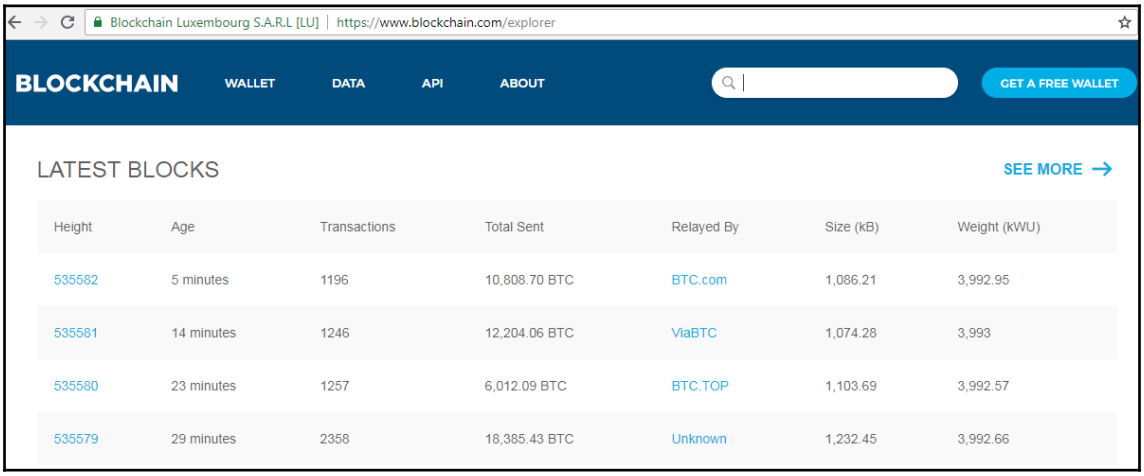
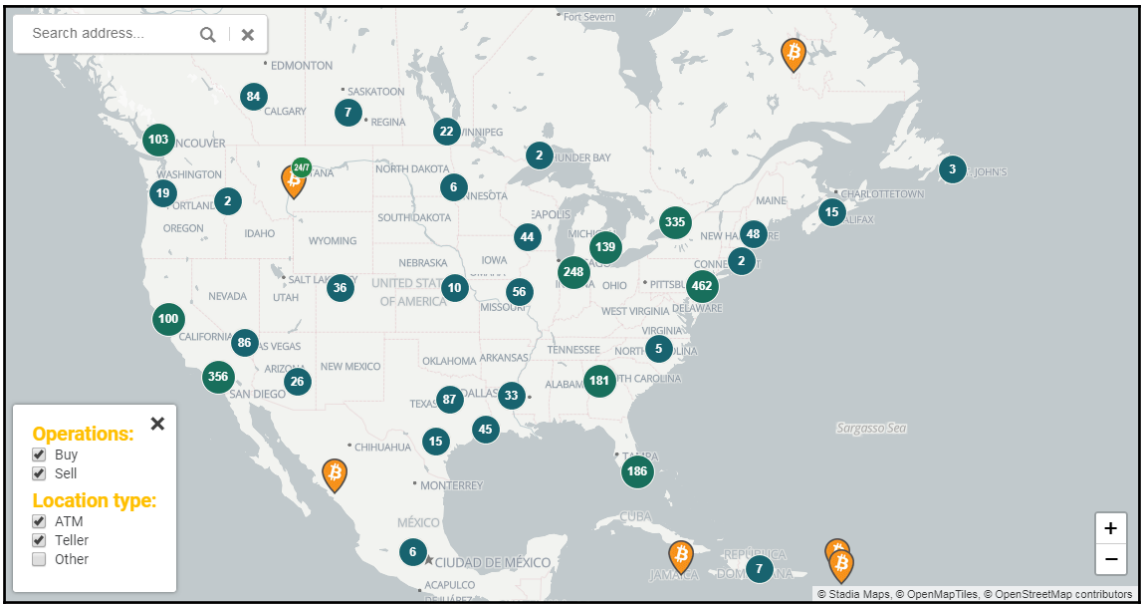
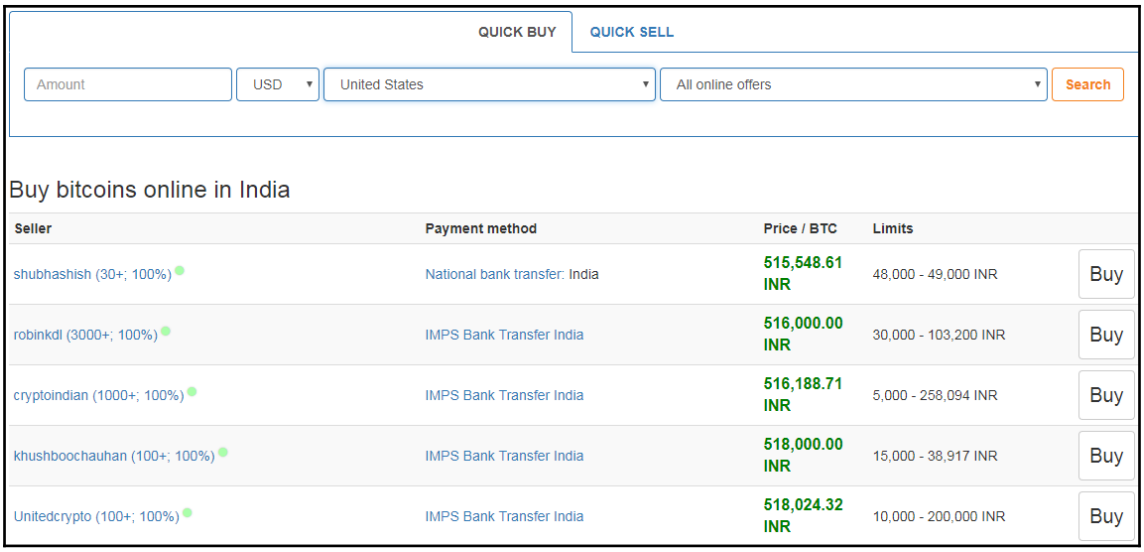
Chapter 1: Getting Started with Bitcoin

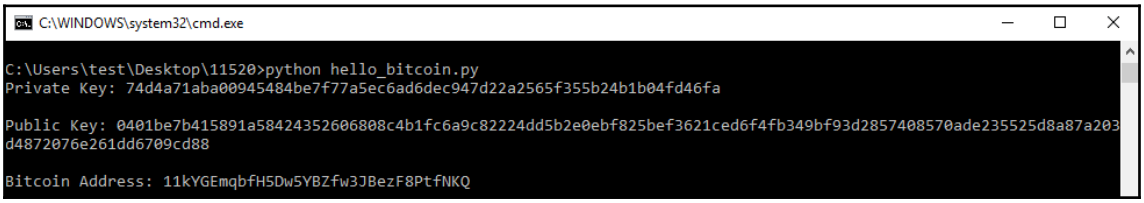


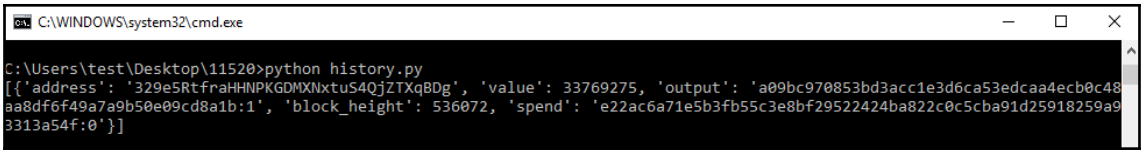
Height	Age	Transactions	Total Sent	Relayed By	Size (kB)	Weight (kWU)
535582	5 minutes	1196	10,808.70 BTC	BTC.com	1,086.21	3,992.95
535581	14 minutes	1246	12,204.06 BTC	ViaBTC	1,074.28	3,993
535580	23 minutes	1257	6,012.09 BTC	BTC.TOP	1,103.69	3,992.57
535579	29 minutes	2358	18,385.43 BTC	Unknown	1,232.45	3,992.66




QUICK BUY
QUICK SELL
Amount
USD
United States
All online offers
Search
Buy bitcoins online in India
Seller
Payment method
Price / BTC
Limits
shubhashish (30+; 100%)
National bank transfer: India
515,548.61 INR
48,000 - 49,000 INR
Buy
robinkdl (3000+; 100%)
IMPS Bank Transfer India
516,000.00 INR
30,000 - 103,200 INR
Buy
cryptoindian (1000+; 100%)
IMPS Bank Transfer India
516,188.71 INR
5,000 - 258,094 INR
Buy
khushboochauhan (100+; 100%)
IMPS Bank Transfer India
518,000.00 INR
15,000 - 38,917 INR
Buy
Unitedcrypto (100+; 100%)
IMPS Bank Transfer India
518,024.32 INR
10,000 - 200,000 INR
Buy

Chapter 2: Programming Bitcoin and Blockchain with Python





Block #536081

Summary	
Number Of Transactions	1084
Output Total	16,746.14329508 BTC
Estimated Transaction Volume	8,615.88047589 BTC
Transaction Fees	0.06291561 BTC
Height	536081 (Main Chain)
Timestamp	2018-08-10 11:58:59
Received Time	2018-08-10 11:58:59
Relayed By	AntPool
Difficulty	5,949,437,371,609.53
Bits	388976507
Size	1141.189 kB
Weight	3992.935 kWU
Version	0x20000000
Nonce	3627638385
Block Reward	12.5 BTC

Hashes	
Hash	00000000000000000002e90b284607359f3415647626447643b9b880ee00e41fa
Previous Block	00000000000000000023e20e03d770d8e82656231ebd8a1c85096cd79b600f21
Next Block(s)	00000000000000000026d92de3cfe93dedb82328a408189638f5635fc0465bd0
Merkle Root	1cb965f1aaf6fb30403bdc2438f4b7958e039ccd7da0e09665b03066a5f27df2

Bitcoin Stats

Summary of bitcoin statistics for the previous 24 hour period.

BLOCK SUMMARY

Blocks Mined	149
Time Between Blocks	9.21 minutes
Bitcoins Mined	1,862.50000000 BTC

MARKET SUMMARY

Market Price	$6,396.51	View Chart
Trade Volume	$482,316,162.23	
Trade Volume	75,891.60000000 BTC	

TRANSACTION SUMMARY

Total Transaction Fees (BTC)	21.77783945 BTC	View Chart
Number of Transactions	221,755	View Chart
Total Output Volume (BTC)	1,080,665.47279284 BTC	View Chart
Estimated Transaction Volume (BTC)	147,471.53013832 BTC	View Chart

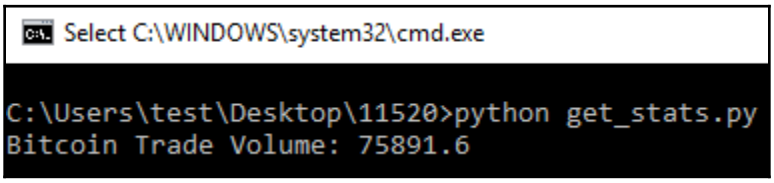
```
Select C:\WINDOWS\system32\cmd.exe

C:\Users\test\Desktop\11520>pip install blockchain
Collecting blockchain
  Downloading https://files.pythonhosted.org/packages/0b/85/ca826affaeaad16506d3b19525f18cf18de394d52d4a9e645ef7dc9d59a2
/blockchain-1.4.4.tar.gz
Collecting enum-compat (from blockchain)
  Downloading https://files.pythonhosted.org/packages/95/6e/26bdcba28b66126f66cf3e4cd03bcd63f7ae330d29ee68b1f6b623550bfa
/enum-compat-0.0.2.tar.gz
Collecting future (from blockchain)
  Downloading https://files.pythonhosted.org/packages/00/2b/8d082ddfed935f3608cc61140df6dcbf0edea1bc3ab52fb6c29ae3e81e85
/future-0.16.0.tar.gz (824kB)
    100% |                                | 829kB 196kB/s
Building wheels for collected packages: blockchain, enum-compat, future
  Running setup.py bdist_wheel for blockchain ... done
  Stored in directory: C:\Users\test\AppData\Local\pip\Cache\wheels\6e\d9\4f\65e76bdafa4f3a950c679c9d999664c604b33077b0a
9d014e8
  Running setup.py bdist_wheel for enum-compat ... done
  Stored in directory: C:\Users\test\AppData\Local\pip\Cache\wheels\b1\69\f4\229af6a49beece0f688c9c73d9188769b89e698361d
21ce96a
  Running setup.py bdist_wheel for future ... done
  Stored in directory: C:\Users\test\AppData\Local\pip\Cache\wheels\bf\c9\a3\c538d90ef17cf7823fa51fc701a7a7a910a80f6a405
bf15b1a
Successfully built blockchain enum-compat future
Installing collected packages: enum-compat, future, blockchain
Successfully installed blockchain-1.4.4 enum-compat-0.0.2 future-0.16.0
```

```
Select C:\WINDOWS\system32\cmd.exe

C:\Users\test\Desktop\11520>python get_exchange_rates.py
Bitcoin Prices in various currencies:
USD 6473.09
AUD 8854.02
BRL 24606.34
CAD 8478.07
CHF 6436.94
CLP 4188478.45
CNY 44362.38
DKK 42119.5
EUR 5654.4
GBP 5069.95
HKD 50814.09
INR 445576.75
ISK 703900.92
JPY 717353.48
KRW 7300999.67
NZD 9809.48
PLN 24232.78
RUB 433159.85
SEK 58869.77
SGD 8872.73
THB 215320.91
TWD 198775.11
```

```
100 euros in Bitcoin: 0.01769439
```

Stats

```
trade_volume_btc : float
miners_revenue_usd : float
btc_mined : long
trade_volume_usd : float
difficulty : float
minutes_between_blocks : float
number_of_transactions : int
hash_rate : float
timestamp : long
mined_blocks : int
blocks_size : int
total_fees_btc : int
total_btc_sent : long
estimated_btc_sent : long
total_btc : long
total_blocks :int
next_retarget : int
estimated_transaction_volume_usd : float
miners_revenue_btc : int
market_price_usd : float
```



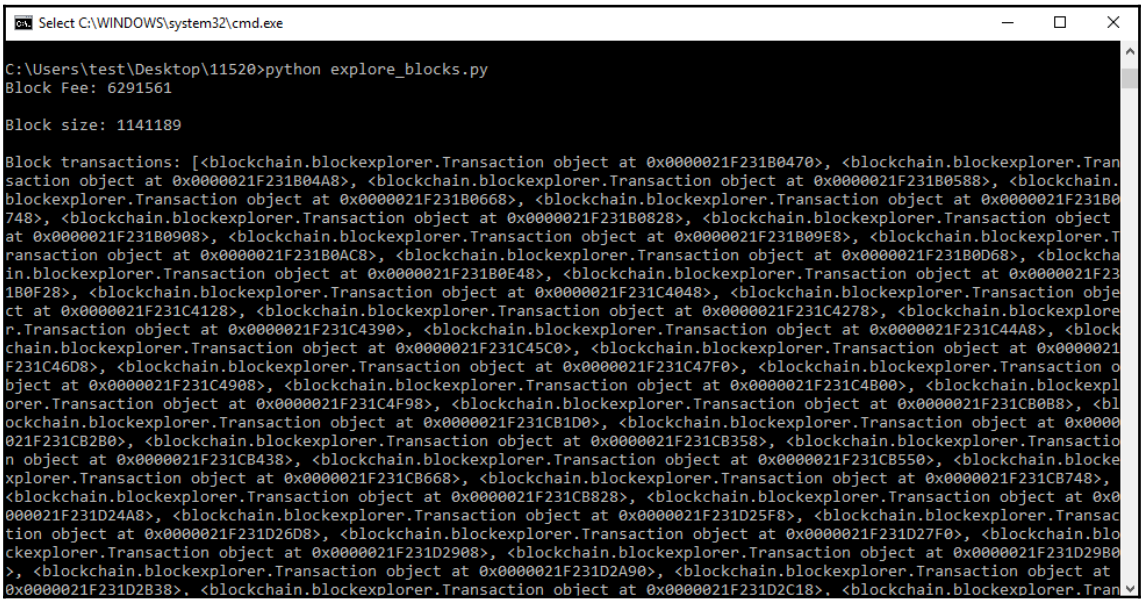

```
Bitcoin mined: 191250000000
```

```
Bitcoin market price: 6355.33
```

Block #536081

Summary	
Number Of Transactions	1084
Output Total	16,746.14329508 BTC
Estimated Transaction Volume	8,615.88047589 BTC
Transaction Fees	0.06291561 BTC

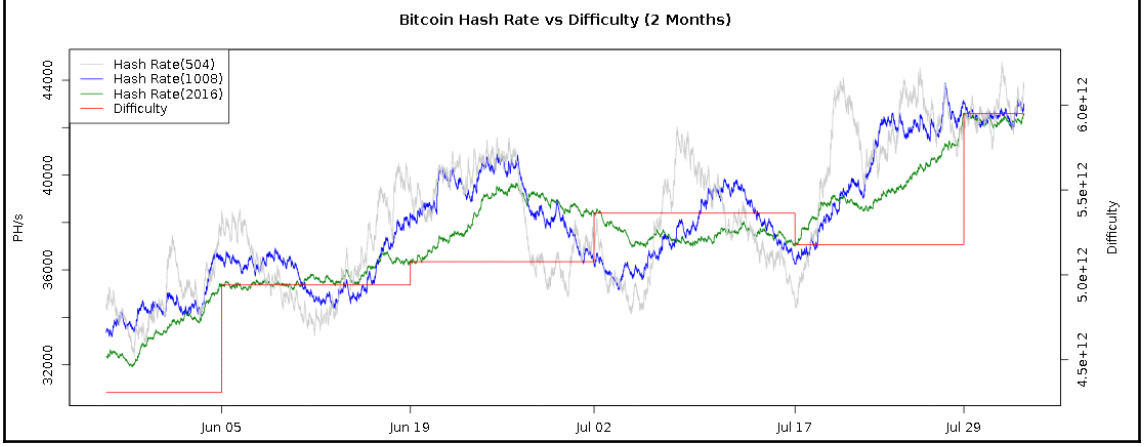
Hashes	
Hash	00000000000000000002e90b284607359f3415647626447643b9b880ee00e41fa
Previous Block	00000000000000000023e20e03d770d8e82656231ebd8a1c85096cd79b600f21
Next Block(s)	00000000000000000026d92de3cfe93dedb82328a408189638f5635fc0465bd0
Merkle Root	1cb965f1aaf6fb30403bdc2438f4b7958e039ccd7da0e09665b03066a5f27df2



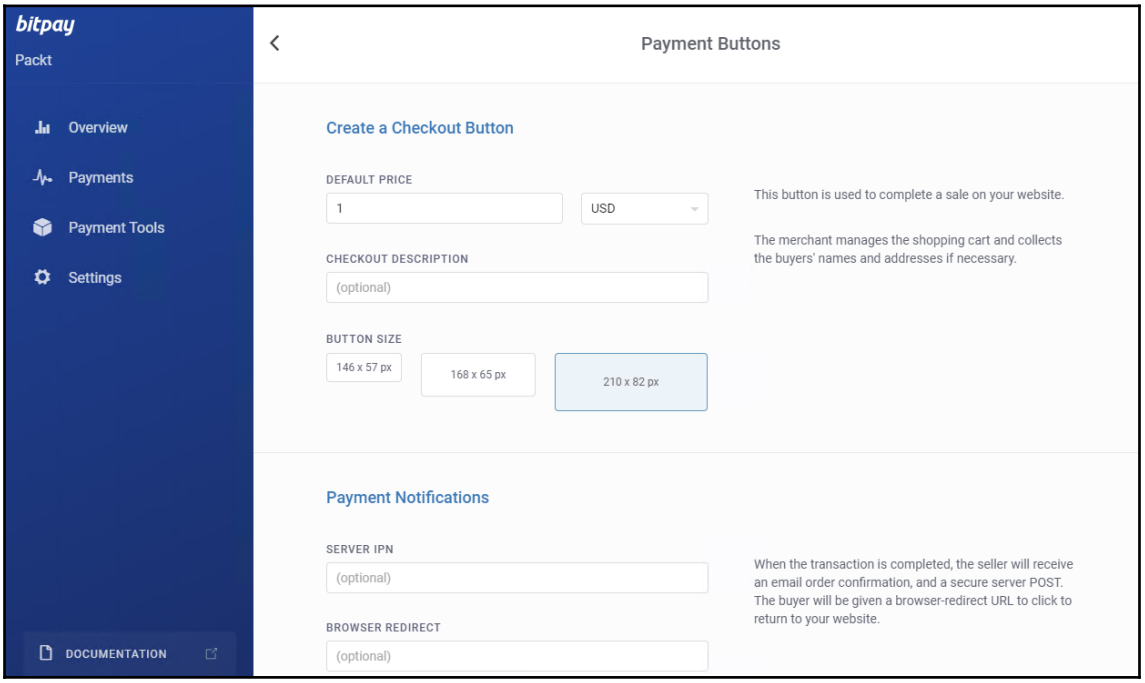
Bitcoin Difficulty:	5,949,437,371,609
Estimated Next Difficulty:	6,172,153,510,638 (+3.74%)
Adjust time:	After 1391 Blocks, About 9.6 days
Hashrate(?):	43,669,949,157 GH/s
Block Generation Time(?):	1 block: 9.9 minutes 3 blocks: 29.8 minutes 6 blocks: 59.5 minutes
Updated:	19:25 (5.0 minutes ago)

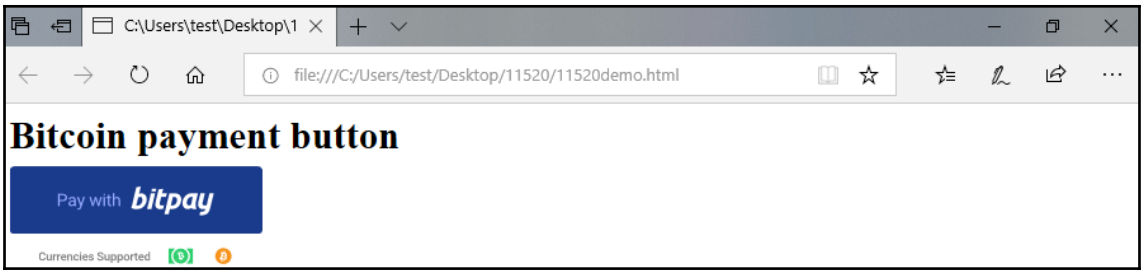
Difficulty: 5949437371609 BTC/USD: 6493.6

1000000	KH/s	1.761e-9	BTC/hour	0.00001144	USD/hour
1000	MH/s	4.227e-8	BTC/day	0.0002745	USD/day
1	GH/s	2.959e-7	BTC/week	0.001921	USD/week
0.001	TH/s	0.000001268	BTC/month	0.008234	USD/month



Chapter 3: Earning Bitcoin Programmatically

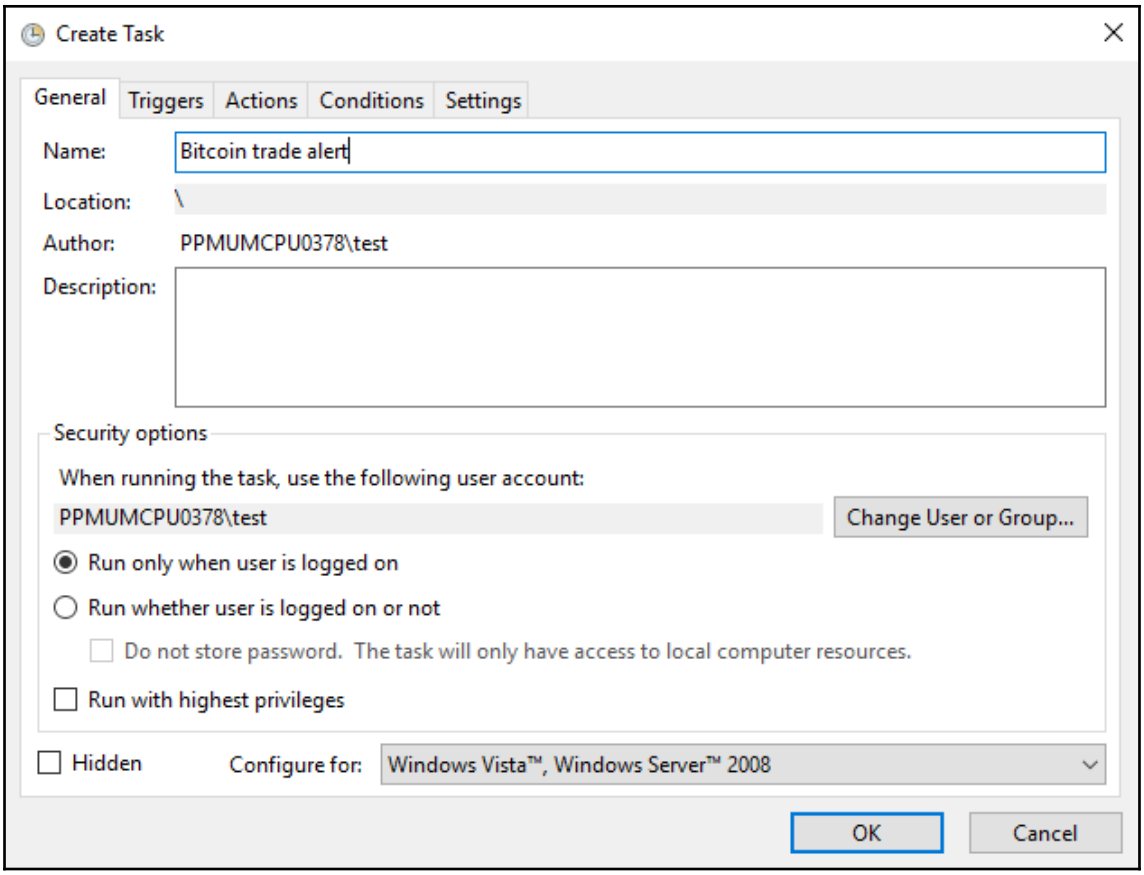


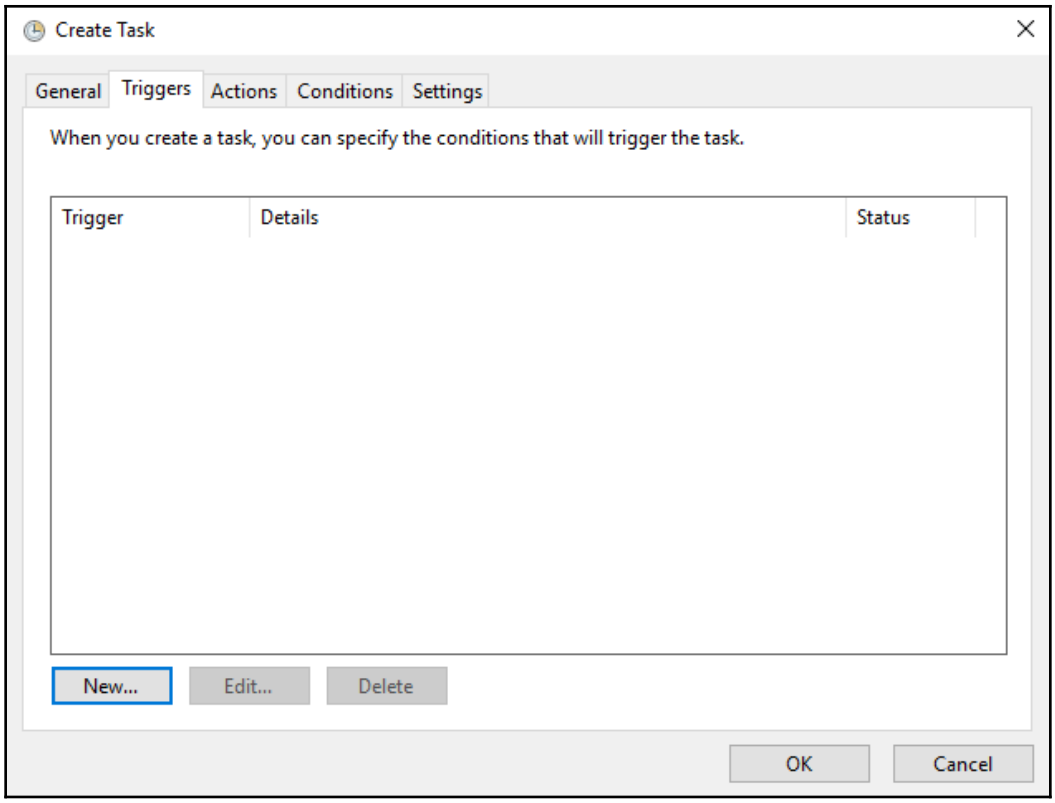


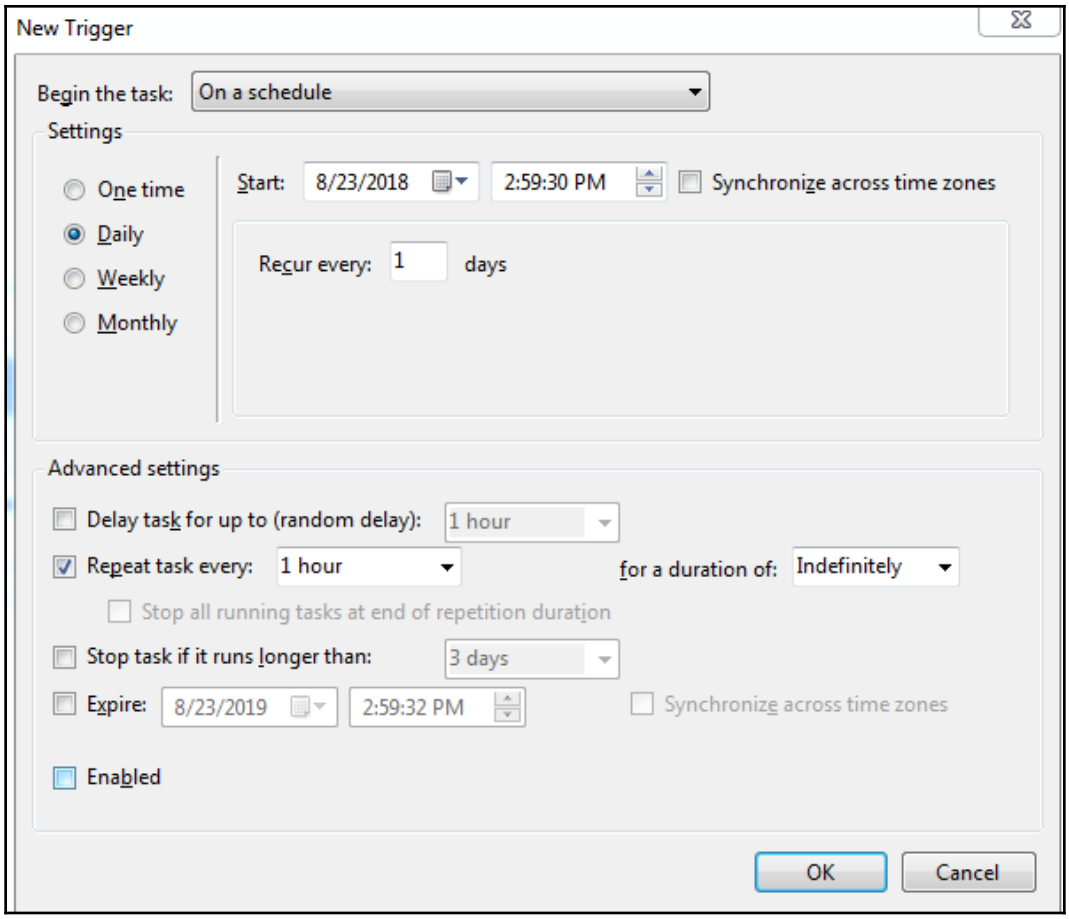

```
C:\Users\test\Desktop\11520>python bitcoin_trade.py

C:\Users\test\Desktop\11520>_
```

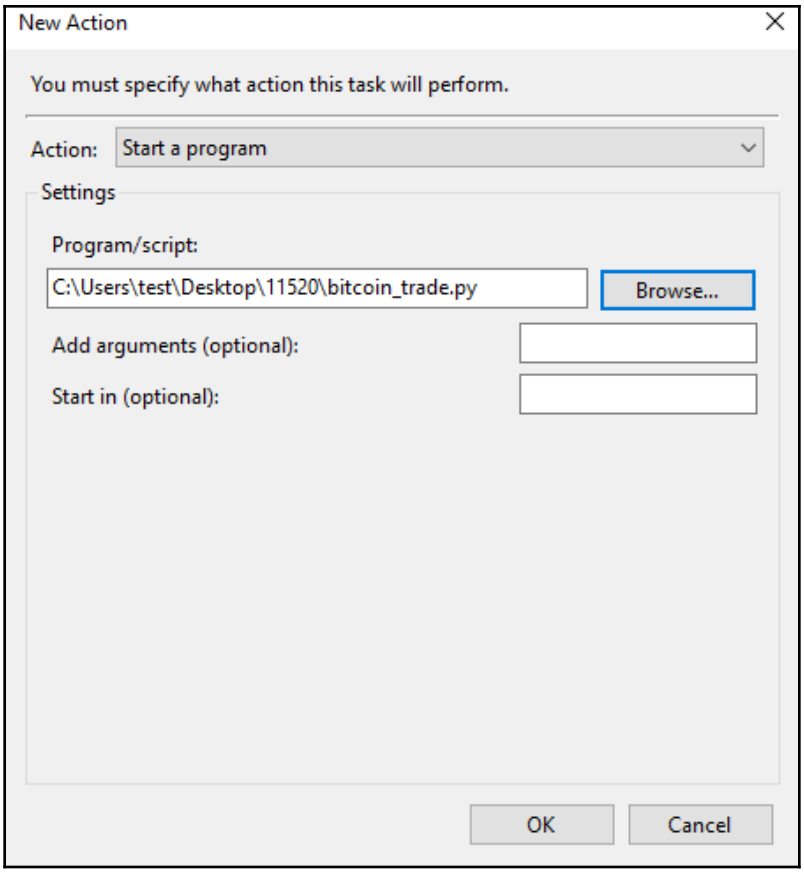
Bitcoin sell Price is 6420.4 which is higher then threshhold price of 6400.
Good time to sell!

Bitcoin Buy Price is 6416.5 which is lower then threshhold price of 6500.
Good time to buy!

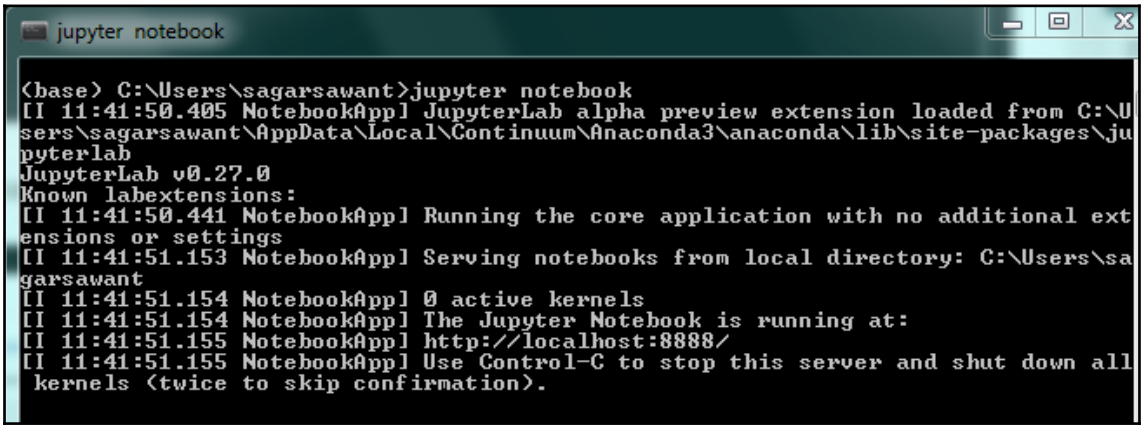

Create Task
General
Triggers
Actions
Conditions
Settings
Name:
Bitcoin trade alert
Location:
\
Author:
PPMUMCPU0378\test
Description:
Security options
When running the task, use the following user account:
PPMUMCPU0378\test
Change User or Group...
Run only when user is logged on
Run whether user is logged on or not
Do not store password. The task will only have access to local computer resources.
Run with highest privileges
Hidden
Configure for:
Windows Vista™, Windows Server™ 2008
OK
Cancel

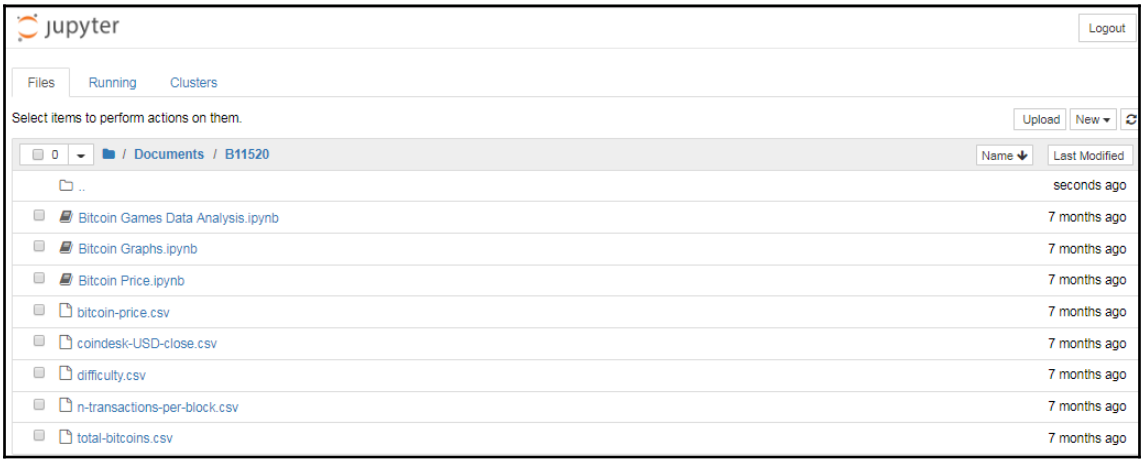

Create Task
General
Triggers
Actions
Conditions
Settings
When you create a task, you can specify the conditions that will trigger the task.
Trigger
Details
Status
New...
Edit...
Delete
OK
Cancel

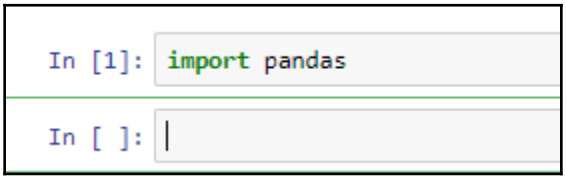

New Trigger
Begin the task:
On a schedule
Settings
One time
Daily
Weekly
Monthly
Start:
8/23/2018
2:59:30 PM
Synchronize across time zones
Recur every:
1
days
Advanced settings
Delay task for up to (random delay):
1 hour
Repeat task every:
1 hour
for a duration of:
Indefinitely
Stop all running tasks at end of repetition duration
Stop task if it runs longer than:
3 days
Expire:
8/23/2019
2:59:32 PM
Synchronize across time zones
Enabled
OK
Cancel


New Action
You must specify what action this task will perform.
Action:
Start a program
Settings
Program/script:
C:\Users\test\Desktop\11520\bitcoin_trade.py
Browse...
Add arguments (optional):
Start in (optional):
OK
Cancel

Chapter 4: Bitcoin Data Analysis



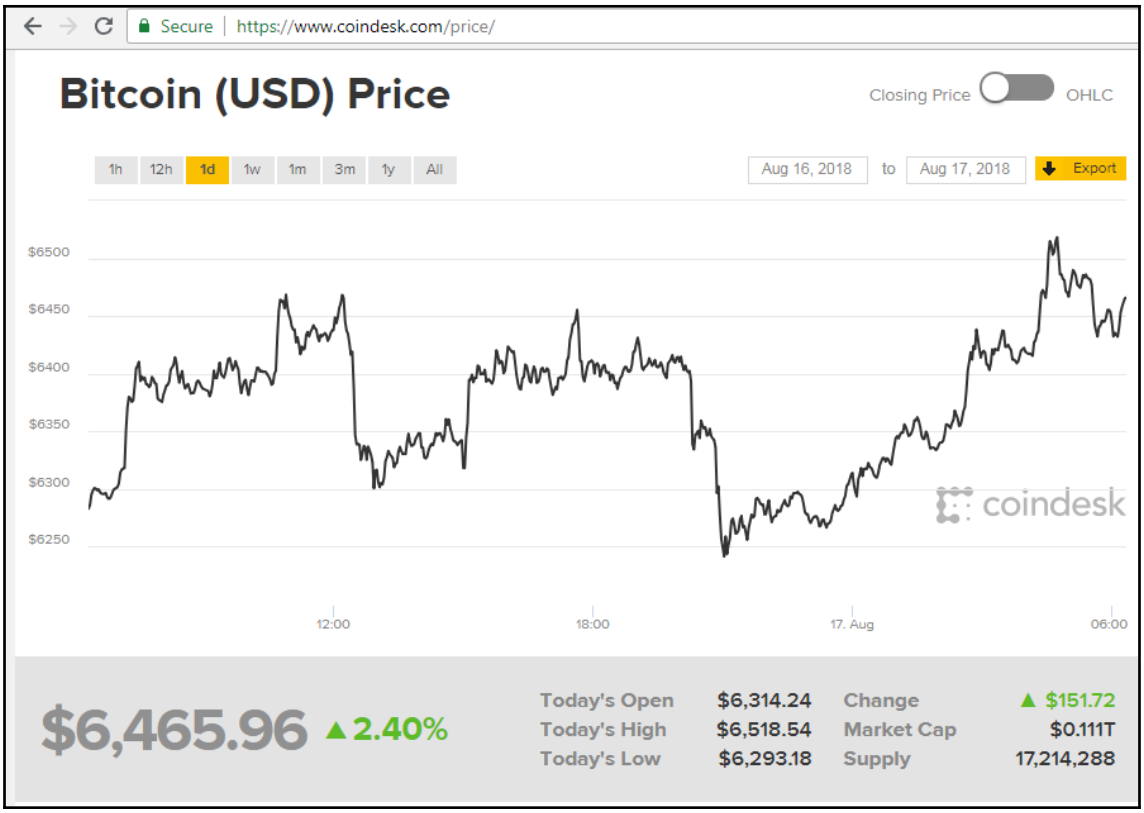




```
In [1]: import pandas

In [2]: import matplotlib.pyplot as plt

In [ ]:
```

```
In [5]: import pandas as pd

In [6]: import matplotlib.pyplot as plt

In [7]: pd.options.mode.chained_assignment = None

In [ ]:
```

```
In [5]: import pandas as pd

In [6]: import matplotlib.pyplot as plt

In [7]: pd.options.mode.chained_assignment = None

In [8]: %matplotlib inline

In [ ]:
```




```
In [5]: import pandas as pd

In [6]: import matplotlib.pyplot as plt

In [7]: pd.options.mode.chained_assignment = None

In [8]: %matplotlib inline

In [9]: price = pd.read_csv("D:/bitcoin-price.csv")

In [ ]:
```

```
In [5]: import pandas as pd

In [6]: import matplotlib.pyplot as plt

In [7]: pd.options.mode.chained_assignment = None

In [8]: %matplotlib inline

In [9]: price = pd.read_csv("D:/bitcoin-price.csv")

In [10]: price.head()
```

Out[10]:

	Date	Close Price
0	2010-07-18 00:00:00	0.09
1	2010-07-19 00:00:00	0.08
2	2010-07-20 00:00:00	0.07
3	2010-07-21 00:00:00	0.08
4	2010-07-22 00:00:00	0.05

```
In [ ]:
```

```
In [5]: import pandas as pd

In [6]: import matplotlib.pyplot as plt

In [7]: pd.options.mode.chained_assignment = None

In [8]: %matplotlib inline

In [9]: price = pd.read_csv("D:/bitcoin-price.csv")

In [10]: price.head()
```

Out[10]:

	Date	Close Price
0	2010-07-18 00:00:00	0.09
1	2010-07-19 00:00:00	0.08
2	2010-07-20 00:00:00	0.07
3	2010-07-21 00:00:00	0.08
4	2010-07-22 00:00:00	0.05

```
In [11]: price.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2592 entries, 0 to 2591
Data columns (total 2 columns):
Date            2592 non-null object
Close Price     2590 non-null float64
dtypes: float64(1), object(1)
memory usage: 40.6+ KB
```

```
In [12]: price.tail()
```

Out[12]:

	Date	Close Price
2587	2017-08-17 00:00:00	4316.34
2588	2017-08-18 00:00:00	4159.46
2589	2017-08-19 15:54:00	4062.17
2590	This data was produced from the CoinDesk price...	NaN
2591	http://www.coindesk.com/price/	NaN

```
In [12]: price.tail()
```

Out[12]:

	Date	Close Price
2587	2017-08-17 00:00:00	4316.34
2588	2017-08-18 00:00:00	4159.46
2589	2017-08-19 15:54:00	4062.17
2590	This data was produced from the CoinDesk price...	NaN
2591	http://www.coindesk.com/price/	NaN

```
In [13]: price = price.dropna()
```

```
In [14]: price.tail()
```

Out[14]:

	Date	Close Price
2585	2017-08-15 00:00:00	4204.43
2586	2017-08-16 00:00:00	4425.30
2587	2017-08-17 00:00:00	4316.34
2588	2017-08-18 00:00:00	4159.46
2589	2017-08-19 15:54:00	4062.17

```
In [15]: price['Date'] = pd.to_datetime(price['Date'], format = "%Y-%m-%d")

In [ ]:
```

```
In [15]: price['Date'] = pd.to_datetime(price['Date'], format = "%Y-%m-%d")

In [16]: price.info()

<class 'pandas.core.frame.DataFrame'>
Int64Index: 2590 entries, 0 to 2589
Data columns (total 2 columns):
Date           2590 non-null datetime64[ns]
Close Price    2590 non-null float64
dtypes: datetime64[ns](1), float64(1)
memory usage: 60.7 KB
```

```
In [14]: price.tail()
```

Out[14]:

	Date	Close Price
2585	2017-08-15 00:00:00	4204.43
2586	2017-08-16 00:00:00	4425.30
2587	2017-08-17 00:00:00	4316.34
2588	2017-08-18 00:00:00	4159.46
2589	2017-08-19 15:54:00	4062.17

```
In [15]: price['Date'] = pd.to_datetime(price['Date'], format = "%Y-%m-%d")

In [16]: price.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2590 entries, 0 to 2589
Data columns (total 2 columns):
Date           2590 non-null datetime64[ns]
Close Price    2590 non-null float64
dtypes: datetime64[ns](1), float64(1)
memory usage: 60.7 KB
```

```
In [17]: price.index = price["Date"]
```

```
In [17]: price.index = price["Date"]

In [18]: del price["Date"]
```

```
In [17]: price.index = price["Date"]

In [18]: del price["Date"]

In [19]: price.head()
```

Out[19]:

Date	Close Price
2010-07-18	0.09
2010-07-19	0.08
2010-07-20	0.07
2010-07-21	0.08
2010-07-22	0.05

```
In [20]: price['2010']
```

Out[20]:

Date	Close Price
2010-07-18	0.09
2010-07-19	0.08
2010-07-20	0.07
2010-07-21	0.08
2010-07-22	0.05
2010-07-23	0.06
2010-07-24	0.05
2010-07-25	0.05
2010-07-26	0.06
2010-07-27	0.06
2010-07-28	0.06
2010-07-29	0.07
2010-07-30	0.06
2010-07-31	0.07
2010-08-01	0.06
2010-08-02	0.06
2010-08-03	0.06
2010-08-04	0.06
2010-08-05	0.06
2010-08-06	0.06
2010-08-07	0.06
2010-08-08	0.06

```
In [21]: price['2017-08-01']
```

Out[21]:

Date	Close Price
2017-08-01	2735.59

```
In [22]: price['2017-08-01':]
```

Out[22]:

Date	Close Price
2017-08-01 00:00:00	2735.59
2017-08-02 00:00:00	2723.58
2017-08-03 00:00:00	2814.36
2017-08-04 00:00:00	2883.68
2017-08-05 00:00:00	3301.76
2017-08-06 00:00:00	3255.00
2017-08-07 00:00:00	3431.97
2017-08-08 00:00:00	3453.16
2017-08-09 00:00:00	3377.54
2017-08-10 00:00:00	3445.28
2017-08-11 00:00:00	3679.61
2017-08-12 00:00:00	3917.65
2017-08-13 00:00:00	4111.20
2017-08-14 00:00:00	4382.74
2017-08-15 00:00:00	4204.43
2017-08-16 00:00:00	4425.30
2017-08-17 00:00:00	4316.34
2017-08-18 00:00:00	4159.46
2017-08-19 15:54:00	4062.17

```
In [23]: price.min()
Out[23]: Close Price    0.05
         dtype: float64
```

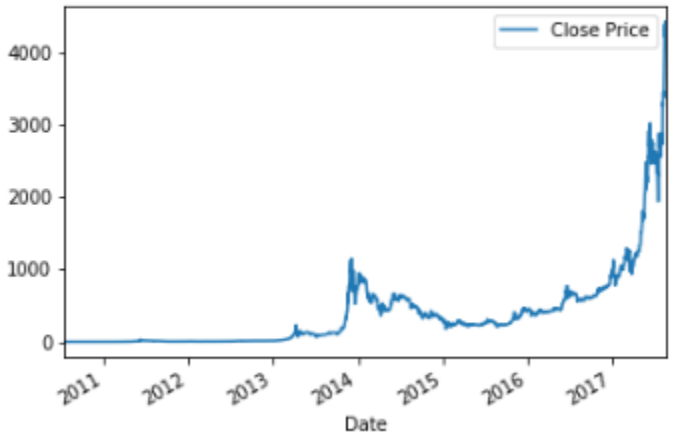
```
In [24]: price.max()
Out[24]: Close Price    4425.3
         dtype: float64
```

```
In [25]: price.describe()
```

Out[25]:

	Close Price
count	2590.000000
mean	381.646371
std	570.770109
min	0.050000
25%	8.752500
50%	233.705000
75%	545.132500
max	4425.300000

```
In [26]: price.plot()
Out[26]: <matplotlib.axes._subplots.AxesSubplot at 0x95d9ef0>
```




```
In [27]: price['2017'].plot()
Out[27]: <matplotlib.axes._subplots.AxesSubplot at 0x9e57ac8>
```

Import Modules

In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt

pd.options.mode.chained_assignment = None
%matplotlib inline
```

Dataset

Dataset used in this notebook are from blockchain.info

Example 1 - Total Bitcoins

Read in the data

In [2]:
```
bitcoins = pd.read_csv("total-bitcoins.csv", header=None, names=['Date', 'Bitcoins'])
```

Explore data

In [3]:
```
bitcoins.head()
```

Out[3]:

	Date	Bitcoins
0	2016-08-28 00:00:00	15841112.5
1	2016-08-29 00:00:00	15842975.0
2	2016-08-30 00:00:00	15845025.0
3	2016-08-31 00:00:00	15846700.0
4	2016-09-01 00:00:00	15848450.0

Explore data

In [3]:
```
bitcoins.head()
```

Out[3]:

	Date	Bitcoins
0	2016-08-28 00:00:00	15841112.5
1	2016-08-29 00:00:00	15842975.0
2	2016-08-30 00:00:00	15845025.0
3	2016-08-31 00:00:00	15846700.0
4	2016-09-01 00:00:00	15848450.0

In [4]:
```
bitcoins.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 364 entries, 0 to 363
Data columns (total 2 columns):
Date        364 non-null object
Bitcoins    364 non-null float64
dtypes: float64(1), object(1)
memory usage: 5.8+ KB
```

Cleanup and manipulate data

In [5]:
```
bitcoins["Date"] = pd.to_datetime(bitcoins["Date"], format="%Y-%m-%d")
```

In [6]:
```
bitcoins.index = bitcoins['Date']
del bitcoins['Date']
bitcoins.head()
```

In [6]:
```
bitcoins.index = bitcoins['Date']
del bitcoins['Date']
bitcoins.head()
```

Out[6]:

Date	Bitcoins
2016-08-28	15841112.5
2016-08-29	15842975.0
2016-08-30	15845025.0
2016-08-31	15846700.0
2016-09-01	15848450.0

In [7]:
```
bitcoins.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 364 entries, 2016-08-28 to 2017-08-26
Data columns (total 1 columns):
Bitcoins    364 non-null float64
dtypes: float64(1)
memory usage: 5.7 KB
```

In [8]:
```
bitcoins.head()
```

Out[8]:

Date	Bitcoins
2016-08-28	15841112.5
2016-08-29	15842975.0
2016-08-30	15845025.0
2016-08-31	15846700.0
2016-09-01	15848450.0

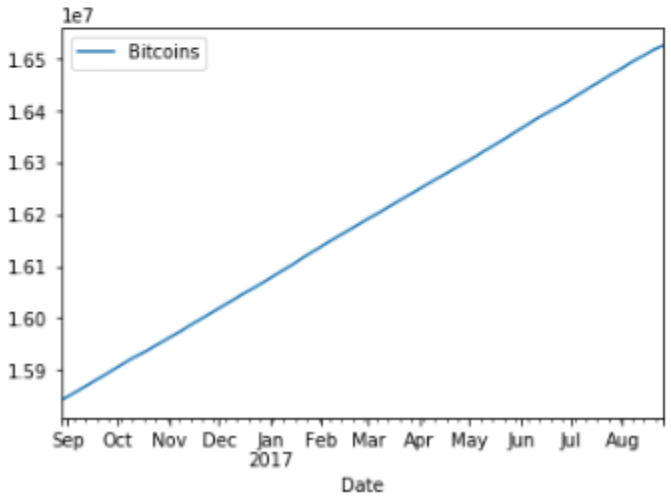
```
In [8]: bitcoins.head()
```

Out[8]:

Date	Bitcoins
2016-08-28	15841112.5
2016-08-29	15842975.0
2016-08-30	15845025.0
2016-08-31	15846700.0
2016-09-01	15848450.0

Visualize Data

```
In [9]: bitcoins.plot()
        plt.show()
```



Read in the data

```
In [15]: difficulty = pd.read_csv("difficulty.csv", header=None, names=['Date', 'Difficulty'])
```

Explore data

In [16]:
```
difficulty.head()
```

Out[16]:

	Date	Difficulty
0	2016-08-28 00:00:00	2.173755e+11
1	2016-08-29 00:00:00	2.184418e+11
2	2016-08-30 00:00:00	2.207559e+11
3	2016-08-31 00:00:00	2.207559e+11
4	2016-09-01 00:00:00	2.207559e+11

In [17]:
```
difficulty.info()
```

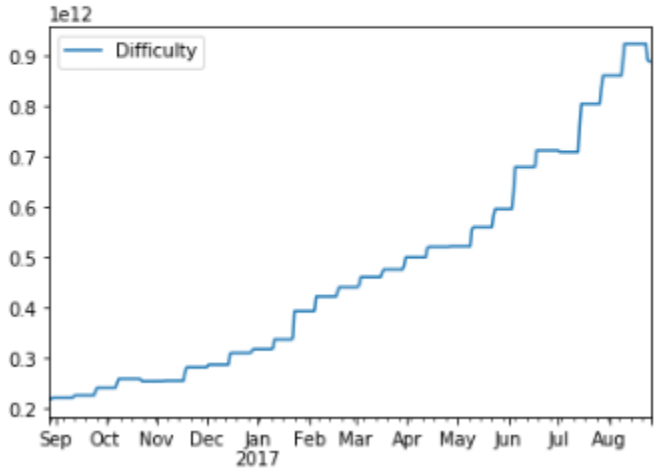
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 364 entries, 0 to 363
Data columns (total 2 columns):
Date          364 non-null object
Difficulty    364 non-null float64
dtypes: float64(1), object(1)
memory usage: 5.8+ KB
```

Cleanup and manipulate data

In [18]:
```
difficulty["Date"] = pd.to_datetime(difficulty["Date"], format="%Y-%m-%d")
difficulty.index = difficulty['Date']
del difficulty['Date']
```

Visualize Data

```
In [19]: difficulty.plot()
         plt.show()
```




```
In [5]: leaders.reset_index(inplace=True)
```

```
In [6]: leaders.head()
```

Out[6]:

	index	2014-11-05	2014-11-06	2014-11-07	
0	longesWinStreakPrizeInSatoshis	125000000	20000000	20000000	
1	longestWinNick	dooglus	leen	leen	
2	longestWinStreakWinnerHash	6848acbf6de4f8e9170bd9d1e28008e5	7ef585ba27ad8eea00d509622570e2d3	7ef585ba27ad8eea00d509622570e2d3	7ef585ba27ad8eea00
3	mostProfitNick	eleph	Throne	leen	
4	mostProfitPrizeInSatoshis	100000000	20000000	20000000	

5 rows × 708 columns

```
In [8]: leaders2.head()
```

Out[8]:

index	longesWinStreakPrizeInSatoshis	longestWinNick	longestWinStreakWinnerHash	mostProfitNick	mostProfitPrizeInSatoshis	mostProfitWi
2014-11-05	125000000	dooglus	6848acbf6de4f8e9170bd9d1e28008e5	eleph	100000000	bd8ecf07a31ac12466871227
2014-11-06	20000000	leen	7ef585ba27ad8eea00d509622570e2d3	Throne	20000000	73ffa73301d1df899c6c4a8d
2014-11-07	20000000	leen	7ef585ba27ad8eea00d509622570e2d3	leen	20000000	7ef585ba27ad8eea00d50962
2014-11-08	10000000	leen	7ef585ba27ad8eea00d509622570e2d3	Throne	10000000	73ffa73301d1df899c6c4a8d
2014-11-09	10000000	mercle	dcc787c0d6a92c45e94a0c4f381cd82c	Throne	10000000	73ffa73301d1df899c6c4a8d

```
In [9]: leaders2.reset_index(inplace=True)
```

```
In [10]: leaders2.head()
```

Out[10]:

index	index	longesWinStreakPrizeInSatoshis	longestWinNick	longestWinStreakWinnerHash	mostProfitNick	mostProfitPr
0	2014-11-05	125000000	dooglus	6848acbf6de4f8e9170bd9d1e28008e5	eleph	
1	2014-11-06	20000000	leen	7ef585ba27ad8eea00d509622570e2d3	Throne	
2	2014-11-07	20000000	leen	7ef585ba27ad8eea00d509622570e2d3	leen	
3	2014-11-08	10000000	leen	7ef585ba27ad8eea00d509622570e2d3	Throne	
4	2014-11-09	10000000	mercle	dcc787c0d6a92c45e94a0c4f381cd82c	Throne	

```
In [11]: leaders2["Date"] = pd.to_datetime(leaders2["index"], format="%Y-%m-%d")
```

```
In [12]: leaders2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 707 entries, 0 to 706
Data columns (total 17 columns):
index                                707 non-null object
longesWinStreakPrizeInSatoshis       707 non-null object
longestWinNick                       671 non-null object
longestWinStreakWinnerHash           707 non-null object
mostProfitNick                       695 non-null object
mostProfitPrizeInSatoshis            707 non-null object
mostProfitWinnerHash                 707 non-null object
mostWonNick                          697 non-null object
mostWonPrizeInSatoshis               707 non-null object
mostWonWinnerHash                    707 non-null object
unlikliestWinNick                    700 non-null object
unlikliestWinPrizeInSatoshis         707 non-null object
unlikliestWinStreakNick              675 non-null object
unlikliestWinStreakPrizeInSatoshis   707 non-null object
unlikliestWinStreakWinnerHash        707 non-null object
unlikliestWinWinnerHash              707 non-null object
Date                                 707 non-null datetime64[ns]
dtypes: datetime64[ns](1), object(16)
memory usage: 94.0+ KB
```

```
In [13]: leaders2.index = leaders2['Date']
         del leaders2['Date']
         del leaders2['index']
```

```
In [14]: leaders2.head()
```

Out[14]:

Date	index	longesWinStreakPrizeInSatoshis	longestWinNick	longestWinStreakWinnerHash	mostProfitNick	mostProfitPrizeInSatoshis	mostProfitWi
2014-11-05		125000000	dooglus	6848acbf6de4f8e9170bd9d1e28008e5	eleph	100000000	bd8ecf07a31ac12466871227
2014-11-06		20000000	leen	7ef585ba27ad8eea00d509622570e2d3	Throne	20000000	73ffa73301d1df899c6c4a8d
2014-11-07		20000000	leen	7ef585ba27ad8eea00d509622570e2d3	leen	20000000	7ef585ba27ad8eea00d50962
2014-11-08		10000000	leen	7ef585ba27ad8eea00d509622570e2d3	Throne	10000000	73ffa73301d1df899c6c4a8d
2014-11-09		10000000	mercle	dcc787c0d6a92c45e94a0c4f381cd82c	Throne	10000000	73ffa73301d1df899c6c4a8d