In [29]:
```
import numpy as np
import matplotlib.pyplot as plt
import scipy.stats as sp
import pandas as pd
```

In [30]:
```
data = pd.read_csv('AMZN.csv')
data.head(5)
```

Out[30]:

	Date	Open	High	Low	Close	Adj Close	Volume
0	2016-03-24	567.109985	583.549988	567.080017	582.950012	582.950012	5185500
1	2016-03-28	584.400024	584.750000	575.559998	579.869995	579.869995	3121500
2	2016-03-29	580.150024	595.849976	576.500000	593.859985	593.859985	4392600
3	2016-03-30	596.710022	603.239990	595.000000	598.690002	598.690002	3890500
4	2016-03-31	599.280029	600.750000	592.210022	593.640015	593.640015	2681800

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

Out[31]:
```
Index(['Date', 'Open', 'High', 'Low', 'Close', 'Adj Close', 'Volume'
], dtype='object')
```

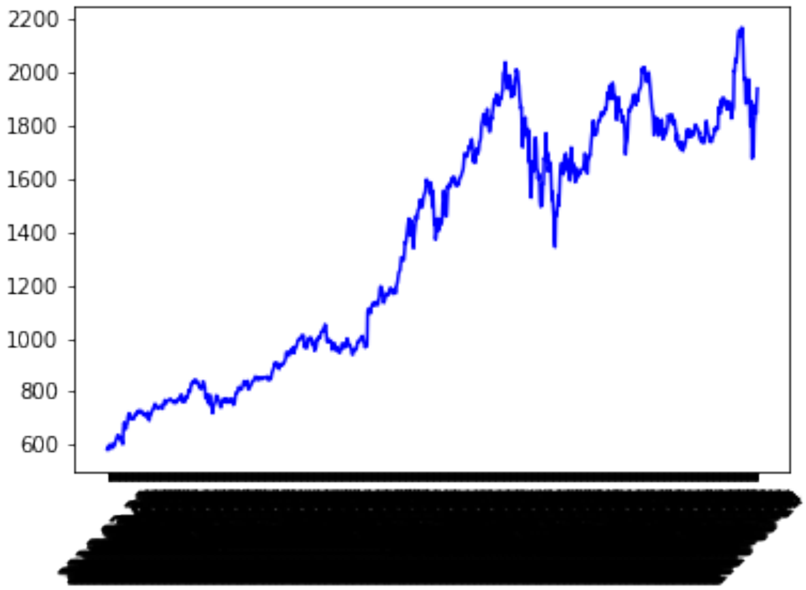
In [32]:
```
price = data['Adj Close']
price.head(5)
```

Out[32]:
```
0    582.950012
1    579.869995
2    593.859985
3    598.690002
4    593.640015
Name: Adj Close, dtype: float64
```

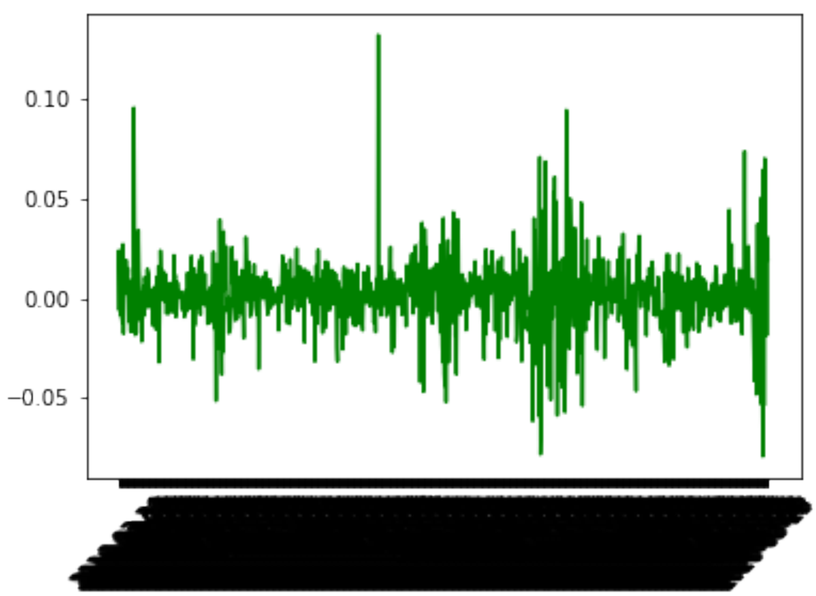
In [33]:
```
# convert dd/mm/yy text format to date format that can be plotted
date_ymd = data['Date'] # in dd/mm/yy format
print(date_ymd[1])
from datetime import datetime
```

```
2016-03-28
```

In [34]:
```
plt.plot_date(date_ymd,price,'b-')
plt.xticks(rotation=45)
plt.show()
```



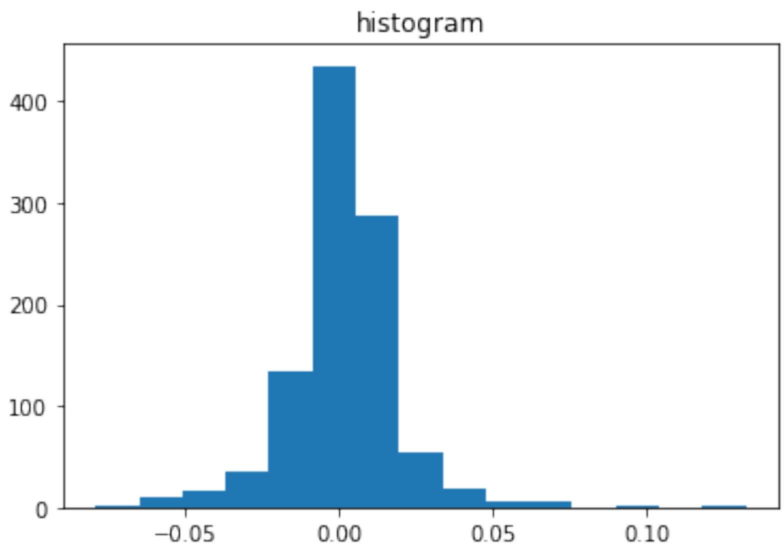
In [35]:
```
N = len(price)
r = np.zeros(N-1) # create array of zeros of size N-1
for j in range(N-1):
    r[j] = (price[j+1]-price[j])/price[j]
    #print(j,price[j],price[j+1],r[j])
plt.plot_date(date_ymd[:-1],r,'g-')
plt.xticks(rotation=45)
plt.show()
```



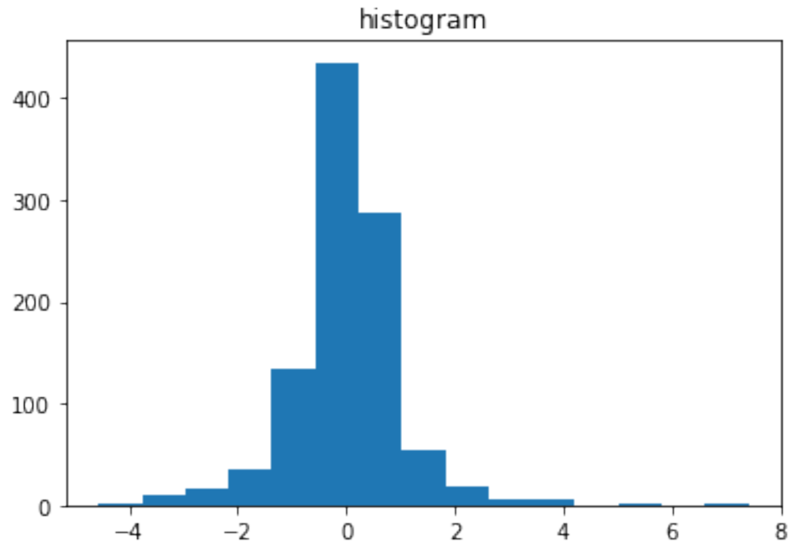
In [36]:
```
# basic statistics
mu = np.mean(r)  # mean of sample
sigma2 = np.var(r,ddof=1) # sample variance
sigma = np.sqrt(sigma2)
print('mu    = ',mu)
print('sigma = ',sigma)
```

```
mu    =  0.001351028155043686
sigma =  0.01766481077936605
```

In [37]:
```
# plot a basic histogram
plt.hist(r, bins = 15)
plt.title("histogram")
plt.show()
```



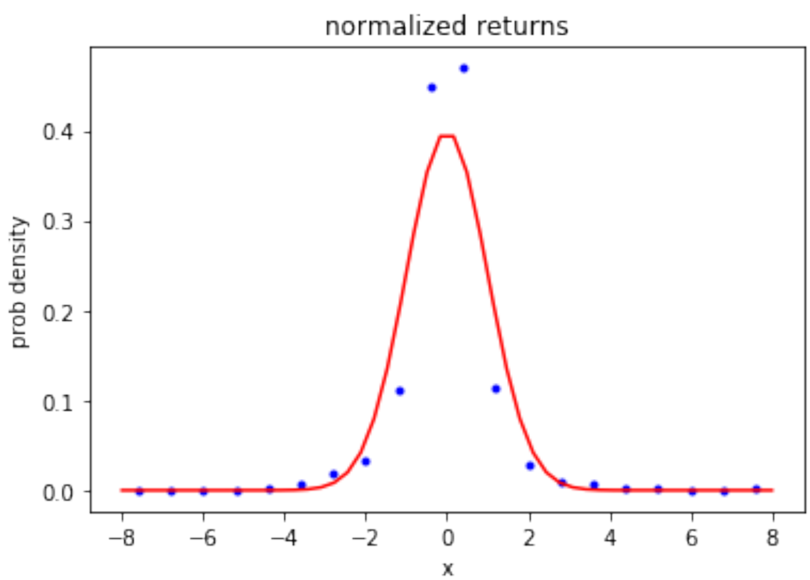
In [38]:
```
# normalize data relative to its mean and standard deviation
z = (r-mu)/sigma
plt.hist(z, bins = 15)
plt.title("histogram")
plt.show()
```



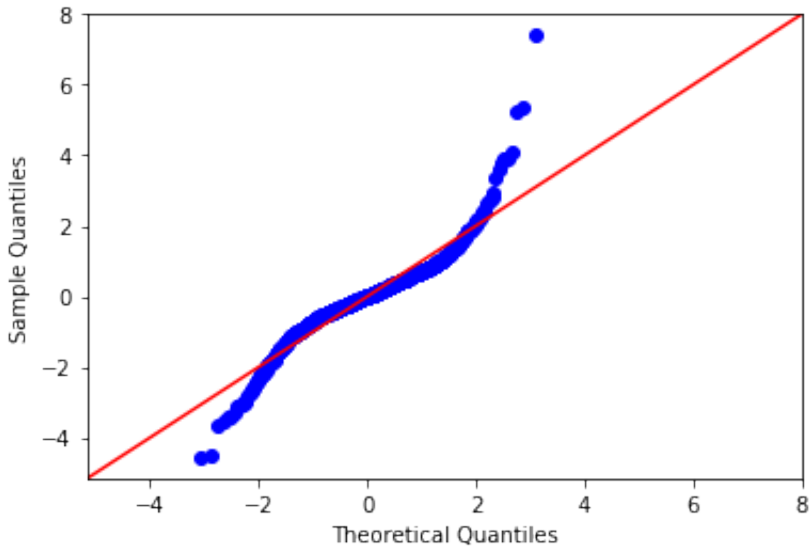
In [39]:
```
### kernel density estimate
nbins = 20  # number of bins
b = 8       # right end (from looking at histogram)
a = -8      # left end (from looking at histogram)
dx = (b-a)/nbins  # bin width
bin_edges = np.linspace(a,b,nbins+1) # nbins+1 points for nbins bins
# counts in each bin
counts, bin_edges = np.histogram(z,bins=bin_edges)
M = sum(counts)
print('M = ',M)
# find bin centers and kernel density
bin_centers = (bin_edges[1:] + bin_edges[:-1])/2
kernel_density = counts/(M*dx)

# plot of kernel density estimate and exact prob density
plt.plot(bin_centers,kernel_density,'b.')
xfine = np.linspace(a,b) # fine array in x for plotting exact curve
plt.plot(xfine,sp.norm.pdf(xfine,0,1),'r-')
plt.title('normalized returns')
plt.xlabel('x')
plt.ylabel('prob density')
plt.show()
```

```
M =  1006
```



In [40]:
```
# quantile-quantile plot (qqplot)
import statsmodels.api as sm # statsmodels needed for qq plot
sm.qqplot(z, dist=sp.norm, loc=0, scale=1, line = '45') # <- CORRECTED
plt.show()
```



The daily Amazon returns cannot be described by a normal distribution.

Because the greatest returns are above the theoretical values and the lowest returns (greatest loss) are below the theoretical values.

So distribution of data has "fat tails" relative to the normal distribution.

In [ ]: