Fall 2010, STAT 430
SAS Examples SAS7
==========

ssh bnk@glue.umd.edu, tap sas913 (or sas82), sas
https://www.statlab.umd.edu/sasdoc/sashtml/onldoc.htm

0. Do Loops
a. Generation of RV's
b. Demonstration of LLN using simulated data
c. Demonstration of CLT using simulated data
d. Proc SUMMARY

0. DO Loops

DO Loops are used when a statement or set of statements needs to be repeated many times. DO loops begin with a DO statement and end with an END statement. The statements in between are repeated the number of times indicated by the index variable in the DO statement.

Example
==========

```
OPTION PS=35 LS=70;

DATA DoLoop;
DO i = 1 TO 100;
x = 3 + 5* RANNOR(59); /* N(3,25) */;
OUTPUT;
END;
RUN;
```

Can apply PROCs to x:

```
proc means data=DoLoop;  <--- The generated data are here!!!
output out=try mean=Mean;
var x;
run;
```

This gives the sample mean and SD of the N(3,25) sample of size 100!!!

The MEANS Procedure

Analysis Variable : x

N	Mean	Std Dev	Minimum	Maximum
100	2.6718930	5.0943355	-8.1962986	16.2624590

OR Which is the same:

```
proc means data=DoLoop;
output;
var x;
run;
```

OR Which is the same:

```
proc means data=DoLoop;
var x;
run;
```

OR Which is the same:

```
/* SAS automatically outputs the data into WORK.DOLOOP */;
proc means data=WORK.DOLOOP;
var x;
run;
```

From the Log Window:
"NOTE: There were 100 observations read from the data set
WORK.DOLOOP."

```
proc print  data=WORK.DOLOOP;
run;
```

The SAS System

```
 Obs     i           x

   1     1      3.1979
   2     2      1.2737
   3     3      5.6871
.....................
.....................
  98     98     2.89315
  99     99    -4.35282
 100    100    -0.27217
```

```
NOTE: the new data set "try" contains only ONE obs "Mean".
If we want to see what is in the data set "try" use:

proc print data=try;
run;
```

Obs	_TYPE_	_FREQ_	Mean
1	0	100	2.67189

```
or:

proc print data=try;
var Mean;
run;
```

Obs	Mean
1	2.67189

```
Example: Verify Cody and Smith p. 356
=====================================

DATA DoLoop;
```

```
DO i = 1 TO 100;
x = 1 + 99* RANUNI(0);
OUTPUT;
END;
RUN;
```

Can apply PROCs to x:

```
proc print data=DoLoop;  <--- The generated data are here!!!
var x;
run;
```

Get data from Unif(1,100)!!!

Obs	x
1	78.0351
2	85.2269
3	21.7456
4	15.7471
5	51.2031
.	.
.	.
.	.
95	19.3039
96	35.7787
97	10.1345
98	28.2734
99	2.7773
100	44.4092

a. Generation of RV's
==========

SAS can generate RV's from a given dist. Some examples:

---

ranbin(seed,n,p)	Binomial distribution b(n,p)	--------

ranexp(seed)	Exponential distribution Exp(1)
rannor(seed)	Standard normal N(0,1)
ranpoi(seed,m)	Poisson distribution Poisson(m)
ranuni(seed)	Uniform distribution Unif(0,1)

Example: Generate N(0,1) and LN(0,1) data
==========

```
DATA random;
DO i = 1 TO 100;
r = RANNOR(0);
y=exp(r); /* Lognormal data */;
output;
END;
RUN;

/* OK!!! */
/* SAS automatically outputs the data into WORK.RANDOM */;
proc print data=WORK.RANDOM;
id i;
run;
```

r ~ N(0,1), y = exp(r) ~ LN(0,1)

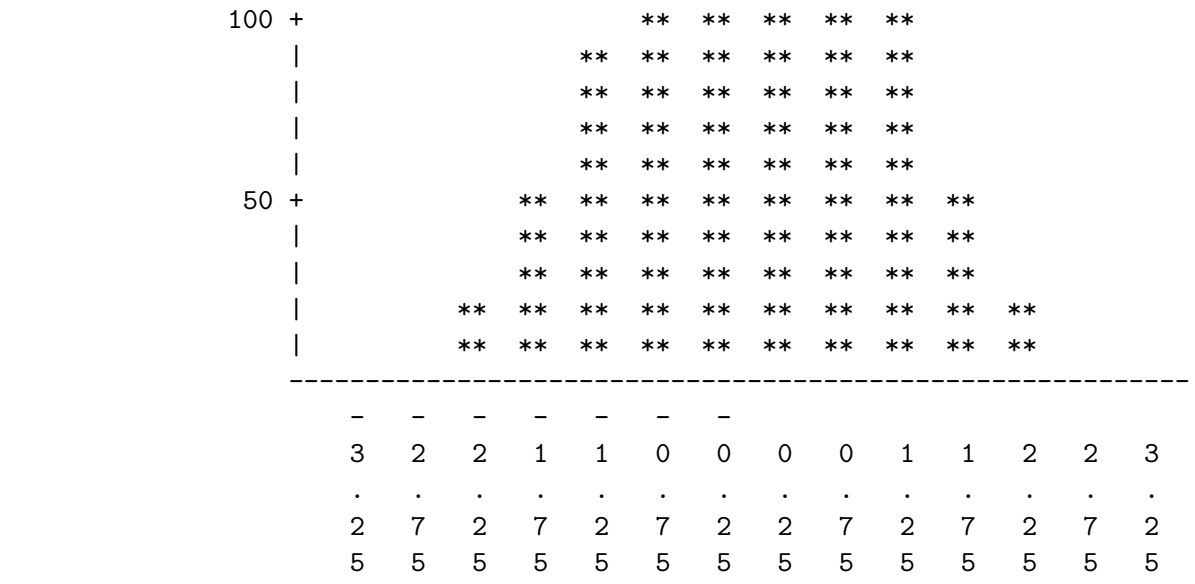
The SAS System

i	r	y
1	1.98409	7.27239
2	-0.62146	0.53716
3	0.75260	2.12251
4	-1.41778	0.24225
5	0.23668	1.26704
6	1.95064	7.03318

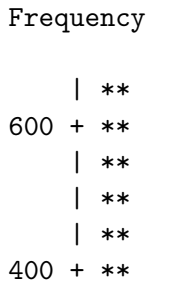
```
                        7     1.24842    3.48485
                        8     0.19324    1.21317
                        9    -1.21987    0.29527
                       10    -0.41083    0.66310
```

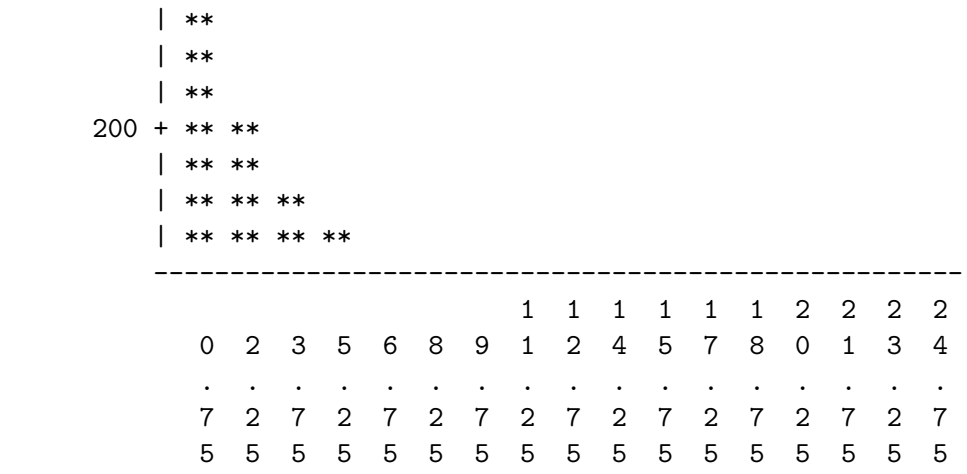
```
/* Now generate 1000 N(0,1) and get the histogram */;
DATA random;
DO i = 1 TO 1000;
r = RANNOR(0);
y=exp(r);
output;
END;
RUN;

/* Nice histogram on GUI using GCHART */;
proc gchart data=WORK.RANDOM;
var r;
run;

/* Simpler using CHART */;
proc chart data=WORK.RANDOM;
var r;
run;
```

```
                              The SAS System

   Frequency

       |                           **  **
       |                           **  **
       |                           **  **
       |                           **  **
   150 +                           **  **  **
       |                         **  **  **  **
       |                         **  **  **  **
       |                         **  **  **  **
       |                         **  **  **  **  **
```




```
/* Histogram of the lognormal data */;
proc chart data=WORK.RANDOM;
var y;
run;
```

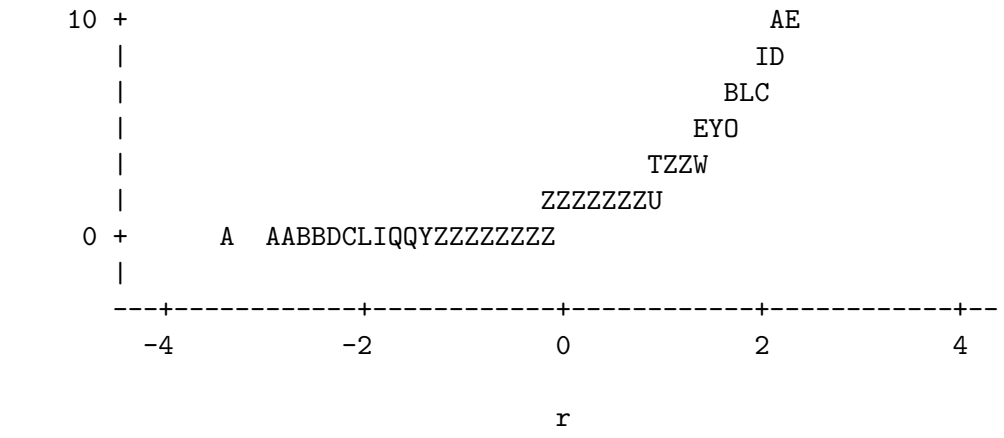





```
proc plot data=WORK.RANDOM;
plot y*r;
run;
```

Plot of y*r. Legend: A = 1 obs, B = 2 obs, etc.

```
 y |
   |
30 +
   |
   |
   |                                                  A
   |                                                  A
   |
20 +
   |
   |
   |
   |
   |                                            B
```



NOTE: 315 obs hidden.

b. Demonstration of LLN using simulated data

1. Demonstration of LLN with iid exp(1) r.vs.

```
options nodate nonumber;
data EXPO;
do sample=1 to 2; /*Number of runs*/;
do n=1 to 100;    /*Number of obs in each run*/;
y = ranexp(7);
output;
end;
end;
run;

proc means data=EXPO;
output out=EXPLLN mean=Mean;
var y;
by sample;
run;
```

```
Results from samples of size N from Exp(1)
==========================================

Note: SD of Exp(1) is 1.

                        The MEANS Procedure

                      Analysis Variable : y

   N            Mean         Std Dev         Minimum         Maximum
 ------------------------------------------------------------------
 100       0.9697695       0.9767435       0.0058798       4.5488584
 ------------------------------------------------------------------
 100       0.9389630       0.8614748       0.0013313       4.1462119
 ------------------------------------------------------------------
10000      0.9986598       0.9980727     0.000056625      10.1128538
------------------------------------------------------------------
10000      1.0040726       1.0006146     0.000030292       9.6453046
------------------------------------------------------------------
1000000    1.0013987       1.0018283     4.2375178E-8     14.4117537
 ------------------------------------------------------------------
1000000    0.9991823       0.9975673     1.4072294E-6     15.3075459
 ------------------------------------------------------------------


2. Demonstration of LLN with iid Unif(0,1) r.vs.

options nodate nonumber;
data UNIF;
do sample=1 to 2; /*Number of runs*/;
do n=1 to 10000;  /*Number of obs in each run*/;
y = RANUNI(0);
output;
end;
end;
run;

proc means data=UNIF;
output out=UNIFLLN mean=Mean;
```

```
var y;
by sample;
run;


Results from samples of size N from Unif(0,1)
=============================================

Note: SD of Unif(0,1) is sqrt(1/12)=0.2886751

                        The MEANS Procedure

                      Analysis Variable : y

   N            Mean         Std Dev         Minimum         Maximum
 ------------------------------------------------------------------
 100       0.5385640       0.3179112       0.0013313       0.9980128
 ------------------------------------------------------------------
 100       0.5622033       0.2852505       0.0246569       0.9936232
 ------------------------------------------------------------------
10000      0.5008272       0.2879409     0.000022463       0.9999637
------------------------------------------------------------------
10000      0.4980235       0.2852955     0.000035856       0.9998519
------------------------------------------------------------------
1000000    0.5000877       0.2886437     1.6298145E-7      0.9999996
------------------------------------------------------------------
1000000    0.5003632       0.2886714     4.2328611E-7      0.9999991
------------------------------------------------------------------

3. Demonstration of LLN with iid N(0,1) r.vs.

options nodate nonumber;
data NORM;
do sample=1 to 2; /*Number of runs*/;
do n=1 to 10000;  /*Number of obs in each run*/;
y = RANNOR(0);
output;
end;
end;
run;
```

```
proc means data=NORM;
output out=NORMLLN mean=Mean;
var y;
by sample;
run;
```

Results from samples of size N from N(0,1)

The MEANS Procedure

Analysis Variable : y

N	Mean	Std Dev	Minimum	Maximum
100	-0.0277138	0.9218208	-2.2105632	2.3709443
100	-0.0957943	1.0360914	-2.9719010	2.3719339
10000	-0.0209077	1.0133153	-3.7308816	3.9232886
10000	-0.0081349	0.9902589	-3.3161692	4.1345405
1000000	-0.0011122	1.0002769	-4.6627020	5.0426949
1000000	-0.000215533	0.9997978	-4.8857179	4.8376764

c. Demonstration of CLT using simulated data

TO Generate 10000 sampleS of size 100 from an exponential distribution and then generate a histogram of the sample means,‘Y.

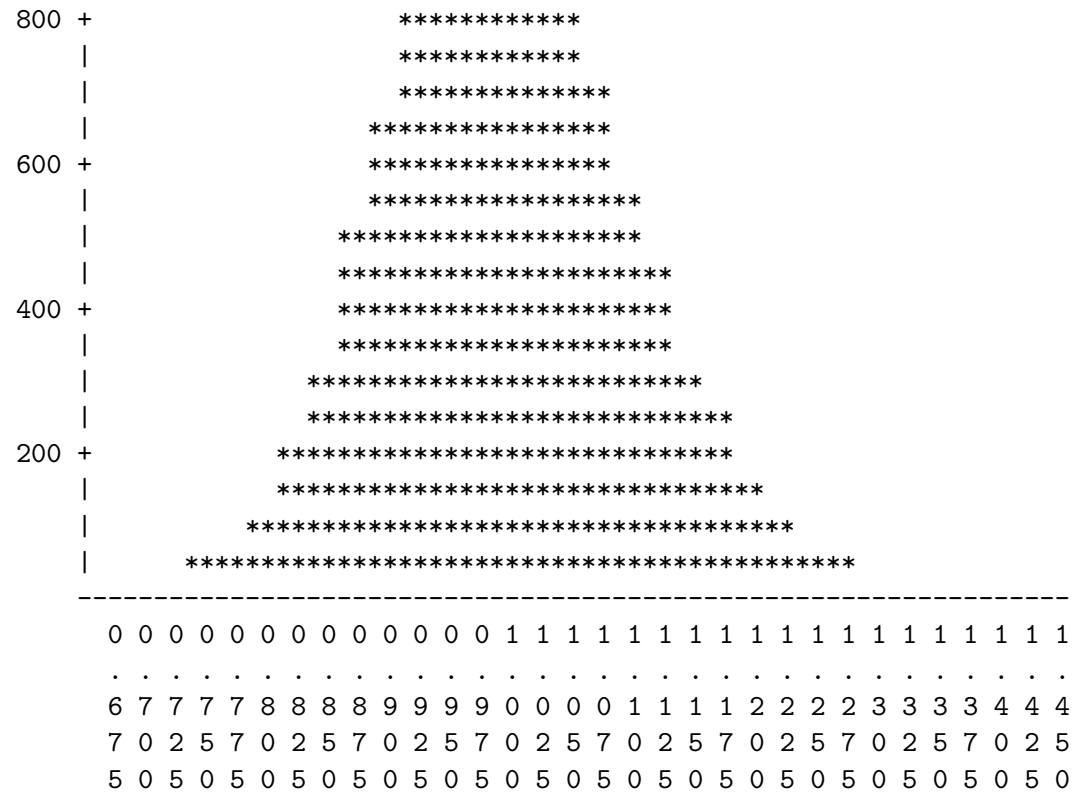
```
OPTION PS=35 LS=70;
```

```
options nodate nonumber;
data one;
do sample=1 to 10000;
do n=1 to 100;
y = ranexp(4637);
output;
end;
end;
run;

proc means data=one noprint;
output out=new mean=Mean;
var y;
by sample;
run;

proc gchart data=new;
vbar mean / space=0;
title'Sampling distribution of ybar';
run;
quit;
```

Simpler to use PROC CHART not GCHART:

```
proc chart data=new;
vbar mean / space=0;
title'Sampling distribution of ybar';
run;
quit;
```

```
                    Sampling distribution of ybar                    1


Frequency

1000 +                         ****
     |                         ******
     |                        ********
     |                        **********
```



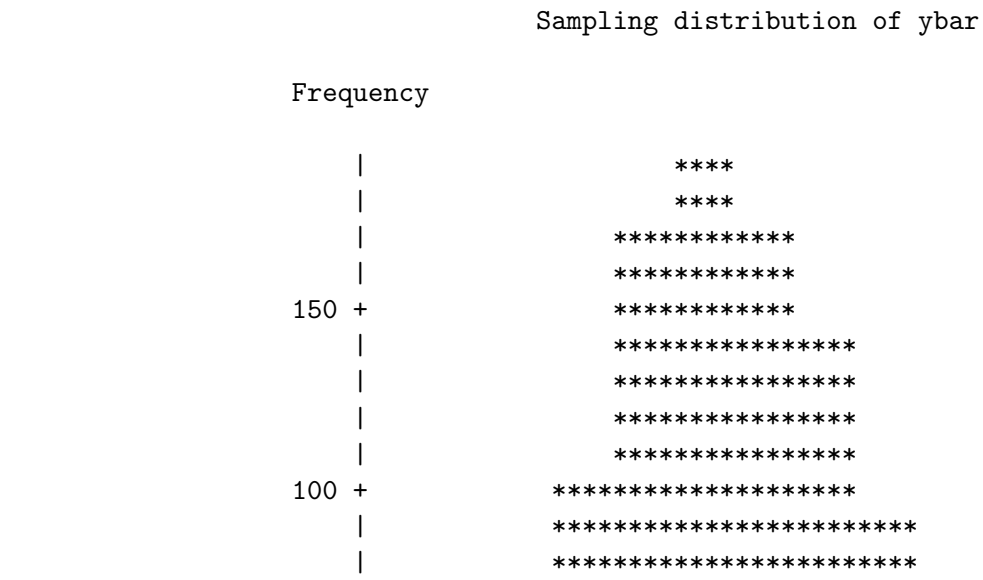
Do Again with different seed.

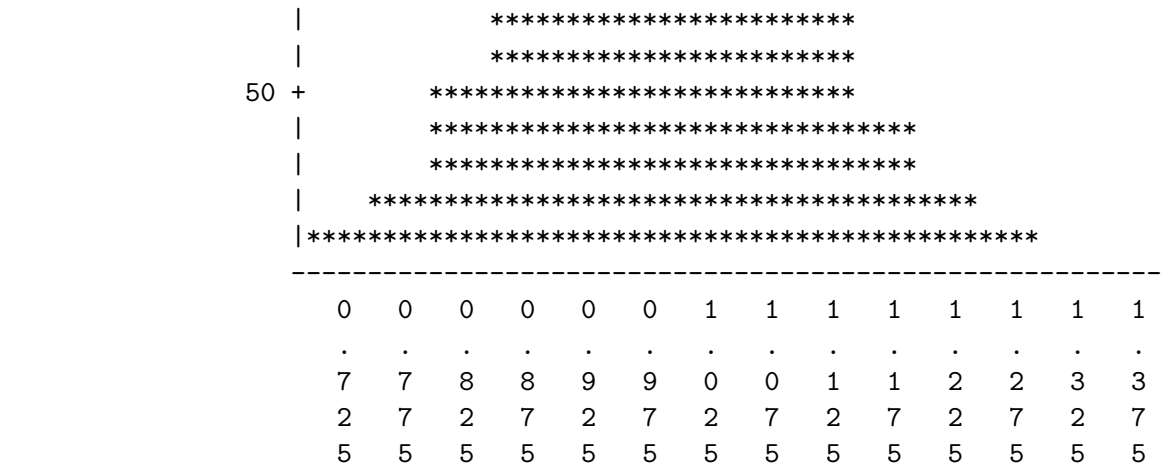
```
OPTION PS=35 LS=70;

options nodate nonumber;
data TWO;
do sample=1 to 1000;
do n=1 to 100;
y = ranexp(2);
output;
end;
end;
run;
```

```
proc means data=TWO noprint;
output out=YDATA mean=Mean;
var y;
by sample;
run;
```

Simpler to use PROC CHART not GCHART:

```
proc chart data=YDATA;
vbar mean / space=0;
title'Sampling distribution of ybar';
run;
quit;
```





Better to use PROC SUMMARY to demostrate CLT
==========

EX 1: Generate 500 samples of size 10 from Poisson(2), then consider
the 500 sample sums and means, min,max, etc.

```
OPTION PS=35 LS=70;
options nodate nonumber;
data POISSON;
do sample=1 to 500; /*Number of runs*/;
do n=1 to 10;  /*Number of obs in each run*/;
y = RANPOI(0,2); <--- Seed=0, Poisson parameter lambda=m=2.
output;
end;
end;
run;
```

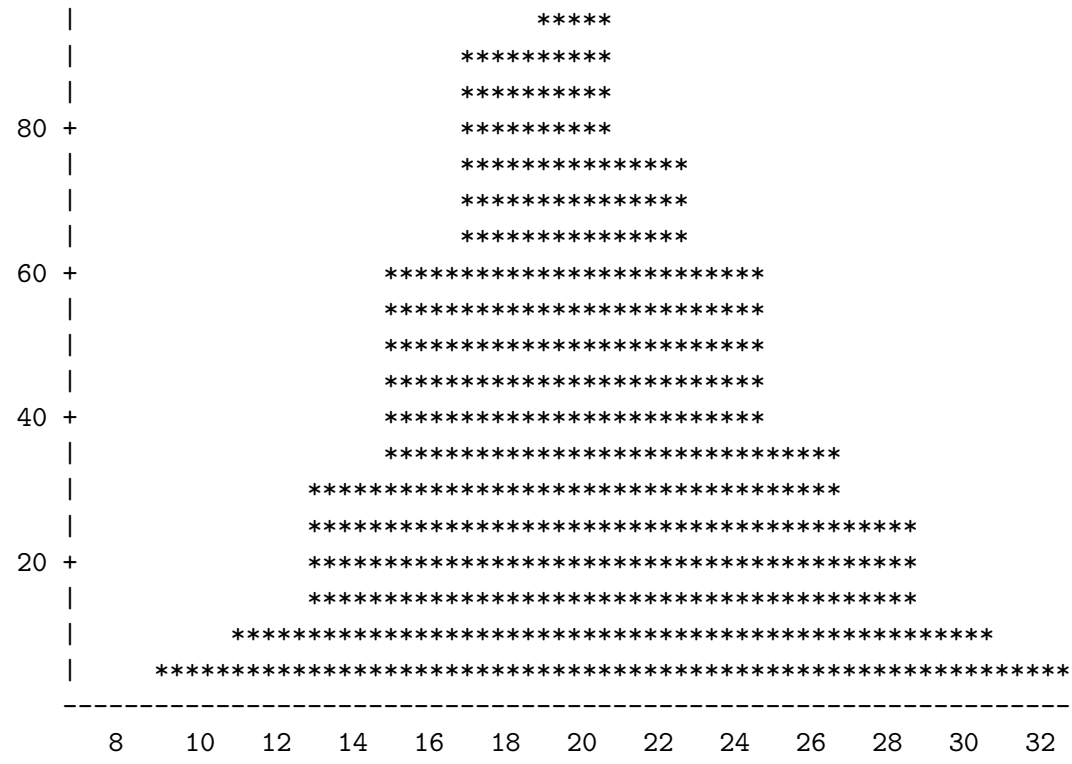
First look at the sample sums.

```
proc summary data=POISSON nway;
```

```
class sample;
var y;
output out=newdata sum=sumy;
run;

proc print data=newdata;
run;

OK! Get sums around 20 since
E(sumy)=E(X_1+...+X_10)=20!!! for Poisson(2)!!!
```

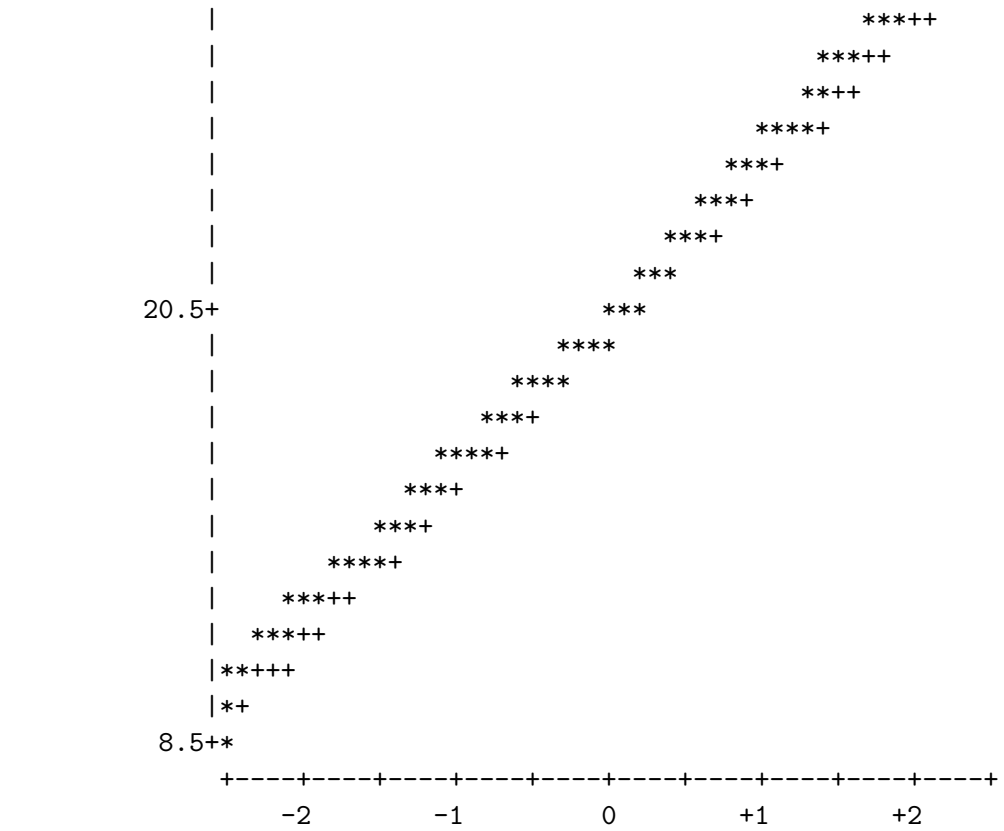
Obs	sample	_TYPE_	_FREQ_	sumy
1	1	1	10	18
2	2	1	10	17
3	3	1	10	21
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.
498	498	1	10	19
499	499	1	10	20
500	500	1	10	18

```
proc chart data=newdata;
vbar sumy / space=0;
title'Sampling distribution of sumy';
run;

OK! Histogram is centered at 20!!! approx.

                  Sampling distribution of sumy

Frequency
```



Test normality with normal probability plot using PROC UNIVARIATE!!!

```
/*normal,box,stem leaf plots using UNIVARIATE*/;
proc UNIVARIATE data=newdata normal plot;
var sumy;
run;
```



Also look at Box plot and stem and leaf. Not bad!!!

```
              Histogram                #             Boxplot
    32.5+**                            3                |
        .*                             2                |
        .*                             2                |
        .***                           6                |
        .******                       12                |
        .*******                      13                |
        .*******                      13                |
        .************                 24                |
        .****************             32                |
        .**************               28              +-----+
```

```
      .******************              38             |     |
      .******************              35             |     |
   20.5+**********************         46             *-----*
      .************************        51             |  +  |
      .******************************  60             |     |
      .***************               29             +-----+
      .******************            35                |
      .************                  23                |
      .*********                     17                |
      .********                      15                |
      .****                           7                |
      .**                             4                |
      .**                             3                |
      .*                              1                |
    8.5+*                             1                |
         ----+----+----+----+----+----+
         * may represent up to 2 counts
```

Also look at basic statistics:

Basic Statistical Measures

Location		Variability		
Mean	19.95000	Std Deviation	4.28301	
Median	20.00000	Variance	18.34419	<-- OK!!!
Mode	18.00000	Range	24.00000	
		Interquartile Range	6.00000	

Now do the same with the 500 sample means. Recall each sample is of size 10 from Poisson(2).

```
proc summary data=POISSON nway;
class sample;
var y;
output out=newdata mean=meany; <--- Use the same name "newdata"!!!
run;                               but we dont have to!!!
```

```
proc print data=newdata; <-- Get also _FREQ_
run;

proc print data=newdata; <-- This is good enough!!! Only sample means!!!
var meany;
run;
```

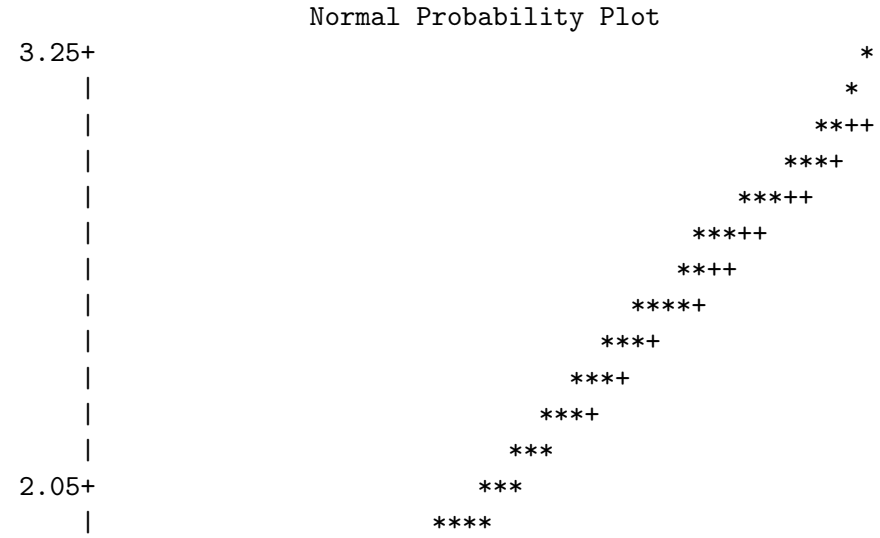
Obs	meany	
1	1.8	
2	1.7	
3	2.1	
4	2.4	
5	1.5	
6	2.8	
7	2.0	<-- OK!!! Since E(Xbar)=2!!!
8	1.8	
9	2.0	
10	2.0	
11	2.2	
12	1.5	
.	.	
.	.	
.	.	
497	3.2	
498	1.9	
499	2.0	
500	1.8	

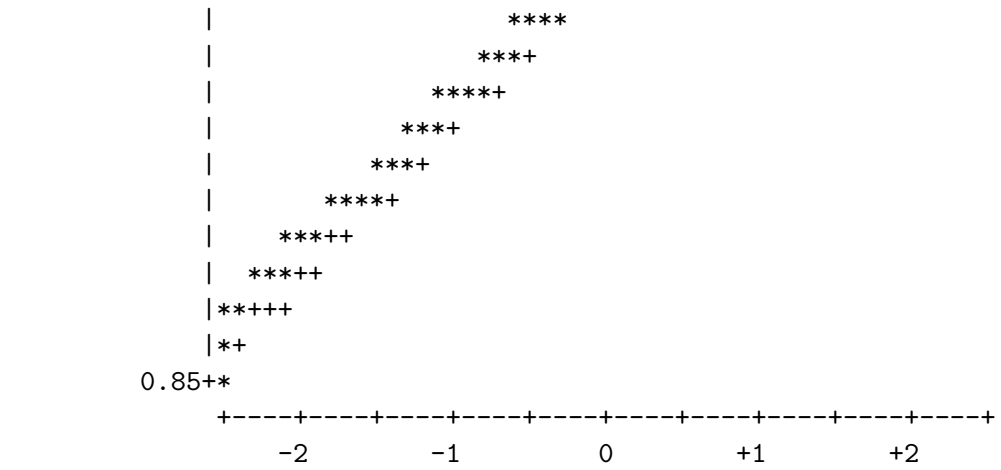
```
Now apply PROC UNIVARIATE to "var meany" in "data=newdata"!!!

/*normal,box,stem leaf plots*/;
proc UNIVARIATE data=newdata normal plot;
var meany;
run;

OK! Mean about 2, var about 2/10!!!
```

Basic Statistical Measures

Location		Variability	
Mean	1.995000	Std Deviation	0.42830
Median	2.000000	Variance	0.18344
Mode	1.800000	Range	2.40000
		Interquartile Range	0.60000

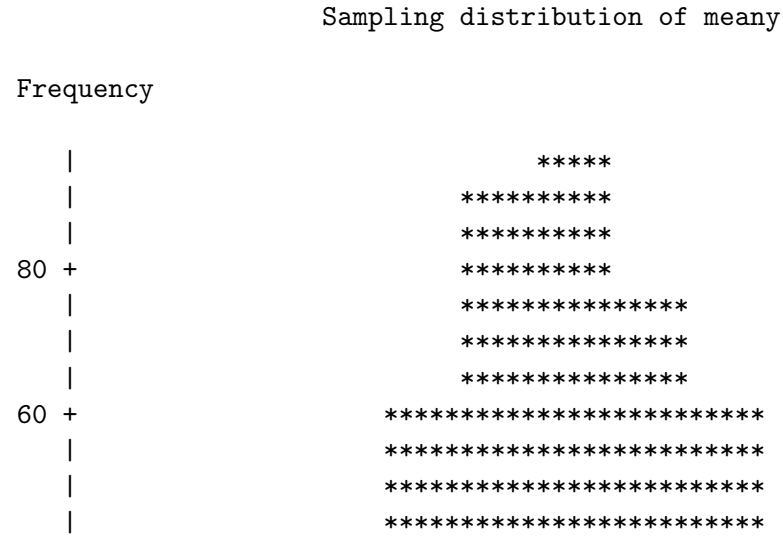
Box plot, normal prob plot, stem and leaf the same as above except for the constant. For example:

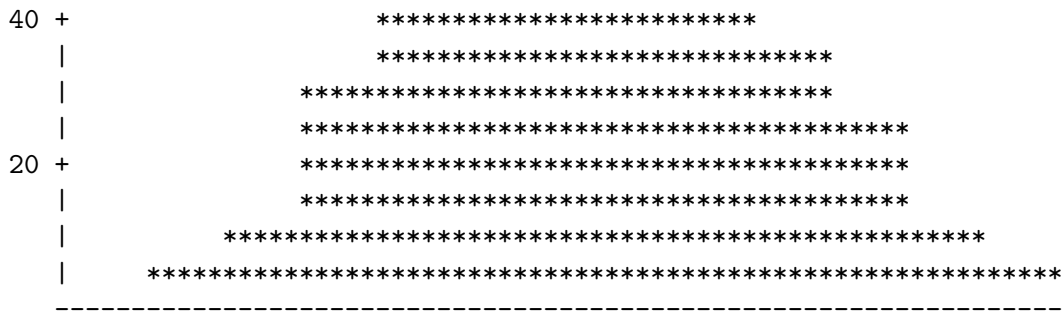





```
proc chart data=newdata;
vbar meany / space=0;
title'Sampling distribution of meany';
run;
```

Again get the same thing but centered at 2!!!

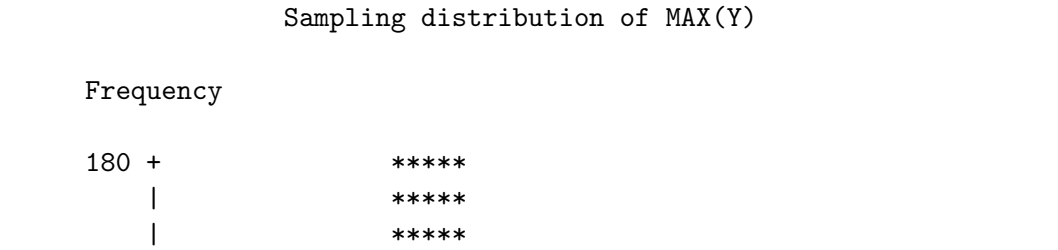


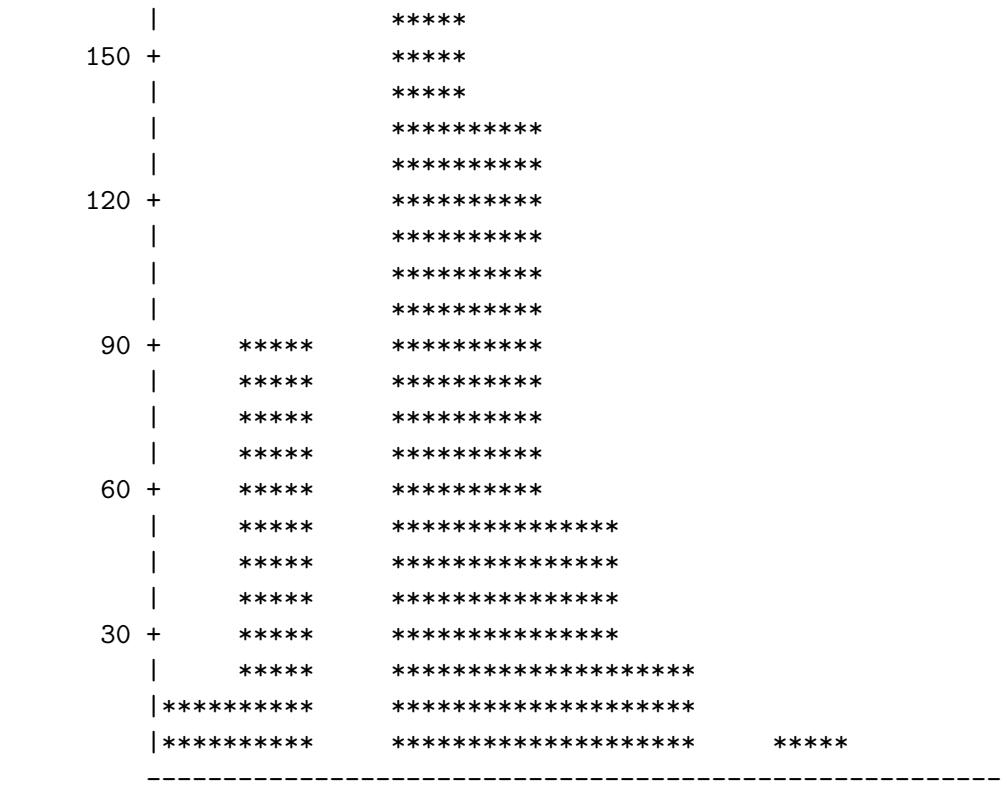


Now do the same with the 500 sample maxima. Recall each sample is of size 10 from Poisson(2).

```
proc summary data=POISSON nway;
class sample;
var y;
output out=newdata max=maxy; <--- Use the same name "newdata"!!!
run;


proc chart data=newdata;
vbar maxy / space=0;
title'Sampling distribution of MAX(Y)';
run;
```

Histogram of Maxy=Max(X_1,...,X_10) is far from normal!!!

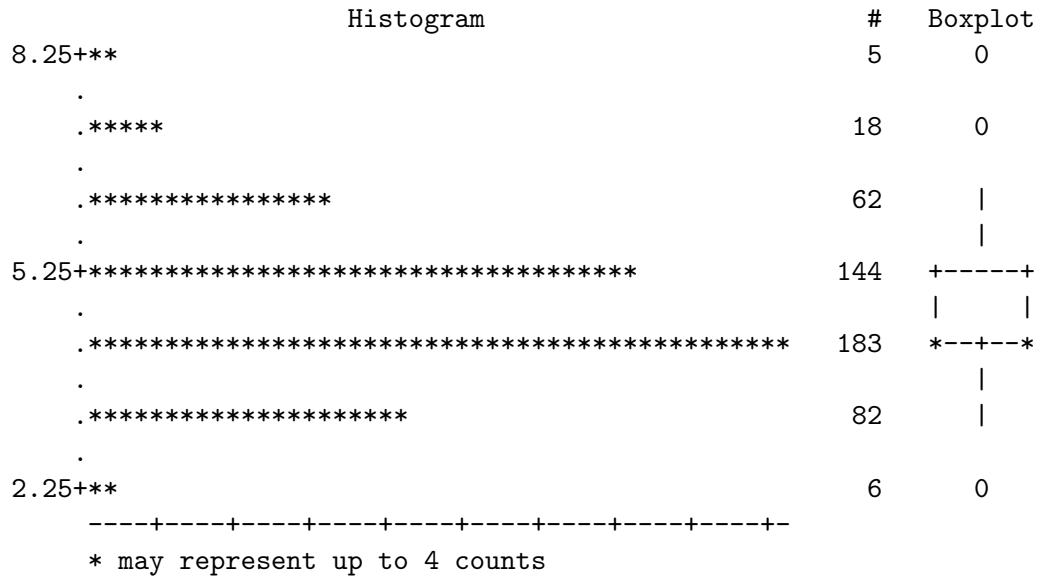


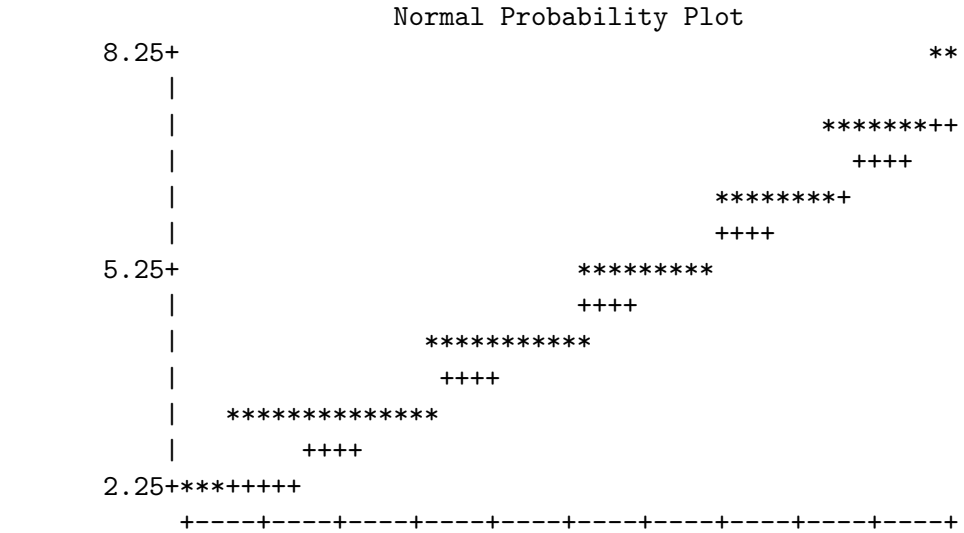



```
/*normal,box,stem leaf plots for sample maxima/;
proc UNIVARIATE data=newdata normal plot;
var maxy;
run;
```

Basic Statistical Measures

Location		Variability	
Mean	4.496000	Std Deviation	1.11196
Median	4.000000	Variance	1.23646
Mode	4.000000	Range	6.00000
		Interquartile Range	1.00000


Histogram
#
Boxplot
8.25+**
5
0
.*****
18
0
.****************
62
|
5.25+**********
144
+-----+
.**********
183
--+--
.**********
82
|
2.25+**
6
0
* may represent up to 4 counts


Normal Probability Plot
8.25+
5.25+
2.25+

```
                -2        -1         0        +1        +2


Get results about sums, means, maxima all at the same time
------------------------------------------------------------
Use: "mean(y)=meany sum(y)=sumy max(y)=maxy" in
PROC SUMMARY. But also "mean=meany sum=sumy max=maxy" works!!!


OPTION PS=35 LS=70;
options nodate nonumber;
data POISSON;
do sample=1 to 500; /*Number of runs*/;
do n=1 to 10;  /*Number of obs in each run*/;
y = RANPOI(0,2); <--- Seed=0, Poisson parameter lambda=m=2.
output;
end;
end;
run;




Get sums, means, maxima all at the same time

proc summary data=POISSON nway;
class sample;
var y;
output out=newdata mean(y)=meany sum(y)=sumy max(y)=maxy;
run;


proc UNIVARIATE data=newdata normal plot;
var meany sumy maxy;
run;

Will give all the results for  meany sumy maxy.
```