Final Exam, Coding

DUE DATE and SUBMISSION INSTRUCTIONS:

This exam is due on Gradescope AND Canvas by **5:59PM on Thursday, December 8th**.

- Upload a PDF of this notebook (with all cells run and all LaTex fully rendered) to Gradescope by **5:59PM on Thursday, December 8th**.
- AND upload this Jupyter notebook to Canvas by **5:59PM on Thursday, December 8th**.

Late Submittals: **Gradescope and Canvas will NOT accept any submissions after 5:59PM. No late exams will be accepted**.

FORMAT:

- Your solutions to theoretical questions should be done in Markdown and LaTex directly below the associated question.
- Your solutions to computational questions should include any specified Python code and results as well as written commentary on your conclusions.

HERE ARE THE RULES:

1. All work, code and analysis, must be your own. It really is that simple.
2. Only work uploaded prior to the deadline will be considered.
3. You MAY use your course notes, posted lecture slides, textbooks, in-class notebooks, and homework solutions as resources. You may also search online for answers to general knowledge questions like the form of a probability distribution function or how to perform a particular operation in Python/Pandas.
4. You may **NOT** copy-paste solutions *from anywhere*.
5. You may **NOT** post to message boards or other online resources asking for help.
6. You may **NOT** collaborate with classmates or anyone else.
7. This is meant to be like a coding portion of your final exam. So, the instructional team will be much less helpful than we typically are with homework. For example, we will not check answers, help debug your code, and so on.
8. If you have a question for us, post it as a **PRIVATE** message on Piazza. If we decide that the question is appropriate for the entire class, then we will add it to a midterm clarifications thread.
9. If something is left open-ended, it is because we want to see how you approach the kinds of problems you will encounter in the wild, where it will not always be clear what sort of tests/methods should be applied. Feel free to ask clarifying questions though.
10. Please stick to software used/taught in this class: Pandas, NumPy, Python 3.9, etc.

Violation of the above rules will result in an immediate academic sanction (you will receive an F in the course), and a trip to the Honor Code Council.

By submitting this assignment, you agree to abide by the rules given above.

Name: CJ Kennedy

NOTES:

- By the due date, turn in anything you have, there are no extensions, and some credit is better than no credit. **DO NOT attempt your first upload at 5:00 PM** as you may encounter slow internet speeds or outages and again you will miss the upload and hence the entire exam.
- Do **NOT** load or use any Python packages that are not available in Anaconda 3.9.
- This should go without saying, but... For any question that asks you to calculate something, you **must show all work to receive credit**. Sparse or nonexistent work will receive sparse or nonexistent credit.

On this exam we are looking for HOW you answer questions and not necessarily WHAT you put for an answer.

Tell the grader where to look for your answer and which question the work goes with.

Be thorough, but be concise.

Use full sentences and proper grammar and correct spelling.

Comment your code.

Demonstrate BOTH what you know about the question AND the coding you used to answer the question

---

Example question

What values of x solve the equation: $x^2 - 5x = -6$?

Solve analytically and then create a plot supporting your answer.

This correct answer would be worth zero points

$x = 2$ and $x = 3$

This answer would be worth full points :

Solution to Example Question:

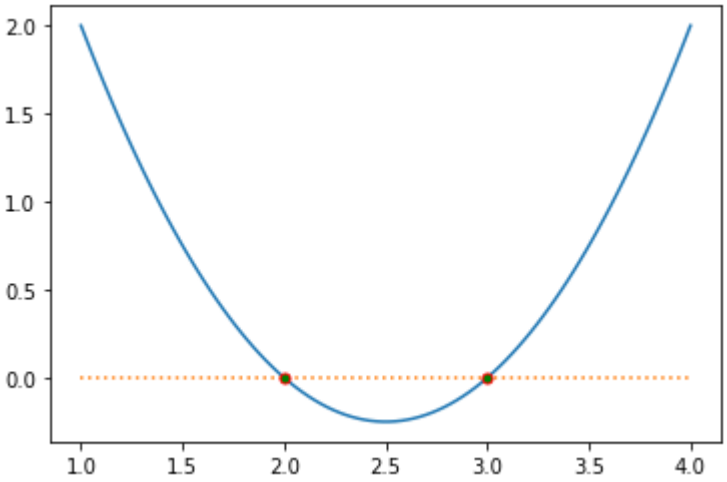
$x^2 - 5x = -6$ can be factored after -6 is added to both sides.

$(x - 2)(x - 3) = 0$. The zero product property reveals that $\boxed{x = 2}$ or $\boxed{x = 3}$.

This equation has two solutions as its graph crosses the x-axis twice as seen below.

In [1]:
```
# AND HERE IS MY COMMENTED CODE TO GO WITH THE EXAMPLE QUESTION #1:

import numpy
import matplotlib.pylab as draw
%matplotlib inline
# matplotlib.pylab is used for plotting. 'inline' ensures the graph opens in Jupyter.

x = numpy.linspace(1,4,100)
# 'linspace' returns evenly spaced numbers over a specified interval

y = x**2-5*x+6
z = 0*x
draw.plot(x,y)
# This is the parabola

draw.plot(x,z, ':')
# This is the x-axis.

draw.plot(2,0, marker="o", markersize=5, markeredgecolor="red", markerfacecolor="greer
draw.plot(3,0, marker="o", markersize=5, markeredgecolor="red", markerfacecolor="greer
# These last two 'draw' commands show the intersection of the parabola and the x-axis.
# The intersections are the sought after solutions. See graph below.
```

Out[1]:
```
[<matplotlib.lines.Line2D at 0x7fb1ad67fa60>]
```



Begin Final Exam (Coding) Here

Here are some imports that were used in the construction of the answer key. DO NOT change the `random.seed()` value.

In [2]:
```
import numpy as np
from scipy import stats
import statsmodels.api as sm
import pandas as pd
import matplotlib.pylab as plt
from scipy.stats import skew, kurtosis, norm
import seaborn
%matplotlib inline
from sklearn.linear_model import LinearRegression
from statsmodels.stats.multicomp import MultiComparison
almostblack = np.array([10,10,10])/255
mycolor = np.array([140,180,10])/255
import random
random.seed(8675309)
```

Note Regarding Assessing Normality:

You are asked to comment on the distribution of various datasets in this assignment. Also, you may, or may not, want to make decisions based on the normality of your data distribution.

If it is helpful, you might consider using:

`from scipy.stats import skew, kurtosis`

You can verify how 'normal' your Pandas data set (named dfData) is by doing something like this:

`print(skew(dfData))`

`print(kurtosis(dfData))`

The general rules you can use when assessing normality:

If the skewness is between -0.5 and 0.5, the data are fairly symmetrical.

If the skewness is between -1 and – 0.5 or between 0.5 and 1, the data are moderately skewed.

If the skewness is less than -1 or greater than 1, the data are highly skewed.

An excess kurtosis above 0 indicates the tails are heavier than the normal distribution.

An excess kurtosis below 0 indicates the tails are lighter than the normal distribution.

An excess kurtosis value of 1 and above or -1 and below represents a sizable departure from normality.

Commenting on how 'normal' your distributions are in this exam by using `skew` and `kurtosis` values can only gain you more points for your description.

Problem 1

For this problem you are welcome to use any built-in methods/functions we have used in class. If a built-in function/method outputs summary statistics or other useful output variables you can use that output directly when answering any questions below UNLESS otherwise specified in the problem (i.e. unless we tell you in the problem to write code to calculate a particular statistic from scratch based on its definition, you can just get that statistic/info from the output of a built-in method/function).

Your friend owns a franchise business that sells and rents stand-up paddleboards.
They have expanded their business and opened up shops in 95 different towns across the U.S.
They have collected data from each of their shops and have asked you to analyze the data. The data is in the file `paddle.csv`

Import the data set `paddle.csv`, and take a look at it.

In [3]:
```
dfSales = pd.read_csv("paddle.csv")
dfSales
```

Out[3]:

	lakes	pop	advertising	competitors	sales
0	1	63	4.3	1	14.1
1	1	25	10.8	2	19.0
2	1	40	11.8	2	19.8
3	1	89	9.6	0	19.9
4	1	33	6.4	2	22.4
...	...	...	...	...	...
90	8	96	3.4	8	11.8
91	8	37	3.6	9	12.7
92	8	75	7.5	9	17.2
93	8	31	11.5	9	19.5
94	8	39	14.3	8	21.3

95 rows × 5 columns

Data Description:

The data file contains the following data for each of the 95 shops:

`lakes` - The number of lakes or reservoirs within a 50 mile radius of the shop.

`pop` - Population density of the town where the shop is located (people per square mile).

`advertising` - The shop's annual amount spent on advertising/promotions (in thousands of dollars).

`competitors` - The number of competing stores within a 50 mile radius of the shop.

`sales` - The shop's annual net sales revenue (in thousands of dollars).

Single-Predictor Regression

[1] Part A) A naive model

Your friend says that since stand-up paddleboarding is a water activity, they would expect annual sales revenue to be dependent on the number of lakes/reservoirs in the region near a given shop.

Your task:

- ***(i) (2 points): Make a simple linear regression (SLR) model with `sales` as the response, predicted by the variable `lakes`.***
- ***Then write the equation for the model in a markdown cell.***

In [4]:
```
# adapted from nb21
y = dfSales["sales"]
bhat1, ahat, rval, pval, stderr = stats.linregress(dfSales["lakes"], dfSales["sales"])
print("Fitted Model: Y = {:.5f} + {:.5f}x".format(ahat, bhat1))
```

```
Fitted Model: Y = 24.63378 + -0.97230x
```

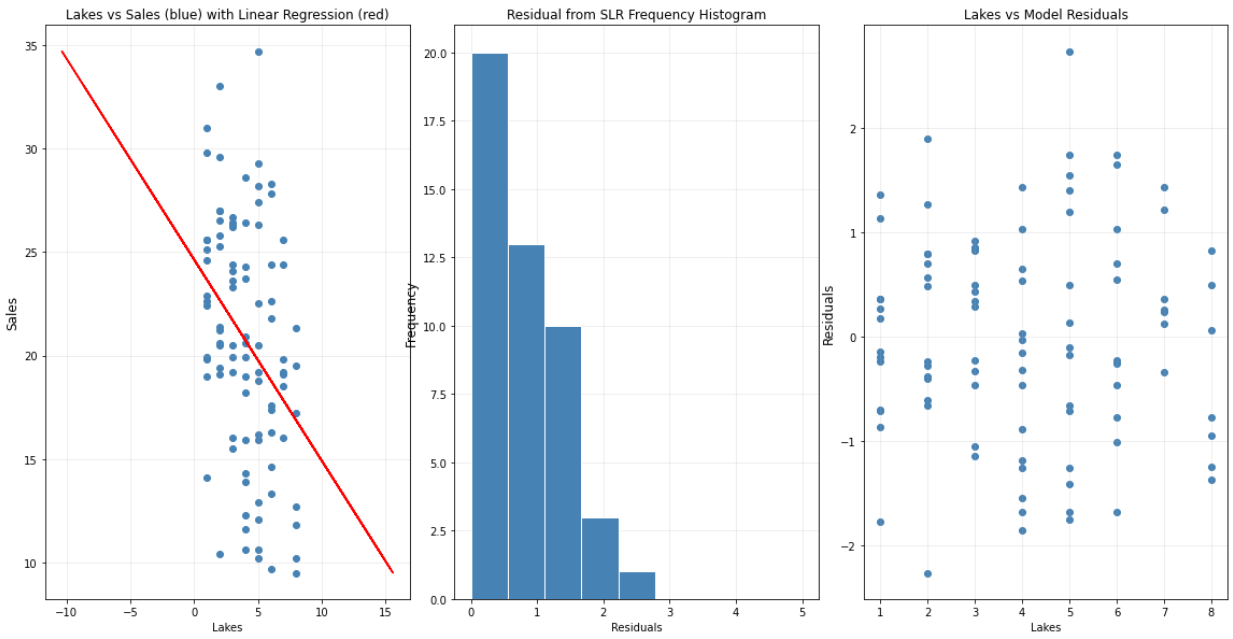
$$Y = 24.63378 + -0.97230x$$

- ***(ii) (3 points) Make a series of 3 side-by-side plots (described below).***
  Title each plot and label your axes on each plot.
  - ***Lefthand plot: scatter plot of the data (i.e. lakes vs sales) and overlay the simple linear regression line you found above.***
  - ***Middle plot: frequency histogram of residuals of your SLR model.***
  - ***Righthand plot: scatter plot where the lake data values are the x-axis and the model residuals are the y-axis.***

In [5]:
```
y = dfSales["sales"]
bhat1, ahat, rval, pval, stderr = stats.linregress(dfSales["lakes"], dfSales["sales"])
x = (y-ahat)/bhat1
```

```
fig, axes = plt.subplots(nrows=1, ncols=3, figsize=(16,8))
fig.tight_layout(h_pad=4)

# Lefthand
axes[0].scatter(dfSales['lakes'], dfSales['sales'], color="steelblue")
axes[0].grid(alpha=0.25)
axes[0].set_xlabel("Lakes", fontsize=10)
axes[0].set_ylabel("Sales", fontsize=12)

axes[0].plot(x,y,'r')
axes[0].set_title("Lakes vs Sales (blue) with Linear Regression (red)");

# Frequency Histogram
x = dfSales["lakes"]
x = sm.add_constant(x)
model = sm.OLS(y, x).fit()
influence = model.get_influence()
resid = influence.resid_studentized_internal
my_bins = np.linspace(0,5,num=10)
axes[1].hist(resid,bins=my_bins,facecolor="steelblue", edgecolor="white")
axes[1].set_title("Residual from SLR Frequency Histogram")
axes[1].set_xlabel("Residuals", fontsize=10)
axes[1].set_ylabel("Frequency", fontsize=12)
axes[1].grid(alpha=0.25)
axes[1].set_axisbelow(True)

# Residuals
axes[2].scatter(dfSales['lakes'], resid, color="steelblue")
axes[2].grid(alpha=0.25)
axes[2].set_xlabel("Lakes", fontsize=10)
axes[2].set_ylabel("Residuals", fontsize=12)
axes[2].set_title("Lakes vs Model Residuals");
model
```

Out[5]: <statsmodels.regression.linear_model.RegressionResultsWrapper at 0x7fb19a2b4d60>



- ***(iii). (2 points) Does this model seem appropriate? For each of the 4 major assumptions of a simple linear regression model, use your plots in part (ii) to decide whether or not the dataset meets those assumptions.***

I list each assumption and say if the dataset meets it:

1) The relationship between variables is linear.

The first plot shows that the relationship between sales and lakes is not linear.

2) Errors are independent of each other.

As seen in the third plot the errors seem independent.

3) Errors are normally distributed with mean of 0.

The mean of the residuals is 0.93 which is NOT 0 and the distribution seems skewed right.

4) Errors have equal variance.

[1] Part B

(2 points) What is the meaning of the slope in the model in terms of the variables in the dataset? Based on the physical meaning of the variables, does the value of this slope seem reasonable or does it surprise you? Explain.

The B for the model is -0.97230. This value is negative so it would say that as lakes are increased, the sales decrease. This would not seam reasonable since paddleboards need to be used on water (like in a lake).

[1] Part C

Your friend would like to know if there's sufficient evidence from this model to conclude that the number of nearby lakes/reservoirs is a statistically significant linear predictor of sales revenue. You decide to do a hypothesis test at the $\alpha = 0.05$ significance level.

- ***(i). (1 points) State the null and alternate hypotheses for your test.***

$H_0 : \beta = 0$

This would mean the slope is zero and lakes has no affect on sales.

$H_1 : \beta \neq 0$

This would mean the slope is nonzero and lakes do have an effect on sales.

- ***(ii). (1 points) What is the appropriate test (by name, not formula) to use for this, and what are the degrees of freedom?***

Test statistic. DF = 95-2 = 93

- ***(iii). (2 points) Calculate the critical value and state the decision rule for this test at an alpha level of 5%.***

In [6]:
```
#adapted from nb21
L_CI = (bhat1 - stats.t.ppf(0.975, len(x)-2) * stderr)
U_CI = (bhat1 + stats.t.ppf(0.975, len(x)-2) * stderr)
print("95% CI = [{:.3f}, {:.3f}], CI width = {:.3f}".format(L_CI, U_CI, U_CI - L_CI))
print("The B stat",np.round(bhat1,4),"is included in the interval and 0 is not include
print("So, zero is not reasonable value for the B stat and a relationship between the
```

```
95% CI = [-1.504, -0.441], CI width = 1.063
The B stat -0.9723 is included in the interval and 0 is not included in the interval
So, zero is not reasonable value for the B stat and a relationship between the variab
les is present.
```

- ***(iv). (2 points) What is the test statistic, p-value, and conclusion of your hypothesis test?***

In [7]:
```
# adapted from nb21
Tb = (bhat1-0)/stderr
print("test statistic Tb = {}".format(Tb))
tpval = 2*(stats.t.cdf(Tb, df=len(x)-2))
print("p-value = {}".format(tpval))
```

```
test statistic Tb = -3.6328870640805593
p-value = 0.0004588002776241505
```

Since the p-value 0.000458 is less than $\alpha = 0.05$ we reject the null hypothesis and conclude that there is sufficient statistical evidence that $\beta \neq 0$ and the relationship between the feature and response is real.

Less lakes within a 50 mile radius is correlated with lower sales.

Every additional lake within a 50 mile radius results in a reduction of 0.97230 units of sales.

[1] Part D

(3 points) What percent of total variation in sales revenue can be explained by the number of nearby lakes/reservoirs? Explain.

In [8]:
```
## adapted from nb22
x = dfSales["lakes"]
y = dfSales["sales"]
yhat = ahat + bhat1*x
# 'ybar' contains the mean of the y-values
ybar = np.mean(y)
```

```
# sum of squared errors
sse = np.sum((y-yhat)**2)
# total sum of squares
sst = np.sum((y-ybar)**2)
# Our hand calculated coefficient of determination.
R2 = 1 - sse/sst
print(R2)
print("Therefore",np.round(R2*100,3),"% of total variation in sales can be explained b
print("This is the proportion that can be explained by the number of nearby lakes.")
print("It is a pretty low number.")
```

```
0.12427620833332198
Therefore 12.428 % of total variation in sales can be explained by the new model
This is the proportion that can be explained by the number of nearby lakes.
It is a pretty low number.
```

Problem 2: Go Big- Multiple Linear Regression

For this problem you are welcome to use any built-in methods/functions we have used in class. If a built-in method outputs summary statistics or other useful output variables you can use that output directly when answering any questions below UNLESS otherwise specified in the problem (i.e. unless we tell you in the problem to write code to calculate a particular statistic from scratch based on its definition, you can just get that statistic/info from the output of a built-in method/function).

You feel unsatisfied with the results from the model in Problem 1.

Problem 2 will involve steps to improve your model from Problem 1 using Multiple Linear Regression (MLR).

We will continue working with the same dataset from Problem 1 (paddle.csv) throughout this problem.

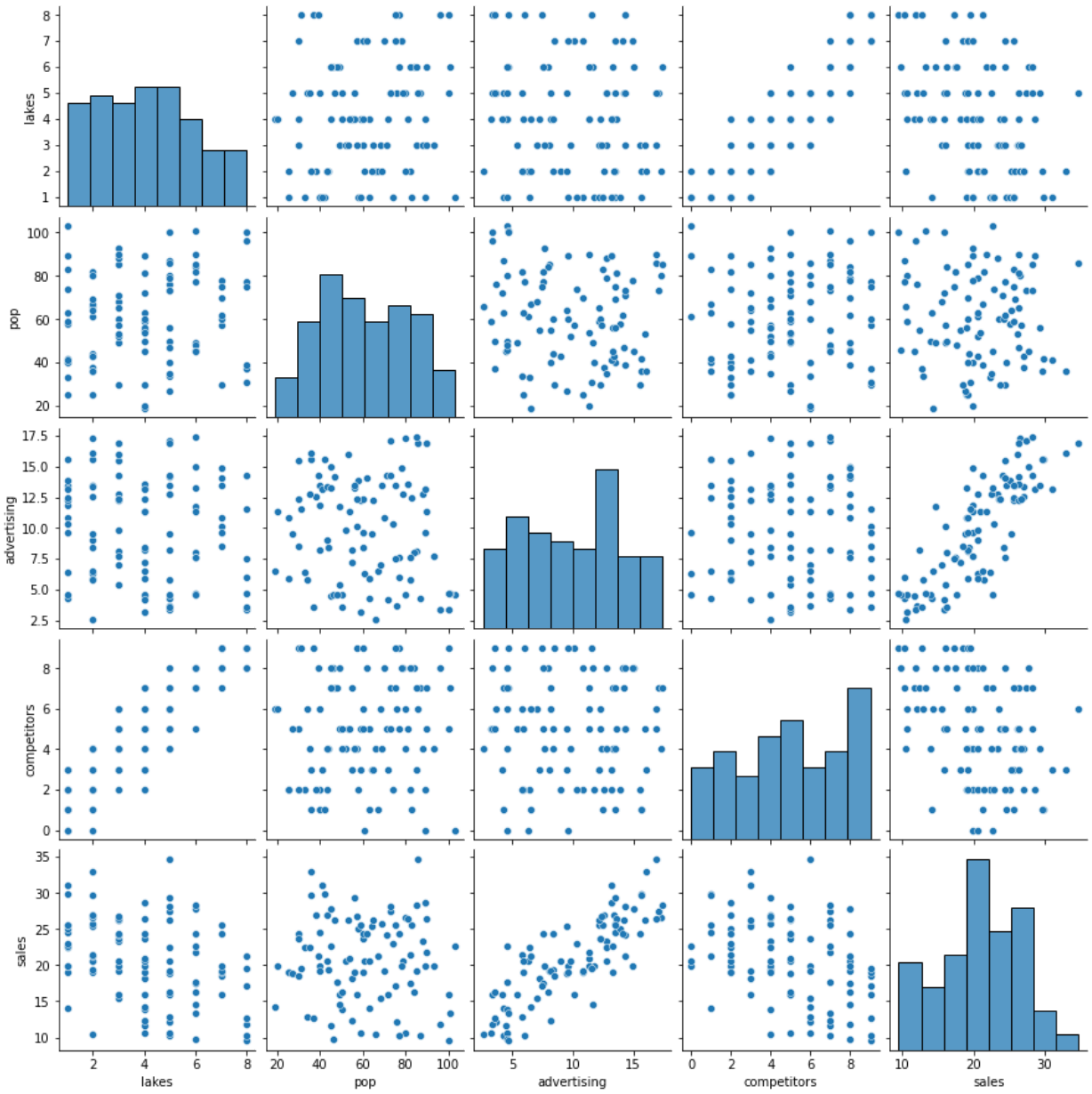
In an effort to improve the model from Problem 1, you decide to add in one additional predictor. To determine which predictor to add next you start with some scatterplots of the dataset.

[2] Part A: Explore

- ***(i). (1 point) Use the function `seaborn.pairplot` to create pairwise scatterplots of the continuous predictors/covariates in the paddle.csv data, both against each other and against the outcome (sales).***
  https://seaborn.pydata.org/generated/seaborn.pairplot.html

In [9]:
```
#Uncomment the code below and replace the input with the name of your dataframe from Q
dfSales = pd.read_csv("paddle.csv")
seaborn.pairplot(dfSales)
```

Out[9]: <seaborn.axisgrid.PairGrid at 0x7fb19a33f790>



- ***(ii). (2 points) Based on your scatterplots, which (if any) explanatory variable(s) appear to have a linear relationship with sales revenue? Explain.***

We look at the bottom row to see a linear relationship between **Advertising** and Sales.

- ***(iii). (2 points) Do any of the explanatory variables exhibit collinearity? Explain.***

Looking at **Advertising** in the third row (or third column), no other variables showed a linear relationship so **no collinearity**.

Looking at **Lakes**, we see a linear relationship and thus **collinearity** with **Competitors** .

[2] Part B: Adjust the Model

(i). (2 points) Choose one additional explanatory variable to add to your model, based on your exploratory analysis in Part 2A. Explain your reasoning for the variable you have chosen.

We choose to add the explanatory variable **Advertising** as it showed no collinearity among the other variables and have the clearest linear relationship with **Sales**

(ii). (2 points) Create a Multiple Linear Regression Model with `sales` as the dependent variable, `lakes` as one of the independent variables and the new variable you have chosen as the other independent variable.

Print the model output summary table for this new model, and then write the equation for this new model in a markdown cell.

In [10]:
```
## adapted from nb22
X = dfSales[["lakes", "advertising"]]
# Add a constant to the 'X' array for the intecept.
X = sm.add_constant(X)
# Collect the response (Dependent variable) data in an array.
y = dfSales["sales"]
MLRmodel = sm.OLS(y, X).fit()
print(MLRmodel.params)
MLRmodel.summary()
```

```
const          12.481377
lakes          -0.691324
advertising     1.110071
dtype: float64
```

Out[10]:

OLS Regression Results

Dep. Variable:	sales	**R-squared:**	0.742
Model:	OLS	**Adj. R-squared:**	0.737
Method:	Least Squares	**F-statistic:**	132.4
Date:	Fri, 09 Dec 2022	**Prob (F-statistic):**	8.31e-28
Time:	00:43:13	**Log-Likelihood:**	-237.35
No. Observations:	95	**AIC:**	480.7
Df Residuals:	92	**BIC:**	488.4
Df Model:	2		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	12.4814	1.056	11.816	0.000	10.383	14.579
lakes	-0.6913	0.147	-4.696	0.000	-0.984	-0.399
advertising	1.1101	0.075	14.849	0.000	0.962	1.259

Omnibus:	0.577	**Durbin-Watson:**	1.607
Prob(Omnibus):	0.749	**Jarque-Bera (JB):**	0.710
Skew:	0.158	**Prob(JB):**	0.701
Kurtosis:	2.718	**Cond. No.**	39.6

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

We write the MLR equation:

$$\texttt{sales} = 12.48 - 0.69 \times \texttt{Lakes} + 1.11 \times \texttt{Advertising}$$

[2] Part C: Inference on MLR Model: Step 1

For this part of the problem you can assume (without checking) that this new model meets the standard assumptions of multiple linear regression.

Lets see if this new model as a whole is significant at an alpha level of 5%. In other words, we want to use a single test to see if at least one of the two explanatory variables really has a significant linear association with the response variable.

- ***(i). (3 points) State the null and alternate hypotheses we should test at an alpha level of 5% , and explain what they mean.***

$H_0 : \beta_1 = \beta_2 = 0$
$H_1 : \beta \neq 0$

The null is checking if all the slopes are equal to 0 and therefore the variables don't have a linear relationship with sales. So the fit of the intercept only model and the above model are equal.

The alternate checks if at least one β_i is different than 0. The fit of ther intercept model is reduced compared to the MLR model.

- ***(ii). (2 points) What is the appropriate test (by name, not formula) to use for this, and what are the degrees of freedom?***

An $\mathbf{F\text{-}Test}$ is a hypothesis test where we check the overall significance of the model. and we have $\mathbf{95\text{-}2\text{-}1 = 92}$ degrees of freedom

- ***(iii). (2 points) Calculate the critical value and state the decision rule for this test at an alpha level of 5%.***

In [11]:
```
n = 95
p = 2
df = n-p-1
F_crt = stats.f.ppf(.95,p-1,1)
print('\n F_critical = {:.3f}'.format(F_crt))
```

```
 F_critical = 161.448
```

If our F value we calculate is less than 161.448, we will reject the null hypothesis and find the model significant as a whole.

- ***(iv). (3 points) Write code to calculate the appropriate test statistic from scratch and then summarize the conclusion of your hypothesis test. (Note for credit you must write code to calculate the appropriate test statistic from its formula using your data and model. You can use the model output summary table to check your answer).***

In [12]:
```
## adapted from nb22
def fit(X, params):
    return params[0] + params[1]*X.lakes + params[2]*X.advertising
yhat = fit(X, MLRmodel.params)
# 'ybar' contains the mean of the y-values
ybar = np.mean(y)
# sum of squared errors
sse = np.sum((y-yhat)**2)
# total sum of squares
sst = np.sum((y-ybar)**2)
# Test statistic calculation
F = ( (sst-sse)/p )/( (sse/df) )
#print('SSE = {:0.3f}, SST = {:.3f}, and R^2 = {:.3f}'.format(sse,sst,R2))
print('\n F = {:.3f}'.format(F))
```

```
F = 132.427
```

Our F value is less than the critical F value.

Therefore we find the model significant as a whole.

[2] Part D: Inference on MLR Model: Step 2

After Part 2C, you should have found that the new model as a whole is significant. So, it will be helpful to know exactly which variable (or variables) have a significant linear association with the response variable.

- ***(i). (2 points) What is the appropriate test (by name, not formula) to test one-by-one whether an individual explanatory variable in your model is a statistically significant linear predictor of the response variable after adjusting for the other explanatory variables?***

The $\mathbf{t\text{-}test}$ is for each individual coefficient B_1, B_2 and tells us whether they are nonzero in presense of one another.

For each slope β, we want to see if they indeed have a linear relationship with sales:
$H_0 : \beta_i = 0$
$H_1 : \beta_i \neq 0$
In this test, we get each B t-value and get then p-value and compare to α.

- ***(ii). (2 points) Explain the conclusion(s) of this test for each of the explanatory variables in your model, using output from your model summary table. (Note: For this part of the problem you do not need to calculate each test statistic from scratch- instead you can just look up the appropriate results from the output summary table and explain your conclusions below).***

In [13]:
```
MLRmodel.summary()
```

Out[13]:

OLS Regression Results

Dep. Variable:	sales	**R-squared:**	0.742
Model:	OLS	**Adj. R-squared:**	0.737
Method:	Least Squares	**F-statistic:**	132.4
Date:	Fri, 09 Dec 2022	**Prob (F-statistic):**	8.31e-28
Time:	00:43:13	**Log-Likelihood:**	-237.35
No. Observations:	95	**AIC:**	480.7
Df Residuals:	92	**BIC:**	488.4
Df Model:	2		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	12.4814	1.056	11.816	0.000	10.383	14.579
lakes	-0.6913	0.147	-4.696	0.000	-0.984	-0.399
advertising	1.1101	0.075	14.849	0.000	0.962	1.259

Omnibus:	0.577	**Durbin-Watson:**	1.607
Prob(Omnibus):	0.749	**Jarque-Bera (JB):**	0.710
Skew:	0.158	**Prob(JB):**	0.701
Kurtosis:	2.718	**Cond. No.**	39.6

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

We look at the "t" values for lakes (-4.696) and advertising (14.849) and then the p-values associate with them.

We are rejecting the null at α = 0.05.

For $lakes$, we find the p value to be less than 0.05 and therefore reject the null.

For $advertising$, we also find the p value to be less than 0.05 and therefore reject the null.

[2] Part E: Quantifying Model Goodness of Fit

(2 points) What percent of total variation in sales revenue can be explained by the new model you found in Part 2B? Is this an improvement compared to the SLR model in Question 1? Explain.

In [14]:
```
# total sum of squares from regression
ssr = np.sum((yhat-ybar)**2)
```

```
percent = ssr/(sse+sst+ssr)
print("Therefore",np.round(percent*100,3),"% of total variation in sales can be explai
print("This IS an improvement from the SLR in Question 1")
print("Since more variation can be explained by the model, the model has been improvec
```

```
Therefore 37.11 % of total variation in sales can be explained by the new model
This IS an improvement from the SLR in Question 1
Since more variation can be explained by the model, the model has been improved.
```

[2] Part F: Final Pass at a Model

You are curious if the model you found above is really the best you can do with the dataset.

In this open-ended problem you will use the same dataset (paddle.csv) but you may use any subset of the columns to build your "best" Multiple Linear Regression model. i.e. you want to create the best model you can to quantify how sales can be explained and predicted by the features in this dataset.

For this problem the dependent variable will still be sales. Any other variables will be candidates for explanatory variables.

Your goal is to demonstrate that you have an understanding of how to choose between columns and validate/check for issues in a multiple linear regression problem.

- ***(i). (4 pts) By adding variables step-by-step to a minimal model or by removing variables step-by-step from the full model, use one or more of the criteria covered in our class to create a reasonable candidate model (i.e. the "best" you can do with the data given). The criteria you use may include (but are not limited to):***
  - ***adjusted*** R^2
  - ***inclusion/removal of most or least-significant t-tests on coefficients***
  - ***F-tests to compare between models***

In addition to code showing the model variations you try, use markdown cell(s) to write-up an explanation of your reasoning for each step of this process as you construct a final version of the model.

Your explanation should include the results of the F-test and t-tests for your final model. If you choose to include an insignificant explanatory variable in your final model you must fully justify your reasoning.

In [15]:
```
## adapted from nb22
X = dfSales[["lakes", "advertising",'competitors','pop']]
# Add a constant to the 'X' array for the intecept.
X = sm.add_constant(X)
# Collect the response (Dependent variable) data in an array.
y = dfSales["sales"]
MLRmodel = sm.OLS(y, X).fit()
print(MLRmodel.summary())
print("We begin with 4 variables")
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  sales   R-squared:                       0.785
Model:                            OLS   Adj. R-squared:                  0.776
Method:                 Least Squares   F-statistic:                     82.25
Date:                Fri, 09 Dec 2022   Prob (F-statistic):           3.17e-29
Time:                        00:43:13   Log-Likelihood:                -228.68
No. Observations:                  95   AIC:                             467.4
Df Residuals:                      90   BIC:                             480.1
Df Model:                           4
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const         12.1010      1.289      9.390      0.000       9.541      14.661
lakes          0.4020      0.308      1.305      0.195      -0.210       1.014
advertising    1.1393      0.070     16.391      0.000       1.001       1.277
competitors   -1.0352      0.256     -4.042      0.000      -1.544      -0.526
pop            0.0127      0.014      0.927      0.356      -0.015       0.040
==============================================================================
Omnibus:                        0.231   Durbin-Watson:                   1.642
Prob(Omnibus):                  0.891   Jarque-Bera (JB):                0.366
Skew:                           0.103   Prob(JB):                        0.833
Kurtosis:                       2.776   Cond. No.                         297.
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly spec
ified.
We begin with 4 variables
```

In [16]:

```
# remove pop
X = dfSales[["lakes", "advertising",'competitors']]
# Add a constant to the 'X' array for the intecept.
X = sm.add_constant(X)
# Collect the response (Dependent variable) data in an array.
y = dfSales["sales"]
MLRmodel = sm.OLS(y, X).fit()
print(MLRmodel.summary())
print("We remove pop")
# remove lakes
X = dfSales[["advertising",'competitors']]
# Add a constant to the 'X' array for the intecept.
X = sm.add_constant(X)
# Collect the response (Dependent variable) data in an array.
y = dfSales["sales"]
MLRmodel = sm.OLS(y, X).fit()
print(MLRmodel.summary())
print("We remove lakes")
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  sales   R-squared:                       0.783
Model:                            OLS   Adj. R-squared:                  0.776
Method:                 Least Squares   F-statistic:                     109.6
Date:                Fri, 09 Dec 2022   Prob (F-statistic):           4.24e-30
Time:                        00:43:13   Log-Likelihood:                -229.13
No. Observations:                  95   AIC:                             466.3
Df Residuals:                      91   BIC:                             476.5
Df Model:                           3
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          12.8771      0.979     13.157      0.000      10.933      14.821
lakes           0.4411      0.305      1.446      0.152      -0.165       1.047
advertising     1.1334      0.069     16.387      0.000       0.996       1.271
competitors    -1.0567      0.255     -4.147      0.000      -1.563      -0.551
==============================================================================
Omnibus:                        0.137   Durbin-Watson:                   1.627
Prob(Omnibus):                  0.934   Jarque-Bera (JB):                0.239
Skew:                           0.085   Prob(JB):                        0.887
Kurtosis:                       2.822   Cond. No.                         43.1
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly spec
ified.
We remove pop
                            OLS Regression Results
==============================================================================
Dep. Variable:                  sales   R-squared:                       0.778
Model:                            OLS   Adj. R-squared:                  0.773
Method:                 Least Squares   F-statistic:                     161.4
Date:                Fri, 09 Dec 2022   Prob (F-statistic):           8.24e-31
Time:                        00:43:13   Log-Likelihood:                -230.21
No. Observations:                  95   AIC:                             466.4
Df Residuals:                      92   BIC:                             474.1
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          13.1652      0.964     13.658      0.000      11.251      15.080
advertising     1.1204      0.069     16.242      0.000       0.983       1.257
competitors    -0.7267      0.114     -6.368      0.000      -0.953      -0.500
==============================================================================
Omnibus:                        0.286   Durbin-Watson:                   1.585
Prob(Omnibus):                  0.867   Jarque-Bera (JB):                0.426
Skew:                           0.112   Prob(JB):                        0.808
Kurtosis:                       2.761   Cond. No.                         40.0
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly spec
ified.
We remove lakes
```

- ***(ii). (2 pts) Give the equation for your final model (in markdown) AND interpret the meaning of the intercept and coefficients in your final model in the context of the problem. Explain how your final model makes sense in the context of the problem (or if it doesn't, you'll want to try to explain why not).***

$$\texttt{sales} = 12.48 + 1.12 \times \texttt{Advertising} - 0.73 \times \texttt{competitors}$$

As advertising is increased for the store, sales should increase and as more competitors are nearby, sales should decrease from competition.

[2] Part G: Back To Your Friend

After finishing your final model, you report the results to your friend. Your friend asks you to answer the following questions:

- ***i). (3 pt) Provide 95% confidence intervals for each of the coefficients in your final model (you can use the summary table to find these and you do not need to provide a confidence interval for the intercept). Since your friend is rusty at statistics, write an explanation/ interpretation of what each of the confidence intervals mean in the context of this dataset.***

The $advertising$ 95% CI is (0.983, 1.257) which means for every thousand dollars spent on advertising, the shop sees an increase of sales of somewhere between 983 and 1257 dollars.

The $advertising$ 95% CI is (-0.953, -0.500) which means for every thousand dollars spent on advertising, the shop sees an decrease of sales of somewhere between 500 and 953 dollars.

This is said with 95% confidence.

- ii). ***(2 pts) Your friend is thinking of opening a new paddleboard shop in a town with population density of 75 people per square mile. The town has 4 lakes nearby and currently only 2 nearby competitors. Based on your final model, how much money do you recommend they allocate to this shop for advertising during the first year so that the shop's first year sales revenue will be at least*** $\$25,000$***? Explain.***

In [17]:
```
advertising = (25000-12.48+.73*2)/1.12
print(advertising)
```

```
22311.589285714283
```

Therefore, I'd recommended spending $22,311 on advertising

- ***iii). (1 pt) Is there any other data (i.e. other possible explanatory variables) you recommend your friend gather in the future to possibly improve the model of sales revenue? Explain.***

It might be good to consider what the average price of a board cost at a shop. This would help to see if cheaper or more expensive boards are feasible.

Problem 3

Narwhals are a threatened species. There are approximately 80,000 left in existence. In an attempt to determine the health of the current narwhal population, a number of these whales were captured and weighed. Since there are not many of these creatures to find, only a small sample was able to be counted and the underlying narwhal weight distribution is unknown.

Given the data:

Narwhal sample weights (pounds): 700, 900, 1000, 1000, 1200, 1400, 1500, 1600, 1600, 1700, 1900, 2000, 2100, 2100, 2300

[3] Part A

(3 points) Produce a 95% confidence interval for narwhal population median weight.

In [18]:
```
x = [700, 900, 1000, 1000, 1200, 1400, 1500, 1600, 1600, 1700, 1900, 2000, 2100, 2100,
#create 95% confidence interval for population median weight
stats.t.interval(0.95, df=len(x)-1, loc=np.median(x), scale=stats.sem(x))
```

Out[18]:
```
(1326.558889566094, 1873.441110433906)
```

[3] Part B

(3 points) Produce a 95% confidence interval for narwhal population mean weight.

In [19]:
```
#create 95% confidence interval for population mean weight
stats.t.interval(0.95, df=len(x)-1, loc=np.mean(x), scale=stats.sem(x))
```

Out[19]:
```
(1259.8922228994272, 1806.7744437672393)
```

[3] Part C

(3 points) Produce a 95% confidence interval for narwhal population standard deviation of weights.

In [20]:
```
#create 95% confidence interval for population std weight
stats.t.interval(0.95, df=len(x)-1, loc=np.std(x), scale=stats.sem(x))
```

Out[20]:
```
(203.5867247660491, 750.4689456338612)
```

Problem 4

Ostriches are typically around 7 to 9 feet tall.

Some folks would claim that ostriches at one zoo are particularly taller than ostriches at another zoo due to influences such as climate and/or diet. We have samples of ostrich height from two different zoos.

(For this problem you can assume that the ostrich populations from which these samples are taken are normally distributed with equal (but unknown) population variance).

(6 points) Create a 95% CI for the difference between the mean ostrich heights for zoo A and zoo B, and interpret your answer.

The heights of ostriches from Zoo A are found in the file `height1.csv`.

The heights of ostriches from Zoo B are found in the file `height2.csv`.

In [21]:
```
# adapted from HW7
dfA = pd.read_csv("height1.csv")
dfB = pd.read_csv("height2.csv")
# get data
x = dfA["height1"]
y = dfB["height2"]
# calculate mean, standard deviation, and count
X = np.mean(x)
Y = np.mean(y)
std1 = np.std(x) # a std
std2 = np.std(y) # b std
m = 23
n = 23
z = 1.96
b1 = X-Y+z*np.sqrt((std1**2)/m + (std2**2)/n) # bound 1
b2 = X-Y-z*np.sqrt((std1**2)/m + (std2**2)/n) # bound 2
print("The difference in mean ostrich heights between each zoo with a 95% CI gives [",
```

```
The difference in mean ostrich heights between each zoo with a 95% CI gives [ 0.98236
, 1.41764 ]
```

Note zero is not included in this interval so 95% of intervals created this way will contain the true ostrich mean height difference.

Specifically, this interval indicates that mean ostrich heights from zoo A are greater than zoo B by somewhere between 0.98236 and 1.41764 feet.

Problem 5

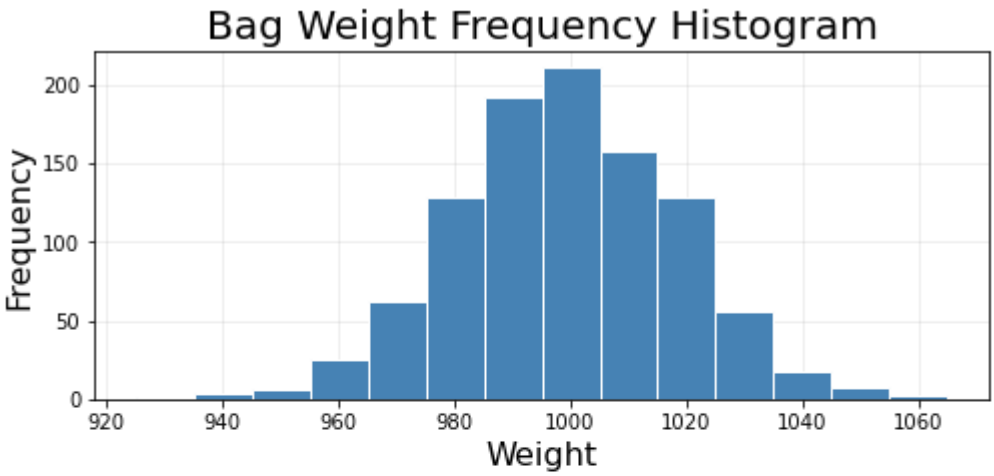
Suppose you are working for a government agency that provides food to war torn countries. The cornmeal packaging facility that you oversee is producing hundreds of thousands of bags of cornmeal for this purpose.

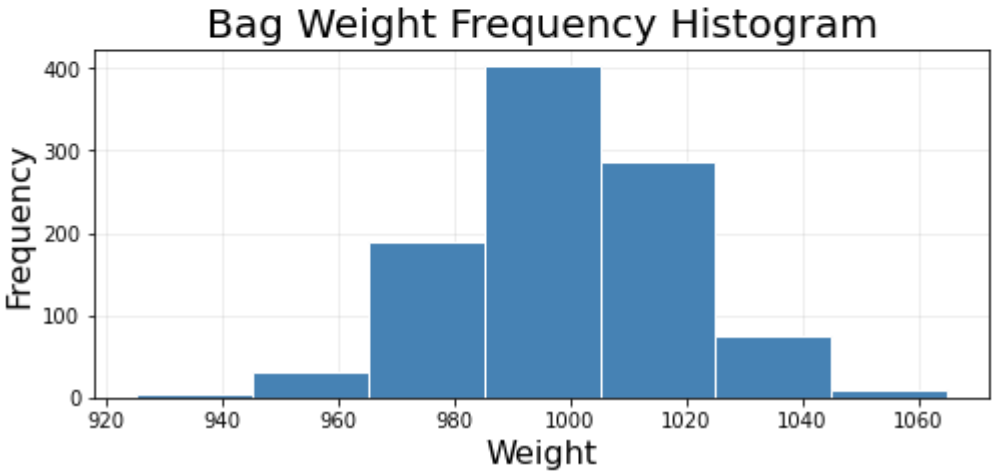
You take a simple random sample of bags from the production line and weigh them. The data for this sample is found in `cornMeal.csv`.

[5] Part A

(1 pt) Create a frequency histogram of the data to get a visual idea of how normal the data appears to be distributed.

In [22]:
```
dfCorn = pd.read_csv("cornMeal.csv")

## adapted from nb03 exercise 2

# Initialize figure subplots
fig, axes = plt.subplots(nrows=2, ncols=1, figsize=(8,8))
## First plot with more bins
my_bins = range(925,1075,10)
dfCorn.hist(column="bagWeight", ax=axes[0],  bins=my_bins, facecolor="steelblue", edge
axes[0].set_title("Bag Weight Frequency Histogram", fontsize=20)
axes[0].set_xlabel("Weight", fontsize=16)
axes[0].set_ylabel("Frequency", fontsize=16)
axes[0].grid(alpha=0.25)
axes[0].set_axisbelow(True)
## Second plot with less bins
my_bins = range(925,1075,20)
dfCorn.hist(column="bagWeight", ax=axes[1], bins=my_bins, facecolor="steelblue", edgec
axes[1].set_title("Bag Weight Frequency Histogram", fontsize=20)
axes[1].set_xlabel("Weight", fontsize=16)
axes[1].set_ylabel("Frequency", fontsize=16)
axes[1].grid(alpha=0.25)
axes[1].set_axisbelow(True)
fig.subplots_adjust(hspace=.5)
```





[5] Part B

Since tax money is paying for each bag to have 1000g of cornmeal, it's important to make sure that each bag actually contains 1000g. Assume that this sample (`cornMeal.csv`) is representative of the population (i.e. bags produced on this production line) from which it was drawn.

(3 points) According to the mean, standard deviation and distribution of this sample, we can infer that what percent of bags from the entire population contain less than 1000g?

In [23]:
```
x = dfCorn["bagWeight"]
n = len(x)
X = np.mean(x)
sigma = np.std(x)
percent = norm.cdf(1000,loc=X,scale=sigma)
print("P( Y \u2264 1000) = x \n x = {:.2f}".format(100*percent),"%")
print("Therefore about 52% of the entire population contains less than 1000g")
```

```
P( Y ≤ 1000) = x
 x = 52.10 %
Therefore about 52% of the entire population contains less than 1000g
```

[5] Part C

The federal director in charge of this project has decided that a production line that produces more than 50% of bags without 1000g of cornmeal is cheating the taxpayers. In other words, the director has decided that too many bags are being filled with less than 1000g of cornmeal and tells you to fix the problem.

Specifically, your task/mandate is to make sure at least 95% of cornmeal bags are filled within 5 grams of the labeled "1000g per bag".

You have the ability to adjust a dial on the production machine that determines the mean amount of corn meal injected into a bag each time **or** you can adjust the shut-off valve so the differences in the filled amount per bag significantly reduces.

You have two advisors working with you on this project. The first advisor tells you to simply adjust the dial on the machine regulating the mean amount injected into each bag so that 1000g is 1.7 standard deviations below the new mean. This would ensure that very few bags (only around 4%) produced are actually below the mandated 1000g. This would surely satisfy the director and fulfill your mandate.

In [24]:
```
# If you are curious, this is where the previously mentioned 4% came from.
pp = norm.cdf(-1.7)
print(pp*100)
```

```
4.456546275854304
```

(2 points) If you follow this advice, then what would you set as the new mean amount of cornmeal to inject into each bag?

In [25]:
```
std = 1.7
#X = 1002.89
X = 1000+1.7*sigma
print("The new mean is confirmed to be",X)
```

```
The new mean is confirmed to be 1032.3
```

[5] Part D

Your second advisor points out that increasing the mean to a higher level will create overfilling of bags or waste.

Therefore, you decide to measure the overage that might occur should you follow the instructions of advisor 1.

(3 points) If you followed the advice of advisor 1, then you would increase the mean amount of cornmeal injected into each bag. With this new mean, what percentage of bags produced going forward would we expect to be greater than $1010g$ ***?***

In [26]:
```
percent = 1 - norm.cdf(1010,loc=X,scale=sigma)
print("P( Y \u2264 1010) = x \n x = {:.5f}".format(100*percent),"%")
print("About",np.round((100*percent),4),"of the entire population should contain more
```

```
P( Y ≤ 1010) = x
 x = 87.97392 %
About 87.9739 of the entire population should contain more than 1010g
```

[5] Part E

Your second advisor goes on to say, in an effort to control overfilling, one **should not** increase the mean amount of cornmeal per bag. This advisor claims you should instead adjust the accuracy of the shut-off valve to control how much (over/under) cornmeal is put into each bag.

(3 points) So, if you decide not to adjust the mean and instead adjust the shut-off valve (i.e. the standard deviation), then what should the new standard deviation be to ensure 98% ***of bags are within 5 grams of the mean?***

In [27]:
```
X = np.mean(x)
sigma = 2.14 # varied until 98% was acheived)
res = norm.cdf(1004,loc=X,scale=sigma)-norm.cdf(994,loc=X,scale=sigma)
print("P(994 \u2264 X \u2264 1004) = {:.6f}".format( res ))
print(res*100)
print("Therefore the std should be around 2.14")
```

```
P(994 ≤ X ≤ 1004) = 0.980532
98.0532126385034
Therefore the std should be around 2.14
```

[5] Part F

(3 points) Using this new standard deviation from Part E (and not adjusting the mean), what percent of bags would we expect to be within 5 grams of the labeled "1000g per bag"? In other words, going forward, what percent of bags would we expect to contain between $995g$ ***and*** $1005g$ ***of cornmeal? Have you satisfied the mandate?***

In [28]:
```
res = norm.cdf(1005,loc=X,scale=sigma)-norm.cdf(995,loc=X,scale=sigma)
print("P(995 \u2264 X \u2264 1005) = {:.6f}".format( res ))
print("Therefore, 96.6% of bags are within 5 grams of cornmeal. We have satisfied the
```

```
P(995 ≤ X ≤ 1005) = 0.966674
Therefore, 96.6% of bags are within 5 grams of cornmeal. We have satisfied the mandat
e
```

Problem 6

Consider a CU CS course with two sections; Section 001 and Section 002.

The .csv file `Section_001.csv` contains the twelve quiz scores of 4 named students from section 001.

The .csv file `Section_002.csv` contains the twelve quiz scores of 4 named students from section 002.

Import the data of these two sets. One of the sections contains four students whose scores **are not** statistically different. The other section contains four students whose scores **are** statistically different.

You can assume that population quiz scores for both sections are normally distributed with the same (unknown) population variance.

In [29]:
```
dfS1 = pd.read_csv("Section_001.csv")
dfS2 = pd.read_csv("Section_002.csv")
dfS1
dfS2
```

Out[29]:

	Eli	Frida	Gob	Hank
0	18.9	16.1	10.6	19.2
1	17.7	15.4	14.9	18.9
2	20.0	16.4	12.9	19.1
3	17.3	14.3	10.7	19.1
4	17.2	16.4	8.2	18.8
5	18.2	16.5	15.9	19.4
6	18.1	16.5	11.9	18.6
7	17.6	15.0	15.5	19.2
8	18.1	16.4	10.3	18.9
9	16.0	14.5	16.2	19.5
10	18.2	15.3	19.2	18.8
11	19.3	16.1	13.5	18.8

[6] Part A

(3 points) Perform ANOVA F-test(s) to determine which data set (section 001 or section 002) contains a group of students whose mean quiz scores are considered different at the $\alpha = 0.05$ ***significance level.***

In [30]:
```
## adapted from nb24

## section 001
# stats.f_oneway returns 2 parameters from the groups of data: F-score and p-value of
F, pval = stats.f_oneway(dfS1["Al"], dfS1["Bob"], dfS1["Cy"],dfS1["Dot"])
# Print out the F-score and the corresponding p-value.
print("Section 1:\n F = {:.5f}".format(F))
print("pval = {:.5f}".format(pval))
# Find the critical F.
k = 4; n = 12;
num = k - 1
den = n - k
F_cr = stats.f.ppf(.95,num,den)
```

```
print("The critical F value is {:.5}".format(F_cr))

## section 002
# stats.f_oneway returns 2 parameters from the groups of data: F-score and p-value of
F, pval = stats.f_oneway(dfS2["Eli"], dfS2["Frida"], dfS2["Gob"],dfS2["Hank"])
# Print out the F-score and the corresponding p-value.
print("Section 2:\n F = {:.5f}".format(F))
print("pval = {:.11f}".format(pval))
# Find the critical F.
k = 4; n = 12;
num = k - 1
den = n - k
F_cr = stats.f.ppf(.95,num,den)
```

```
Section 1:
 F = 0.01345
pval = 0.99784
The critical F value is 4.0662
Section 2:
 F = 26.67605
pval = 0.00000000055
```

The F-score for the **second** section is greater than the critical F: $26.67 > 4.06$, so we reject the null hypothesis imposed for that section.

Or: In section 2, at least one of the group means is different from the others.

This is also shown by the pval for section 2, 0.0000000005, being less than the significance level $\alpha = .05$

[6] Part B

(3 points) Now that you have chosen the data set (the proper CS section) that contains four students whose scores are considered different at the 5% level, use Tukey's Honest Significant Difference (HSD) test to determine which students in this section are statistically different using the MultiComparison module and interpret the results, i.e. compare/rank the students according to quiz scores.

In [31]:

```
## adapted from nb24
data = dfS2.values.T.flatten()  # Pandas to NumPy, Transpose, and 'flatten'
  # data = [5 4 4 3 9 4 4 8 7 5 1 5 9 8 8 10 5 10]
# Create an empty array to hold the names of the archers.
labels = []
# This for-loop puts the archers name in once for each of their scores.
# .count() is a built-in function that returns the number of times an object
#  appears in a list.
for col in dfS2.columns:
    labels = labels + [col]*dfS2[col].count()
mc = MultiComparison(data, labels)
result = mc.tukeyhsd()
print(result)
```

```
Multiple Comparison of Means - Tukey HSD, FWER=0.05
====================================================
group1 group2 meandiff p-adj   lower   upper  reject
----------------------------------------------------
   Eli  Frida  -2.3083 0.0098 -4.1714 -0.4453   True
   Eli    Gob  -4.7333    0.0 -6.5964 -2.8703   True
   Eli   Hank    0.975 0.5077 -0.8881  2.8381  False
 Frida    Gob   -2.425 0.0062 -4.2881 -0.5619   True
 Frida   Hank   3.2833 0.0001  1.4203  5.1464   True
   Gob   Hank   5.7083    0.0  3.8453  7.5714   True
----------------------------------------------------
```

We have 4 groups so we make 4 pairwise comparisons. From the reject column, we find Eli's and Hank's means are not different.

So, we conclude we have sufficient statistical evidence to believe the everyone else has different means. We inspect samples means...

In [32]:
```
print("Eli mean: {:.3f}".format(dfS2["Eli"].mean()))
print("Frida mean: {:.3f}".format(dfS2["Frida"].mean()))
print("Gob mean: {:.3f}".format(dfS2["Gob"].mean()))
print("Hank mean: {:.3f}".format(dfS2["Hank"].mean()))
```

```
Eli mean: 18.050
Frida mean: 15.742
Gob mean: 13.317
Hank mean: 19.025
```

Therefore we say:

$\mu_{Hank} = \mu_{Eli} > \mu_{Frida} > \mu_{Gob}$.

Problem 7

In this question you will import `x.csv` and compare its distribution to a distribution of sample means taken from samples of `x.csv`.

In [33]:
```
dfX = pd.read_csv("x.csv")
dfX
```

Out[33]:

	x
0	28.09
1	19.14
2	134.34
3	42.41
4	84.75
...	...
995	7.35
996	42.21
997	133.64
998	71.75
999	71.38

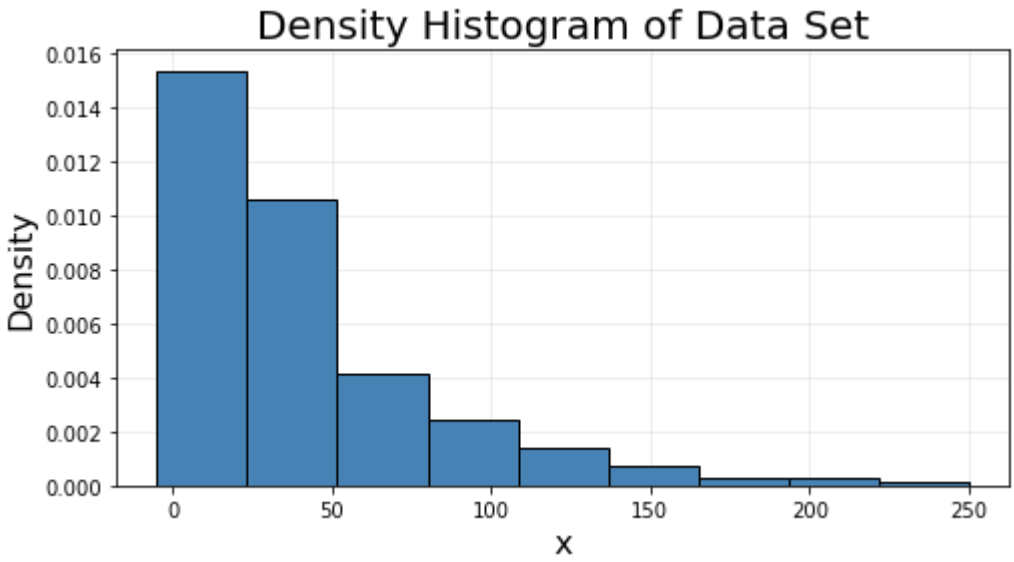
1000 rows × 1 columns

[7] Part A

(3 points) Create a** density histogram **of the data set, find its mean, and describe its shape.

In [34]:
```
## density histogram

# Adapted from nb03
fig, ax = plt.subplots(figsize=(8,4))
my_bins = np.linspace(-5,250,10)
mycolor = 'steelblue'
dfX.hist(column="x", density=True, ax=ax, bins=my_bins, facecolor=mycolor, edgecolor='
ax.set_title("Density Histogram of Data Set", fontsize=20)
ax.set_xlabel("x", fontsize=16)
ax.set_ylabel("Density", fontsize=16)
ax.grid(alpha=0.25)
ax.set_axisbelow(True)

## mean

X = dfX['x'].mean()
print("The mean of the dataset is:",np.round(X,2))

X = dfX['x'].median()
print("The median of the dataset is:",np.round(X,2))
```

```
The mean of the dataset is: 41.35
The median of the dataset is: 27.8
```



The density histogram is right-skewed or more data lies closer towards 0.

[7] Part B

(4 points) Complete the functions below to graph a sample distribution of sample means from this file. Do not change the names or any of the existing code.

In [37]:
```
# 'x' refers to the values in our file x.csv.

# Create a function that returns a single sample mean from a single
#  random sample of size 'sample_size=5' obtained from 'x'.
# Uncomment the function below and use it.

def est_mean_from_sample(sample_size=5):
    sample = np.zeros(sample_size)
    for i in range(0,sample_size):
        sample[i] = np.random.choice(dfX['x'])
    # sample mean calculation
    return np.mean(sample)

# Create a function that returns an array of sample means by
#  calling 'est_mean_from_sample' 'numb_samples' amount of times.
# Uncomment the function below and use it.

def array_of_means(sample_size=5, numb_samples=12):
    ret = np.zeros(numb_samples)
    for i in range(0,numb_samples):
        ret[i] = est_mean_from_sample(sample_size)
    return ret

# Create a function that graphs a distribution of the
#  sample means returned from 'array_of_means'.
# Uncomment the function below and use it.

def sample_mean_dist(sample_size=5, numb_samples=12):
    fig, ax = plt.subplots(figsize=(8,4))
    my_bins = np.linspace(0,100,10)
    mycolor = 'steelblue'
    data = array_of_means(sample_size,numb_samples)
```

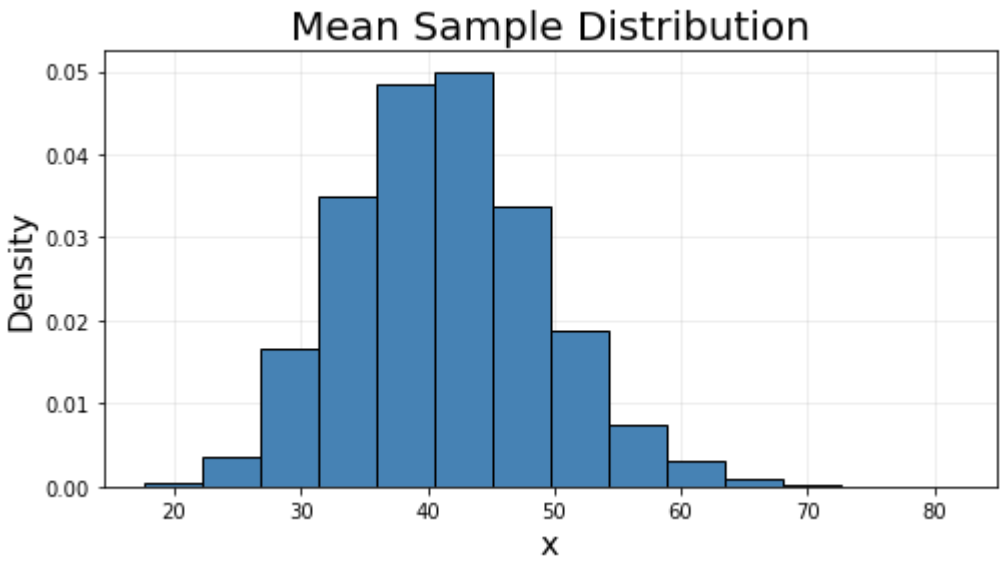
```
    low = min(data)
    high = max(data)
    my_bins = np.linspace(low,high,15)
    plt.hist(data, bins=my_bins, density=True, facecolor=mycolor, edgecolor="black")
    ax.set_title("Mean Sample Distribution", fontsize=20)
    ax.set_xlabel("x", fontsize=16)
    ax.set_ylabel("Density", fontsize=16)
    ax.grid(alpha=0.25)
    ax.set_axisbelow(True)
    print("The mean of the sample distribution is",np.round(np.mean(data),4))
```

[7] Part C

(2 points) Now call `sample_mean_dist()` with sample_size=31 and numb_samples=10000. How close is the mean of the sample distribution to the mean of the distribution of `x.csv` ? Is the distribution of sample means more or less skewed than `x.csv` ? What is the name of the theorem that explains these observations?

In [38]:
```
# Call the graphing function.
# Uncomment the function call below to use it.

sample_mean_dist(sample_size=31, numb_samples=10000)
```

```
The mean of the sample distribution is 41.2971
```



The mean of the sample distribution is 41.2971 which is very close to 41.35.

The distribution of the sample means is much less skewed the the x distribution.

The name of this theorem is the central limit theorem.

In [ ]: