Results for the Envision Priming Study.

Overview of the findings:

> The 8 credits primed with high score examples, the p-value is 0.0005. Primed group scored on average 30 points more. Really significant!! Very unlikely due to chance.
>
> The 2 credits WITHOUT an example, the p-value is 0.05. Non primed group scored on average 5.4 points more. Interesting that the non primed group scored higher!
>
> Credit QL1.3 with a negative example, the p-value is 0.02. Non primed group scored on average 2 points more. Also, interesting the non primed group scored higher!

In [1]:
```
import numpy as np
import pandas as pd
```

In [2]:
```
%matplotlib inline
```

Importing data from the database (as of 8/15/2015)

In [3]:
```
engineers = pd.read_csv("envision-role-model-engineer.csv")
```

In [4]:
```
ratings = pd.read_csv("envision-role-model-rating.csv")
```

In [17]:
```
engineers.rename(columns={'id': 'engineer_id'}, inplace=True)
```

In [12]:
```
len(ratings)
```

Out[12]:
```
58
```

In total, we have 58 responses.

Merging the two databases. engineers includes their name and version. ratings includes their scores.

In [21]:
```
all_ratings = pd.merge(engineers, ratings, how='inner', on='enginee
r_id')
```

In [24]:
```
all_ratings.rename(columns={'id': 'rating_id'}, inplace=True)
```

In [26]:
```
primed = all_ratings[all_ratings['version'] == 1]
```

In [27]:
```
len(primed)
```

Out[27]:
```
30
```

In [28]:
```
non_primed = all_ratings[all_ratings['version'] == 0]
```

In [29]:
```
len(non_primed)
```

Out[29]:
```
28
```

30 primed respones and 28 non primed

In [31]:
```
primed.columns
```

Out[31]:
```
Index(['engineer_id', 'name', 'version', 'rating_id', 'total_tim
e',
       'QL1_2_inc', 'QL1_2_loa', 'QL1_2_exp', 'QL1_3_inc', 'QL
1_3_loa',
       'QL1_3_exp', 'QL2_3_inc', 'QL2_3_loa', 'QL2_3_exp', 'QL
3_2_inc',
       'QL3_2_loa', 'QL3_2_exp', 'QL3_3_inc', 'QL3_3_loa', 'QL
3_3_exp',
       'NW1_2_inc', 'NW1_2_loa', 'NW1_2_exp', 'NW1_5_inc', 'NW
1_5_loa',
       'NW1_5_exp', 'NW2_1_inc', 'NW2_1_loa', 'NW2_1_exp', 'NW
2_3_inc',
       'NW2_3_loa', 'NW2_3_exp', 'NW3_4_inc', 'NW3_4_loa', 'NW
3_4_exp',
       'CR1_1_inc', 'CR1_1_loa', 'CR1_1_exp', 'CR2_2_inc', 'CR
2_2_loa',
       'CR2_2_exp'],
      dtype='object')
```

T-test comparing Primed and Non Primed results

Adding the 8 credits with or without examples together for t-test.

In [37]:
```
primed['total'] = (primed.QL1_2_loa + primed.QL2_3_loa + primed.QL
3_2_loa + primed.QL3_3_loa + primed.NW1_2_loa + primed.NW2_1_loa +
primed.NW2_3_loa + primed.CR2_2_loa)
```

```
/usr/local/lib/python3.4/site-packages/IPython/kernel/__main__.p
y:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the the caveats in the documentation: http://pandas.pydata.or
g/pandas-docs/stable/indexing.html#indexing-view-versus-copy
  if __name__ == '__main__':
```

In [41]:
```
non_primed['total'] = (non_primed.QL1_2_loa + non_primed.QL2_3_loa
+ non_primed.QL3_2_loa + non_primed.QL3_3_loa + non_primed.NW1_2_lo
a + non_primed.NW2_1_loa + non_primed.NW2_3_loa + non_primed.CR
2_2_loa)
```

```
/usr/local/lib/python3.4/site-packages/IPython/kernel/__main__.p
y:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the the caveats in the documentation: http://pandas.pydata.or
g/pandas-docs/stable/indexing.html#indexing-view-versus-copy
  if __name__ == '__main__':
```

comparing sum, mean and median for the 8 credits with or without examples depending on version. Note the sum is not the best measure because reponses are not equal between groups.

Non Primed

In [116]:
```
sum_mean_median = [non_primed['total'].sum(), non_primed['total'].m
ean().round(2), non_primed['total'].median()]
print("sum:{}, mean:{}, median:{}".format(sum_mean_median[0], sum_m
ean_median[1], sum_mean_median[2]))
```

```
sum:1864, mean:66.57, median:65.5
```

Primed

In [118]:
```
sum_mean_median = [primed['total'].sum(), primed['total'].mean().ro

und(2), primed['total'].median()]
print("sum:{}, mean:{}, median:{}".format(sum_mean_median[0], sum_m
ean_median[1], sum_mean_median[2]))
```

```
sum:2902, mean:96.73, median:103.0
```

An average 30 point difference between groups over just 8 questions.

checking normal distribution

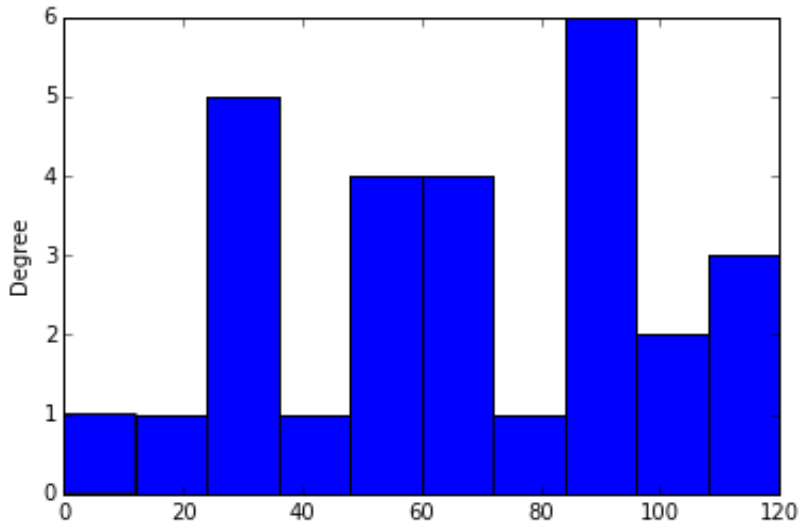
In [66]:
```
from scipy.stats import mstats

mstats.normaltest(non_primed['total'], axis=0)
```

Out[66]:
```
NormaltestResult(statistic=1.4965770314375912, pvalue=0.4731756928
8881023)
```

In [74]:
```
non_primed['total'].plot(kind='hist')
```

Out[74]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x10b84e4e0>
```

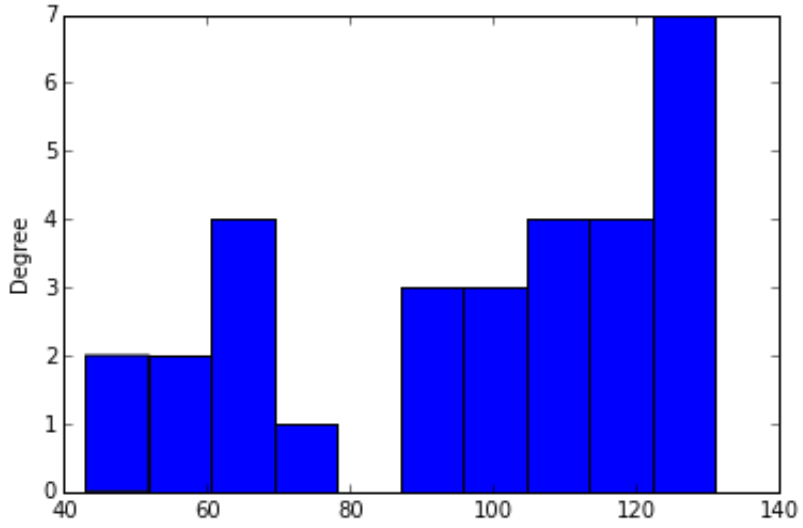


In [67]:
```
mstats.normaltest(primed['total'], axis=0)
```

Out[67]:
```
NormaltestResult(statistic=5.1489840170213395, pvalue=0.0761925181
20086872)
```

In [75]:
```
primed['total'].plot(kind='hist')
```

Out[75]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x10b90f6a0>
```



Results appear normal (enough) to use a t-test. Both datasets have a p value >0.05. Primed results are barely above at 0.07.

T-Test

In [60]:
```
from scipy.stats import ttest_ind

ttest_ind(primed['total'], non_primed['total'])
```

Out[60]:
```
Ttest_indResult(statistic=3.715435736768637, pvalue=0.000469834969
14903417)
```

^Really significant p-value!! Highly unlikely the difference is due to chance.

Below are the results for the no example questions (NW3.4 and CR1.1)

In [77]:
```
non_primed['no_ex_total'] = (non_primed.NW3_4_loa + non_primed.CR
1_1_loa)
```

```
/usr/local/lib/python3.4/site-packages/IPython/kernel/__main__.p
y:1: SettingWithCopyWarning:

A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the the caveats in the documentation: http://pandas.pydata.or
g/pandas-docs/stable/indexing.html#indexing-view-versus-copy
  if __name__ == '__main__':
```

In [78]:
```
primed['no_ex_total'] = (primed.NW3_4_loa + primed.CR1_1_loa)
```

```
/usr/local/lib/python3.4/site-packages/IPython/kernel/__main__.p
y:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the the caveats in the documentation: http://pandas.pydata.or
g/pandas-docs/stable/indexing.html#indexing-view-versus-copy
  if __name__ == '__main__':
```

First comparing sum, mean, and median. Note the sum is not a great measure because the reponses are not equal between groups.

In [114]:
```
sum_mean_median = [non_primed['no_ex_total'].sum(), non_primed['n
o_ex_total'].mean().round(2), non_primed['no_ex_total'].median()]
print("sum:{}, mean:{}, median:{}".format(sum_mean_median[0], sum_m
ean_median[1], sum_mean_median[2]))
```

```
sum:493, mean:17.61, median:13.0
```

In [115]:
```
sum_mean_median = [primed['no_ex_total'].sum(), primed['no_ex_tota
l'].mean().round(2), primed['no_ex_total'].median()]
print("sum:{}, mean:{}, median:{}".format(sum_mean_median[0], sum_m
ean_median[1], sum_mean_median[2]))
```

```
sum:367, mean:12.23, median:9.0
```

For the no example questions, the non-primed group scored better. The primed scores appear to negatively influenced when no longer provided.

Checking Normal distribution (even though Mann Whiteny Test does not require)

Non Primed

```
In [119]: mstats.normaltest(non_primed['no_ex_total'], axis=0)
```

```
Out[119]: NormaltestResult(statistic=4.1839785658015245, pvalue=0.1234413316
          811705)
```

Primed

```
In [120]: mstats.normaltest(primed['no_ex_total'], axis=0)
```

```
Out[120]: NormaltestResult(statistic=5.9557446889156145, pvalue=0.0509010185
          68454949)
```

Doesn't really matter but not normally distributed.

Mann Whitney U Test

This is the two scores added together

```
In [85]: from scipy.stats import mannwhitneyu

         mannwhitneyu(non_primed['no_ex_total'], primed['no_ex_total'])
```

```
Out[85]: MannwhitneyuResult(statistic=314.5, pvalue=0.049865619881486078)
```

95% probability the priming intervention influenced the outcome.

Credit NW3.4 - 1 of 2 questions with no exmaple.

Non Primed

```
In [111]: sum_mean_median = [non_primed['NW3_4_loa'].sum(), non_primed['NW
          3_4_loa'].mean().round(2), non_primed['NW3_4_loa'].median()]
          print("sum:{}, mean:{}, median:{}".format(sum_mean_median[0], sum_m
          ean_median[1], sum_mean_median[2]))
```

```
          sum:249, mean:8.89, median:9.0
```

Primed

```
In [112]: sum_mean_median = [primed['NW3_4_loa'].sum(), primed['NW3_4_loa'].m
          ean().round(2), primed['NW3_4_loa'].median()]
          print("sum:{}, mean:{}, median:{}".format(sum_mean_median[0], sum_m
          ean_median[1], sum_mean_median[2]))

          sum:181, mean:6.03, median:3.0
```

The non primed group scored higher.

Mann Whiteny for NW3.4

```
In [86]: mannwhitneyu(non_primed['NW3_4_loa'], primed['NW3_4_loa'])

Out[86]: MannwhitneyuResult(statistic=313.0, pvalue=0.043709644558397064)
```

'>95% chance the intervention influenced the outcome.

Credit CR1.1 - 1 of 2 questions with no exmaple

Non Primed

```
In [125]: sum_mean_median = [non_primed['CR1_1_loa'].sum(), non_primed['CR
          1_1_loa'].mean().round(2), non_primed['CR1_1_loa'].median()]
          print("sum:{}, mean:{}, median:{}".format(sum_mean_median[0], sum_m
          ean_median[1], sum_mean_median[2]))

          sum:244, mean:8.71, median:4.0
```

Primed

```
In [126]: sum_mean_median = [primed['CR1_1_loa'].sum(), primed['CR1_1_loa'].m
          ean().round(2), primed['CR1_1_loa'].median()]
          print("sum:{}, mean:{}, median:{}".format(sum_mean_median[0], sum_m
          ean_median[1], sum_mean_median[2]))

          sum:186, mean:6.2, median:0.0
```

The non primed group scored higher than the primed.

Mann Whiteny for CR1.1

In [87]:
```
mannwhitneyu(non_primed['CR1_1_loa'], primed['CR1_1_loa'])
```

Out[87]:
```
MannwhitneyuResult(statistic=320.0, pvalue=0.05214693114403364)
```

95% probability the intervention effected the outcome.

Negative Example Credits

Total score of the two credits

In [91]:
```
primed['neg_ex_total'] = (primed.QL1_3_loa + primed.NW1_5_loa)
```

```
/usr/local/lib/python3.4/site-packages/IPython/kernel/__main__.p
y:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the the caveats in the documentation: http://pandas.pydata.or
g/pandas-docs/stable/indexing.html#indexing-view-versus-copy
  if __name__ == '__main__':
```

In [92]:
```
non_primed['neg_ex_total'] = (non_primed.QL1_3_loa + non_primed.NW
1_5_loa)
```

```
/usr/local/lib/python3.4/site-packages/IPython/kernel/__main__.p
y:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the the caveats in the documentation: http://pandas.pydata.or
g/pandas-docs/stable/indexing.html#indexing-view-versus-copy
  if __name__ == '__main__':
```

Sum, median, mode

Non Primed

In [130]:
```
sum_mean_median = [non_primed['neg_ex_total'].sum(), non_primed['neg_ex_total'].mean().round(2), non_primed['neg_ex_total'].median()]
print("sum:{}, mean:{}, median:{}".format(sum_mean_median[0], sum_mean_median[1], sum_mean_median[2]))
```

```
sum:275, mean:9.82, median:7.0
```

Primed

In [131]:
```
sum_mean_median = [primed['neg_ex_total'].sum(), primed['neg_ex_total'].mean().round(2), primed['neg_ex_total'].median()]
print("sum:{}, mean:{}, median:{}".format(sum_mean_median[0], sum_mean_median[1], sum_mean_median[2]))
```

```
sum:206, mean:6.87, median:5.0
```

The non primed group scored higher.

Mann Whitney for negatvie examples added together

In [93]:
```
mannwhitneyu(non_primed['neg_ex_total'], primed['neg_ex_total'])
```

Out[93]:
```
MannwhitneyuResult(statistic=330.0, pvalue=0.080455247559231224)
```

Negative Example Credit QL1.3

In [133]:
```
sum_mean_median = [non_primed['QL1_3_loa'].sum(), non_primed['QL1_3_loa'].mean().round(2), non_primed['QL1_3_loa'].median()]
print("sum:{}, mean:{}, median:{}".format(sum_mean_median[0], sum_mean_median[1], sum_mean_median[2]))
```

```
sum:144, mean:5.14, median:2.0
```

In [136]:
```
sum_mean_median = [primed['QL1_3_loa'].sum(), primed['QL1_3_loa'].mean().round(2), primed['QL1_3_loa'].median()]
print("sum:{}, mean:{}, median:{}".format(sum_mean_median[0], sum_mean_median[1], sum_mean_median[2]))
```

```
sum:89, mean:2.97, median:1.0
```

The non primed group scored more points on average.

Mann Whitney for negative example QL1.3

In [94]:
```
# mann whitney u test for QL1.3
mannwhitneyu(non_primed['QL1_3_loa'], primed['QL1_3_loa'])
```

Out[94]:
```
MannwhitneyuResult(statistic=290.0, pvalue=0.019684967670828155)
```

99% probability the intervention influenced the outcome.

Negative Example Credit NW1.5

Non Primed

In [138]:
```
sum_mean_median = [non_primed['NW1_5_loa'].sum(), non_primed['NW
1_5_loa'].mean().round(2), non_primed['NW1_5_loa'].median()]
print("sum:{}, mean:{}, median:{}".format(sum_mean_median[0], sum_m
ean_median[1], sum_mean_median[2]))
```

```
sum:131, mean:4.68, median:2.0
```

Primed

In [137]:
```
sum_mean_median = [primed['NW1_5_loa'].sum(), primed['NW1_5_loa'].m
ean().round(2), primed['NW1_5_loa'].median()]
print("sum:{}, mean:{}, median:{}".format(sum_mean_median[0], sum_m
ean_median[1], sum_mean_median[2]))
```

```
sum:117, mean:3.9, median:1.0
```

Non primed group on average scored more points than the primed.

Mann whitney for negative example NW1.5

In [95]:
```
mannwhitneyu(non_primed['NW1_5_loa'], primed['NW1_5_loa'])
```

Out[95]:
```
MannwhitneyuResult(statistic=389.0, pvalue=0.30690139915943737)
```

Not signifcant.