SAS Project (26.5 pts + 1 Bonus)
Due Oct 12

Objectives

- **Part 1: Introduction to SAS**
  - **1.1) Get familiar with SAS**
  - **1.2) DATA statement and proc print**
  - **1.3) infiling and log files**
- **Part 2: Descriptive Statistics**
  - **2.1) Histograms**
  - **2.2) Boxplot**
- **Part 3: Relative Frequency Interpretation to Probability**
  - **3.1) Display long-run convergence**
- **Part 4: Probability Distributions**
  - **4.1) Generate random samples using specific distributions**
  - **4.2) Use samples to gain insight into the relationship between population and sample**
- **Part 5: QQplot**
  - **5.1) Normal Distribution**
  - **5.2) Nonnormal Distributions**
- **Part 6: Sampling Distributions**
  - **6.1) effect of size**
  - **6.2) Central Limit Theorem**

Remember:
a) Please only turn in one paper per group.
a) Please put all of your names, STAT 503 and the project # on the front of the project.
b) Label each part and put them in logical order.
c) ALWAYS include your SAS code at the beginning of each problem.

1.1 Getting Familiar with SAS:

Be sure that you can load the program either using the ITAP computers or via goremote (please see the SAS introduction handout for details) and know how to run a program.

1.2 DATALINES:

Creating datasets in SAS. Remember to always check to see if you have one or two variables.

SAS Learning code: (a1.sas)
Note: From the SAS editor, the commands are color coded:
blue: commands black: responses green: comments blue green: numbers
purple: text/titles

```
data a1; /*creates a data set called a1 */
    *Note: data sets have to start with a letter;
input x y @@; /* define two variables x and y, @@ is used here to
  allow more than one observation per line */
    /*Note: If you only wanted to use one variable, the code would be
      input x @@; */
datalines;
1 1 0 8 1 6 0 1 0 1 2 5
```

```
0 3 1 0 1 0 1 4 2 4 1 0
0 0 0 1 1 2 1 1 0 4 1 0
1 4 1 0 1 3 0 0 0 1 0 1
1 0 1 1 2 3 0 2 1 4 2 6
2 6 1 0 1 1 0 1 2 8 1 3
1 3 0 5 1 0 5 5 0 2 3 3
0 1 1 0 1 0 0 0 0 3
; /* data input completes here, notice that the semicolon must occupy
  one line by itself */
run; /* indicates that when program is submitted, this is the boundary
  of the data step */

proc print data=a1; run;
 /* This actually sends the dataset to the output window */

quit; /*stops the program */
```

SAS Learning Output

The SAS System

Obs	x	y
1	1	1
2	0	8
3	1	6
4	0	1
5	0	1
6	2	5
⋮	⋮	⋮
43	0	1
44	1	0
45	1	0
46	0	0
47	0	3

Problem 1 (1 pts.)
Modify the datalines code (a1.sas) to create a new dataset in SAS using the data from Exercise 2.2.4 (p. 39) in section 2.2:
Name the dataset and variables appropriately for the dataset (see page 31). Points will be taken off if you name the dataset a1 and the variable x.
Your submission should consist of your code and the print out of your dataset.

Problem 2 (1 pts.)
Modify the datalines code (a1.sas) to create a new dataset in SAS using the data from Exercise 2.4.3 (p. 51) where the X values are listed in the problem and all of the Y values are '1':
Name the dataset and variable appropriately for the dataset (see page 30). Points will be taken off if you name the dataset a1 and the variables x and y.
Your submission should consist of your code and the print out of your dataset.

1.3. INFILING:

An alternate way of creating a dataset is to have SAS copy an existing data file on **your** computer, such as a text or data file. To do this you use an infile statement and direct SAS to the location of the file on your computer. For convenience, I suggest you use your H: drive for storing the datasets. If you don't use your H: drive, use the built in 'Explorer' to determine what the directory your files are in. Remember to always check to see if your inflie has one or two variables.

Additionally, SAS keeps a log of what you have submitted thru the SAS system, all of this is shown in the LOG window in SAS. Here SAS will direct you to errors in your SAS code and often it tries to indicate what it thinks you have done wrong if anything. If your SAS code does not run, your first step is to look at the log file to see what happened.

SAS Learning code: (a2.sas)

```
data a2; /* This creates the same dataset as above but using an infile
    statement */
infile 'H:\a2.dat';
input x y;
run;
proc print data=a2;
run;
quit;
```

Example SAS Learning log file:

```
8    data a2; /* This creates the same dataset as above but using an infile statement */
9    infile 'H:\My Documents\Stat 503\SAS\a2.dat';
10   input x y;
11   run;

NOTE: The infile 'H:\My Documents\Stat 503\SAS\a2.dat' is:
      Filename=H:\My Documents\Stat 503\SAS\a2.dat,
      RECFM=V,LRECL=256,File Size (bytes)=272,
      Last Modified=24Aug2009:15:25:26,
      Create Time=24Aug2009:15:25:26

NOTE: 47 records were read from the infile 'H:\My Documents\Stat 503\SAS\a2.dat'.
      The minimum record length was 3.
      The maximum record length was 4.
NOTE: The data set WORK.A2 has 47 observations and 2 variables.
NOTE: DATA statement used (Total process time):
      real time           0.01 seconds
      cpu time            0.00 seconds

12   proc print data=a2;
13   run;

NOTE: There were 47 observations read from the data set WORK.A2.
NOTE: PROCEDURE PRINT used (Total process time):
      real time           0.01 seconds
      cpu time            0.01 seconds
14   quit;
```

Problem 3 (1 pts.)

Consider the number of dendrite branches from 36 nerve cells from brains of newborn guinea pigs presented in Exercise 2.2.4 (p. 39). This data file is called E2.2.4.dat. Modify the SAS infiling code (a2.sas) to infile this dataset into SAS.

Your submission should consist of your code and the log file. Note: your log file should only be for this problem and contain no errors in it.

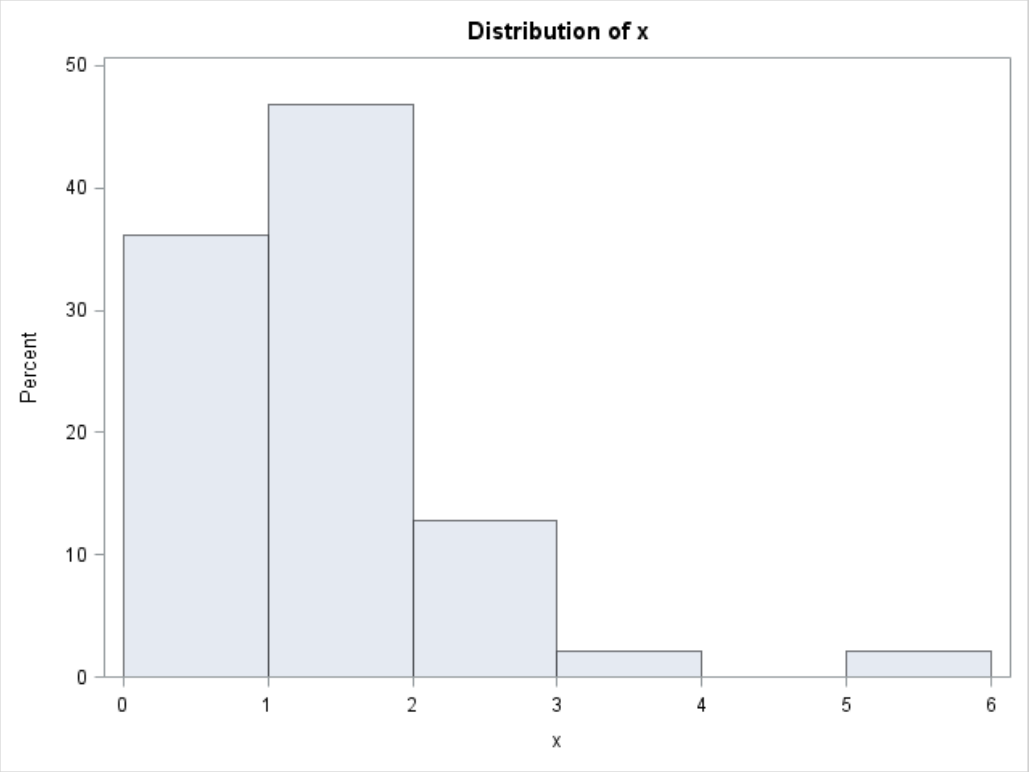
2.1. HISTOGRAMS:

To produce histograms in SAS we use the univariate procedure or “proc univariate”. Procedures are inherent SAS commands which produce specific statistical analysis or output. The univariate procedure is one that handles univariate data, or single variables at a time. Univariate, by default, produces lots of numerical statistics but here we’d only like to use it just to create some histograms. I have added a title to the plot. This should always be done in your labs or homework. Be sure that the title is appropriate for the problem; not just copied from the sample code.

SAS Learning code: (a3.sas)

```
data a2;
infile 'H:\a2.dat';
input x y;
run;
proc print data=a2;
run;

title1 'Example of histograms';
proc univariate data=a2 noprint; /* noprint is used to suppress
   summary statistics*/
histogram x / endpoints = 0 to 6 by 1; /* another histogram for x; by
   1, gives the class width: 0 to 6 gives range for the x-axis  */
run;
quit;
```

SAS Learning Output



Problem 4 (1 pts.)

Produce a histogram that matches the histogram that you created in the homework for Problem 2.2.4. I have removed the SAS code that I previously posted in the key; however, if the code is identical, that is, all of the variable, data set, and titles are the same, you will receive half credit for this problem.

Your submission should consist of your code and histogram.

2.2. BOXPLOTS:

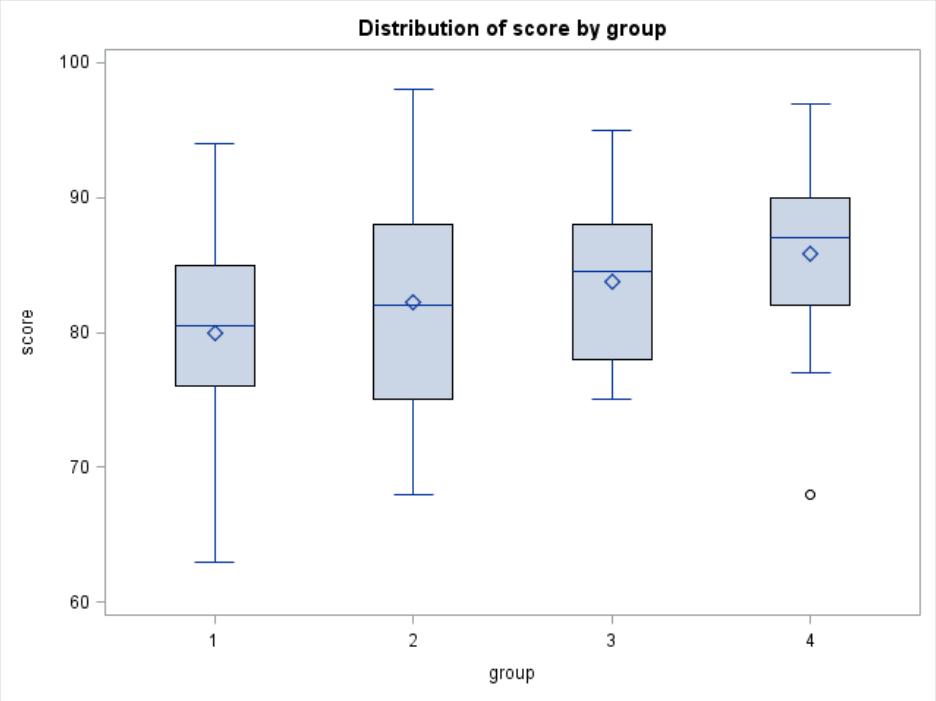
Creating boxplots in SAS.

The "proc boxplot" code requires 2 variables. For side-by-side boxplots, the group variable indicates which of the groups the boxplot is in. However, even if you only have one “group", you need to specify a variable for "group". In the output, the ‘◇’ indicates the location of the mean. If you want to read in a text variable, place a dollar sign ($) after the variable name. The space before the $ is optional. Always check to see what the order of the variables are when they are read in the ‘input’ line. This does not have to be the same as the order of the variables in the proc boxplot.

SAS Learning code: (a4.sas)

```
data a4;
infile 'H:\a4.dat';
input group $ score;
run;

title1 'Sample Boxplot';
proc boxplot data=a4 ;
plot score*group/boxstyle=schematic idsymbol=circle; /* This creates a
   modified boxplot(s) of the score variable for each group in the
   group variable. Note, if there is only one group it will produce a
   single boxplot, if there are multiple groups it will create side-by-
   side boxplots */
run;
quit;
```

SAS Learning Output



Problem 5 (1 pts.)
Consider the study of milk production in sheep presented in Exercise 2.4.3 (p. 51) of our text. Modify the SAS code (a4.sas) to reproduce modified boxplots that you created in your homework. Again, please change the title of the plot. I have removed the SAS code that I previously posted in the key; however, if the code is identical, that is, all of the variable, data set, and titles are the same, you will receive half credit for this problem.
Your submission should consist of your code and the one figure of the boxplot.

Bonus B1 (1 pts.) Consider the Radish Growth using three different light treatments presented in Example 2.5.3 (p. 55) of our text. (dataset Ex2.5.3.dat). Modify the SAS infiling code (a4.sas) to produce side-by-side modified boxplots as in Figure 2.5.3. In the data set for this problem, the second variable is text so be sure to include the $ after it. Again, please change the title of the plot.
Your submission should consist of your code and the one figure of the side-by-side boxplots.

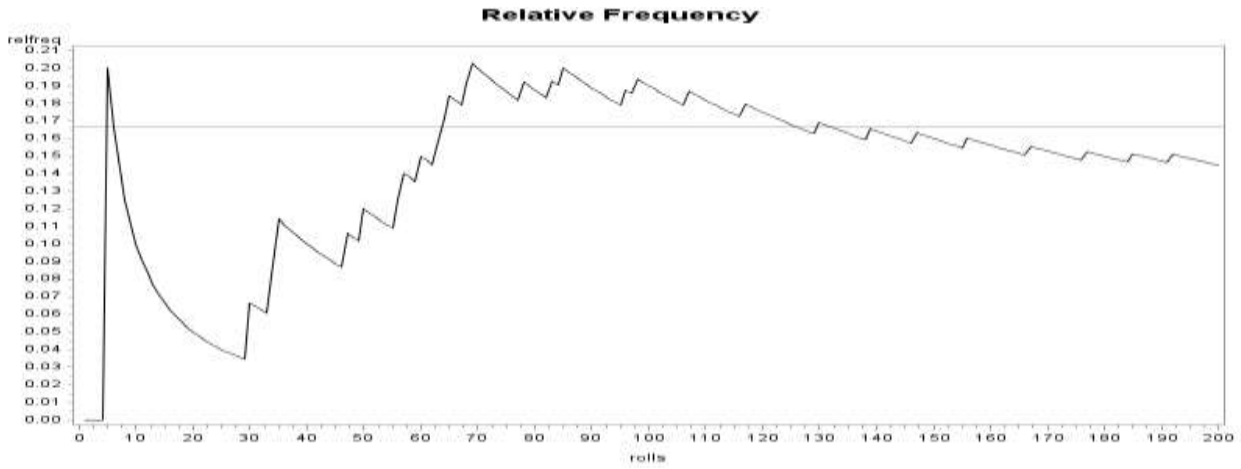
3.1 Long-Run Convergence:

In this objective, we are going to examine the frequentist view of probability; that is, does the long term percentage match the theoretical value. Suppose we are given a six-sided die and would like to determine the probability that someone would roll a 1. To do this, you decide to roll the die 200 times and each time record whether a 1 appeared (1 = Y, 0 = N). To simulate this information, we will use the correct probability of $\frac{1}{6}$ and will simulate rolling the die for 200 rolls. The plot generated will show the relative frequency *so far* as the number of rolls increases. Notice that the relative frequency converges to the probability of 1/6. You can re-run the program to re-generate the plot. To accomplish this, you need to be sure that the editor file (b1.sas) is active.

SAS Learning code: (b5.sas)

```
data converge;
freq = 0;
do rolls = 1 to 200; *roll the dice 200 times;
   x = int(6*rand('uniform')) + 1;  *generates the roll of the dice;
   if x = 1 then freq = freq + 1;  *increases frequency if the roll = 1;
   relfreq = freq/rolls; *calculates the relative frequency;
   output;
end;

proc print data=converge; run;
title1 'Relative Frequency';
symbol1 v=none i=join c=black;
proc gplot data=converge;
   plot relfreq * rolls/vref=0.16666667; *plots the data with a line at 1/6;
run;
quit;
```

SAS Learning Output



Problem 6 (1.5 pts.)
Run the learning code 10 times. How many of the plots stabilize to the correct value?
Your submission should consist of at least one of the plots that converges and at least one of the plots that doesn't converge in addition to the number of plots that stabilize to the correct value. The input code is not required because you are using the learning code with no modifications.

4.1. Generate random samples using specific distributions:

We are going to generate two different random samples; one using the binomial distribution, and the second using the normal distribution. Both learning codes will be provided. However, the output codes will be identical so only one sample will be provided.

a) Binomial Distribution.

Suppose we want to simulate the experiment of flipping a fair coin 40 times. This can be done by the function RAND('Binomial',p,n). We can also repeat this experiment by generating multiple Binomial random variables (a sample).

SAS Learning code: (a6.sas)

```
*random number generation for binomial is RAND('Binomial',p,n);

title1 'Binomial Random Numbers';
data RandomBinomial;
   do i=1 to 20;
      p = 0.5;
      n = 40;
      x1=rand('Binomial',p,n);
      output;
   end;
proc print data=RandomBinomial;
   var x1;
run;

*proc means prints out the mean and standard deviation of the sample data
set;
proc means data=RandomBinomial;
   var x1;
run;
quit;
```

Sample SAS Learning output:

Binomial Random Numbers

Obs	x1
1	21
2	19
3	19
4	17
5	19
6	18
7	22
8	18
9	19
10	20
11	16
12	16
13	21
14	19
15	22
16	25

Obs	x1
17	22
18	14
19	24
20	21

Binomial Random Numbers

The MEANS Procedure

Analysis Variable : x1				
N	Mean	Std Dev	Minimum	Maximum
20	19.6000000	2.7414940	14.0000000	25.0000000

Note: Each time that you run the program, you will get different values and different means and standard deviations.

b) Normal Distribution.

Suppose we want to simulate the experiment of observing 20 interspike-time intervals as described on p. 122 of your text (Ex. 4.1.3) and want to do this 3 times. From the problem μ = 15.6 and σ = 0.4. This can be done by the function RAND('Normal',mu,sigma).

SAS Learning code: (a7.sas)

```
*random number generation for normal is RAND('Normal',mu,sigma);

title1 'Normal Random Numbers';
data RandomNormal;
   do i=1 to 20;
      mu = 15.6;
      sigma = 0.4;
      x1=rand('Normal',mu,sigma);
      output;
   end;
proc print data=RandomNormal;
   var x1;
run;

proc means data=RandomNormal;
   var x1;
run;
quit;
```

Problem 7 (3 pts.)

a) Run b4.sas three times. For each of the times, compare the calculated mean and standard deviation with the true values by stating the difference between each of the values and the true value.

b) Run b5.sas three times. For each of the times, compare the calculated mean and standard deviation with the true values by stating the difference between each of the values and the true value.

Your submission should consist of the output from the means for each of the 6 runs (no printout of the generated data set) in addition to the information requested above which can be hand written or typed next to the corresponding output. The input code is not required because you are using the learning code with no modifications.

5.1 QQplots: Normal

A Normal QQPlot can be used to visually assess the Normality of a data set. The Normal QQplot plots the observed data against the corresponding z-score which is calculated using the percentiles of the standard Normal. If the data is Normal, the points fall approximately along a line. The problem is that it is hard to determine what 'approximate' is. The QQplot code is a simple command line within proc univariate.

I have added in the histogram plot so you can visually look at the density curve. The blue line is the normal distribution with the estimated μ and σ; the red line is a smoothed curve of the histogram itself.

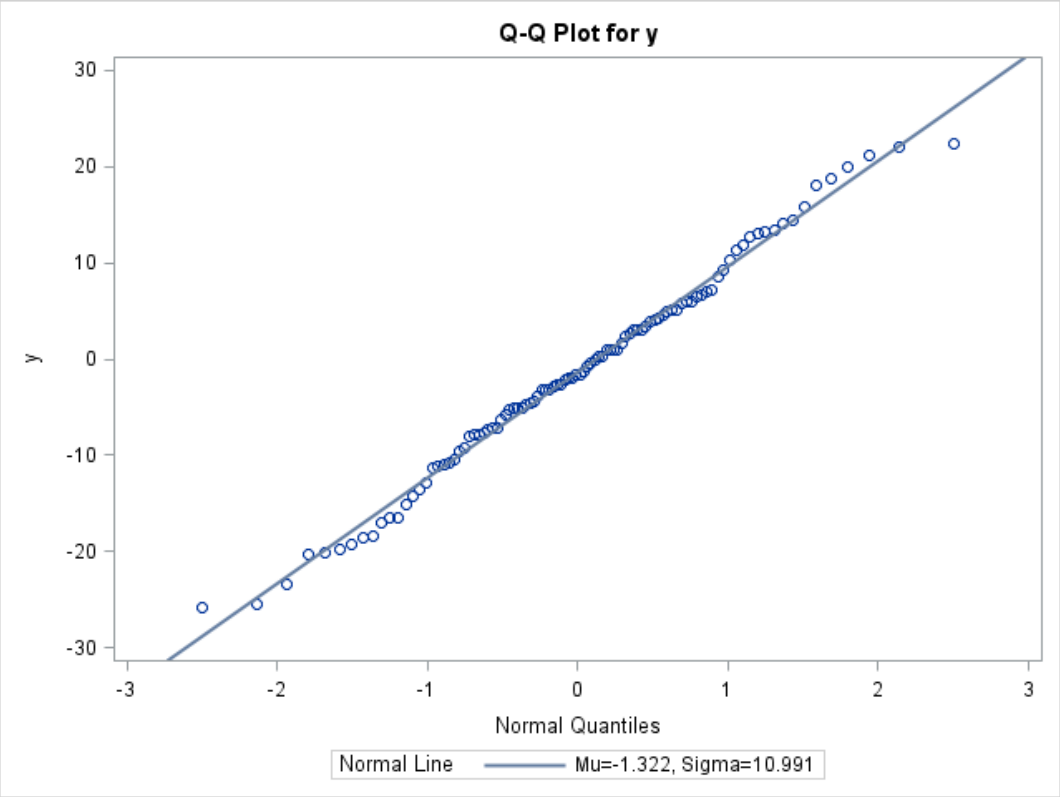
SAS Learning code: (a8.sas)

```
*Normal distribution, mu=0, sigma = 10;
title1 'Normal Distribution';

%Let points = 100; *this is the number of data points in the sample;
%Let mu = 0; *mu = 0 not changing in this question;
%Let sigma = 10; *sigma = 10 not changing in this question;
data norm;
   do x=1 to &points;
      y=rand('normal',&mu,&sigma); output;
   end;

title1 'QQplot Normal Distribution, n = ' &points;
symbol1 v=circle;
proc univariate data=norm;
histogram y/ normal kernel (L=2);
qqplot y/normal (mu=est sigma=est) ;
run;

quit;
```

SAS Learning Output

QQplot Normal Distribution, n = 100

The UNIVARIATE Procedure



Problem 8 (2.5 pts.)

Run a8.sas 3 times with each of 10, 100 and 1000 data points for a total of 9 different times. For each of the runs, save the QQplot. All of these QQplots are from a normal distribution, therefore, all of them form a 'good' line. Which number of data points has the 'best' line? Hint: Besides looking at the line itself, look at the estimated value of μ and σ. Which is the 'worst'? If you would have just looked at the 'worst' line, would you think it came from a normal distribution if you didn't know that it came from a normal distribution? Why or why not?

Your submission should consist the 'best' line from each of the 3 different number of data points from the QQPlot generated from SAS and the answer to the questions above.

5.2 QQplots: Not Normal

It is helpful in judging whether a distribution is normal or not to see examples of cases where it is not normal. The only difference between the previous learning code and the learning codes in this objective are the type of random number used.

right skewed: exponential distribution ($\lambda = 2$) with $\mu = 0.5$ and $\sigma = 0.5$

left skewed: Beta distribution (on [0,1], $\alpha = 2$, $\beta = 0.5$) with $\mu = 0.8$ and $\sigma = 0.0457$

long tailed: Standard Cauchy with median = 0 and σ = ?

short tailed: Uniform (on [0.2]) with $\mu = 1$ and $\sigma = 0.3333$

I have included the histogram again so you can be assured that the distribution is what I say it is.

SAS Learning code: (a9.sas)

```
*nonnormal distributions
 right skewed: exponential distribution (lambda=2) with mu=0.5 and sigma=0.5
 left  skewed: Beta distribution (on [0,1],  alpha = 2, beta = 0.5)
        with mu = 0.8 and sigma = 0.0457
 long tailed: Standard Cauchy with median = 0 and sigma = ?
 short tailed: Uniform (on [0,2]) with mu = 1 and sigma = 0.3333

*Normal distribution, mu=0, sigma = 10;
title1 'Normal Distribution';

%Let points = 100; *this is the number of data points in the sample;
%Let right = exp(2)*rand('Exponential');
%Let left = rand('Beta',2,0.5);
%Let long = rand('Cauchy');
%Let short = 2*rand('Uniform');

data notnorm;
   do x=1 to &points;
      y=&right; output; *change this to &left, &long, &short when
         appropriate;
   end;

title1 'QQplot Distribution Right Skewed'; *change this to
   left skewed, long tailed or short tailed as appropriate;
symbol1 v=circle;
proc univariate data=notnorm;
histogram y/ normal kernel (L=2);
qqplot y/normal (mu=est sigma=est) ;
run;

quit;
```

Problem 9 (4.5 pts.)

Run a9.sas 3 times with each of the different not normal distributions. Describe the shape of the QQplot for each of the types of distributions. Go back to the previous problem; do any of the 9 QQplots display a not normal distribution.

Your submission should consist of the most represented line for each of the distributions with a description of the line. In addition, if any of the 'Normal' QQplots display a non-normal distribution, present that with the distribution that it represents.

6.Sampling Distributions:

In this objective, we will use the the random number generation tool to look at the sampling distribution of the sample mean and sample proportion. This allows a visual inspection of how well the Normal distribution fits the randomly generated data. The larger the sample size, the better the overall fit.

We will be using both a normal distribution and a uniform distribution. For a uniform distribution, the theoretical values for the population mean is ½ and for the population standard deviation is $\sqrt{\frac{1}{12}}$.

SAS Learning code: (a10.sas)

```
*Code modified from from www.stanford.edu/~kcobb/hrp259/lab1_EG.doc
6/20/2012: 4 pm;
%LET repeats=1000; *we are not changing this in this question;
%LET n=1; *you will change this value in this question;
%Let uni = rand('Uniform'); *you will choose either &uni or &norm as
appropriate;
%Let norm = rand ('Normal',0,1);
data test;
do j=1 to &repeats by 1;
            avg=0;
            do i=1 to &n by 1;
                  avg=avg+&norm;
           end;
        avg=avg/&n;
      output;
      drop i;
end;
run;

title1 'Normal Distribution repeats = ' &n; *Change to Uniform when
appropriate;
proc means data=test; *prints out means and standard deviations;
   var avg;
run;

proc univariate data=test noprint;
      histogram avg/  normal kernel(L=2);
run;

quit;
```

Problem 10 (6 pts.)

Run the preceding learning code for n = 1, n = 2 and n = 6 for both normal distribution and the uniform distribution

Your submission should consist of the six generated histograms, the six outputs from proc means, two tables consisting of the three means from the output and the theoretical value and the three standard deviations from the output and the theoretical value for both the normal distribution and the uniform distribution. The theoretical value for the standard deviation for each run is the standard error given the appropriate value of n. Does the calculated values match the theoretical values? In addition, please indicate which of the graphs look like they are a normal distribution (this can be written directly on the output). No code is required because you are making only minor modifications to the learning code.