merge(ins_vio, vio, how="left", on="vid")
         plt.figure(figsize = (12,8))
         #plt.legend(loc = "best")
         sns.boxplot(x=ins_vio["year"], y=ins_vio["score"], hue=ins_vio["risk_category"], hue_order=("L
```

Out[90]: <matplotlib.axes._subplots.AxesSubplot at 0x7fba047908e0>



---

1 8: Open Ended Question

1.1 Question 8a

1.1.1 Compute Something Interesting

Play with the data and try to compute something interesting about the data. Please try to use at least one of groupby, pivot, or merge (or all of the above).

Please show your work in the cell below and describe in words what you found in the same cell. This question will be graded leniently but good solutions may be used to create future homework problems.

1.1.2 Grading

Since the question is more open ended, we will have a more relaxed rubric, classifying your answers into the following three categories:

- **Great** (4 points): Uses a combination of pandas operations (such as groupby, pivot, merge) to answer a relevant question about the data. The text description provides a reasonable interpretation of the result.
- **Passing** (1-3 points): Computation is flawed or very simple. The text description is incomplete but makes some sense.
- **Unsatisfactory** (0 points): No computation is performed, or a computation with completely wrong results.

Please have both your code and your explanation in the same one cell below. Any work in any other cell will not be graded.

In [91]:
```
#YOUR CODE HERE

food_trucks = bus[bus["postal5"].isnull()]
trad_rest = bus[bus["postal5"].notnull()]

all_ins_vio = pd.merge(ins, ins2vio, how="left", on="iid")
all_ins_vio = pd.merge(all_ins_vio, vio, how="left", on="vid")
ft_vio = pd.merge(food_trucks, all_ins_vio, on="bid")
trad_vio = pd.merge(trad_rest, all_ins_vio, on="bid")
```

```
most_common_ft_vios = ft_vio.groupby("description").count().sort_values("bid", ascending=False)
most_common_ft_vios = most_common_ft_vios.rename(columns={"bid": "count"})["count"]
most_common_trad_vios = trad_vio.groupby("description").count().sort_values("bid", ascending=Fa
most_common_trad_vios = most_common_trad_vios.rename(columns={"bid": "count"})["count"]

avg_ft_vios_per_loc = sum(most_common_ft_vios) / food_trucks.shape[0]
avg_trad_vios_per_loc = sum(most_common_trad_vios) / trad_rest.shape[0]

ft_vios_by_loc = ft_vio.groupby("bid").count().sort_values("name", ascending=False)
ft_vios_by_loc = ft_vios_by_loc.rename(columns={"name": "count"})["count"]
trad_vios_by_loc = trad_vio.groupby("bid").count().sort_values("name", ascending=False)
trad_vios_by_loc = trad_vios_by_loc.rename(columns={"name": "count"})["count"]

fts_over_avg = ft_vios_by_loc[(ft_vios_by_loc - avg_ft_vios_per_loc) > 0]
trad_over_avg = trad_vios_by_loc[(trad_vios_by_loc - avg_trad_vios_per_loc) > 0]
worst_fts = len(fts_over_avg) / food_trucks.shape[0]
worst_trad = len(trad_over_avg) / trad_rest.shape[0]

ft_severity = ft_vio.replace({"risk_category": {"Low Risk": 1, "Moderate Risk": 2, "High Risk"
trad_severity = trad_vio.replace({"risk_category": {"Low Risk": 1, "Moderate Risk": 2, "High Ri
avg_ft_severity = np.sum(ft_severity["risk_category"]) / ft_severity["risk_category"].count()
avg_trad_severity = np.sum(trad_severity["risk_category"]) / trad_severity["risk_category"].cou

summary_df = pd.DataFrame({"type" : ["food trucks", "traditional restaurants"],
                           "avg # vios per loc" : [avg_ft_vios_per_loc, avg_trad_vios_per_loc]
                           "% offenders over avg" : [worst_fts, worst_trad],
                           "severity of avg vio" : [avg_ft_severity, avg_trad_severity]})
summary_df = summary_df.round(3)
summary_df

# I didn't understand the reference in Q3e until I checked Piazza and realized it was describi
# Given a way to look into these, I decided to try to see if anything interesting would emerge
# for non-permanent restaurants and possibly comparing them to brick and mortar establishments

# I began by separating the bus table to food_truck and trad_rest, and then took the median of
# scores with all_scores. Then I merged the two split business tables with their median scores
# that food trucks had a slightly higher median inspection score than restaurants (94 to 90),
# simplistic and not very informative, as there was no other way to contextualize the data. So
# against the violations table to see if food trucks had different types of inspections issues
# restaurants. The two groups seemed to have different violations, but more interestingly, tra
# averaged more violations per location (6.1) compared to food trucks (3.9). This seemed like
# worth digging into, and I expanded the comparisons by trying to find the number of outsize o
# above average violations, and also the severity of the violations recorded by remapping Low
# and High -> 3 to give a numerical idea of the average violations issued. There, the two grou
```

Out[91]:

	type	avg # vios per loc	% offenders over avg	severity of avg vio
0	food trucks	3.905	0.502	1.629
1	traditional restaurants	6.063	0.397	1.668

1.1.3 Grading

Since the question is more open ended, we will have a more relaxed rubric, classifying your answers into the following three categories:

- **Great** (4 points): The chart is well designed, and the data computation is correct. The text written articulates a reasonable metric and correctly describes the relevant insight and answer to the question you are interested in.
- **Passing** (1-3 points): A chart is produced but with some flaws such as bad encoding. The text written is incomplete but makes some sense.
- **Unsatisfactory** (0 points): No chart is created, or a chart with completely wrong results.

We will lean towards being generous with the grading. We might also either discuss in discussion or post on Piazza some examplar analysis you have done (with your permission)!

You should have the following in your answers: * a few visualizations; Please limit your visualizations to 5 plots. * a few sentences (not too long please!)

Please note that you will only receive support in OH and Piazza for Matplotlib and seaborn questions. However, you may use some other Python libraries to help you create you visualizations. If you do so, make sure it is compatible with the PDF export (e.g., Plotly does not create PDFs properly, which we need for Gradescope).

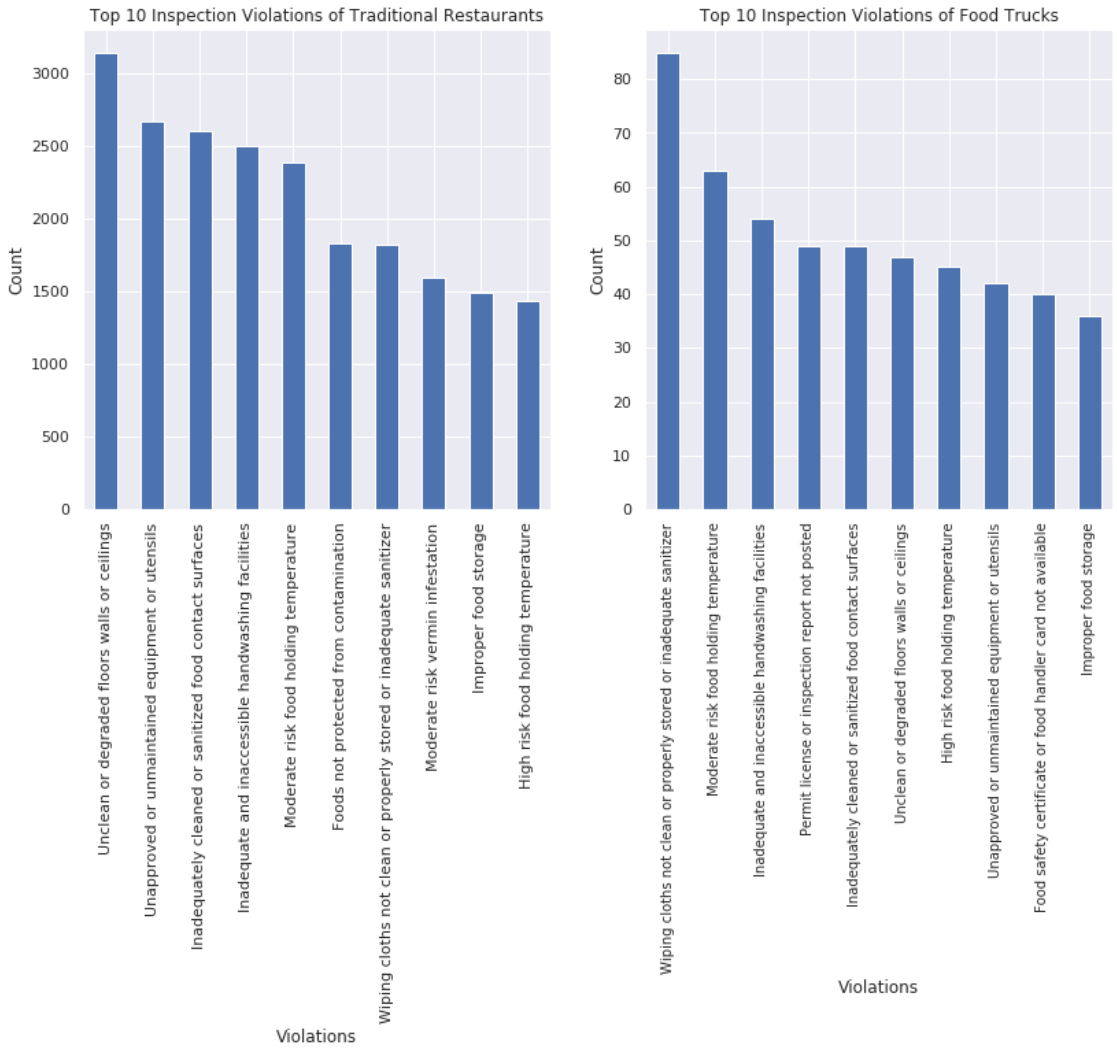
In [93]:
```
# YOUR DATA PROCESSING AND PLOTTING HERE

fig, axes = plt.subplots(nrows=1, ncols=2, figsize=(12,6))

top10_trad = most_common_trad_vios.head(10)
top10_trad.plot(ax=axes[0], kind='bar')
axes[0].set_xlabel("Violations")
axes[0].set_ylabel("Count")
axes[0].set_title("Top 10 Inspection Violations of Traditional Restaurants")

top10_ft = most_common_ft_vios.head(10)
top10_ft.plot(ax=axes[1], kind='bar')
axes[1].set_xlabel("Violations")
axes[1].set_ylabel("Count")
axes[1].set_title("Top 10 Inspection Violations of Food Trucks")
plt.xticks(fontsize='small', rotation=90);

# The initial chart is a simple comparison of the most common violations cited for each group.
```



In [94]:
```
# This helps to understand the types of issues each restaurant format encounters, as tradition
# seem to encounter issues with maintenance or sanitation, while food trucks seem to encounter
# related issues (not being visible or available).
```

In [95]:
```
vis_ft = ft_severity.groupby("risk_category").count().rename(columns={"bid": "count"})["count"]
vis_trad = trad_severity.groupby("risk_category").count().rename(columns={"bid": "count"})["cou

fig, axes = plt.subplots(nrows=1, ncols=2, figsize=(12,6))

vis_ft.plot(ax=axes[0], kind='bar')
axes[0].set_xlabel("Risk Category")
axes[0].set_ylabel("Count")
axes[0].set_title("Violation Severity of Traditional Restaurants")
```

```
vis_trad.plot(ax=axes[1], kind='bar')
axes[1].set_xlabel("Risk Category")
axes[1].set_ylabel("Count")
axes[1].set_title("Violation Severity of Food Trucks")
plt.xticks(fontsize='small', rotation=90);
```

In [96]:
```
# Earlier in Q8a it was determined that the average violation was of roughly similar severity
# trucks (1.63) and traditional restaurants (1.67), but a look at the visualization suggests t
# severe infractions (High and Moderate Risk) as a proportion with traditional restaurants, th
# is not significantly large.

# It's a common view to think of food trucks as less clean than traditional restaurants, but f
# available, I think we are able to show and prove that they are at least as clean as traditio
# if not even a little better as some of the most common violations are unrelated to food safe
```