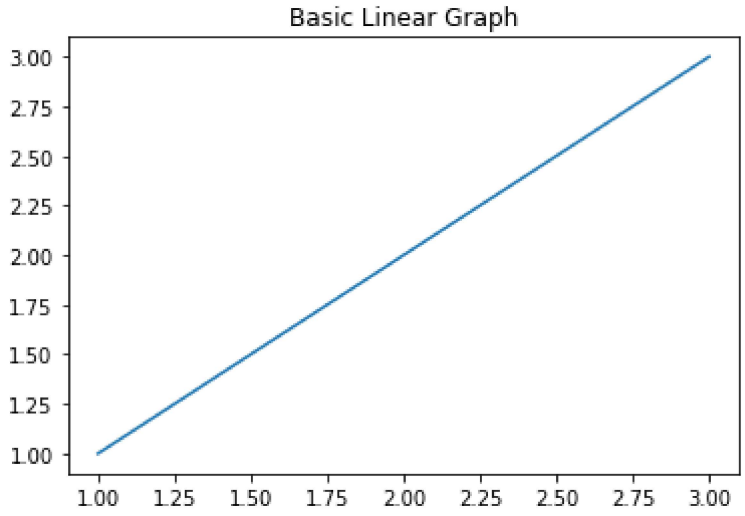
Day Objectives :-

04-07 2019

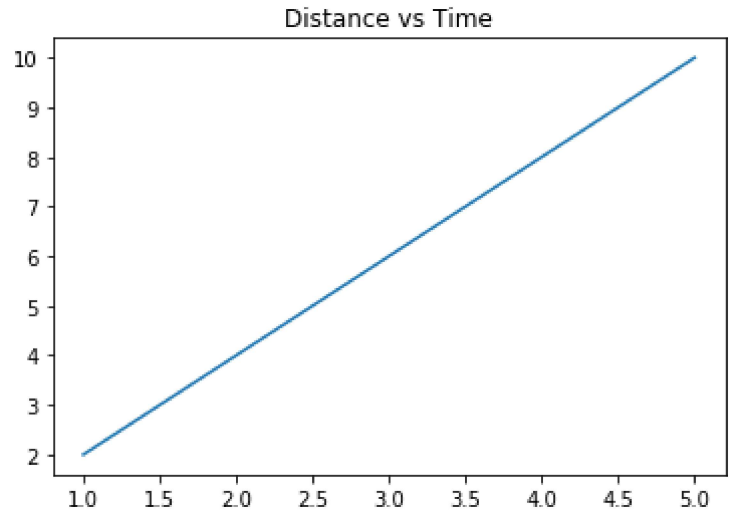
- Data Visualization using Matplotlin Library
  - Basic 2D plotting functions
  - Plotting 2D Graphs from CSV dataset
    - Income Dataset
    - AppStore Dataset

In [7]:
```
# Linear Graph

import matplotlib.pyplot as plt

plt.title('Basic Linear Graph')
plt.plot([1,2,3],[1,2,3])
plt.show()
```

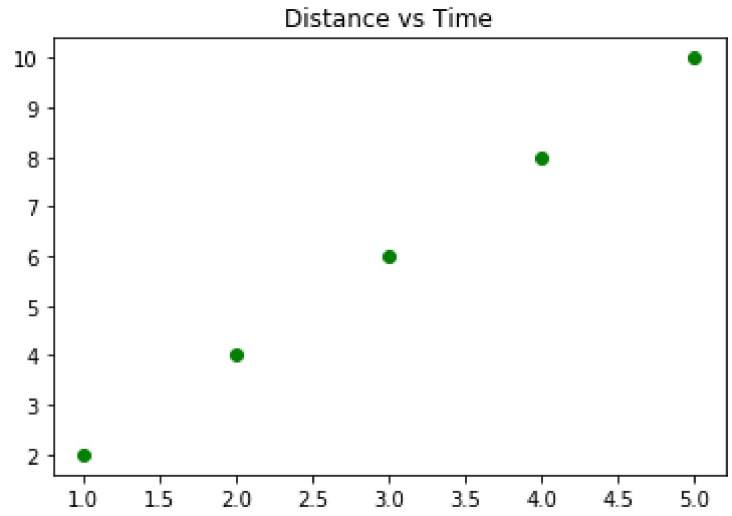


In [12]:

```
# Graph to plot distance and time

distance = list(range(2,11,2))
time = list(range(1,6))
#distance
#time
#d=2*t

plt.title("Distance vs Time")
plt.plot(time,distance)
plt.show()
```

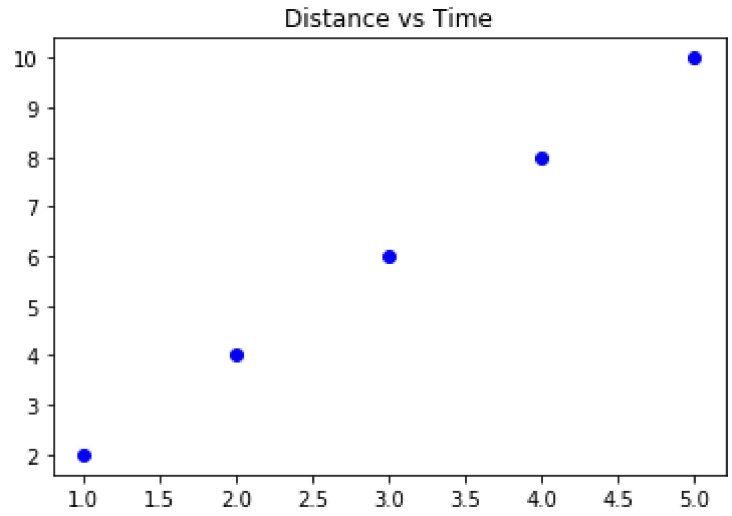


In [13]:
```
# Graph to plot distance and time with green colour dots

distance = list(range(2,11,2))
time = list(range(1,6))
#distance
#time
#d=2*t

plt.title("Distance vs Time")
plt.plot(time,distance,'go')
plt.show()
```

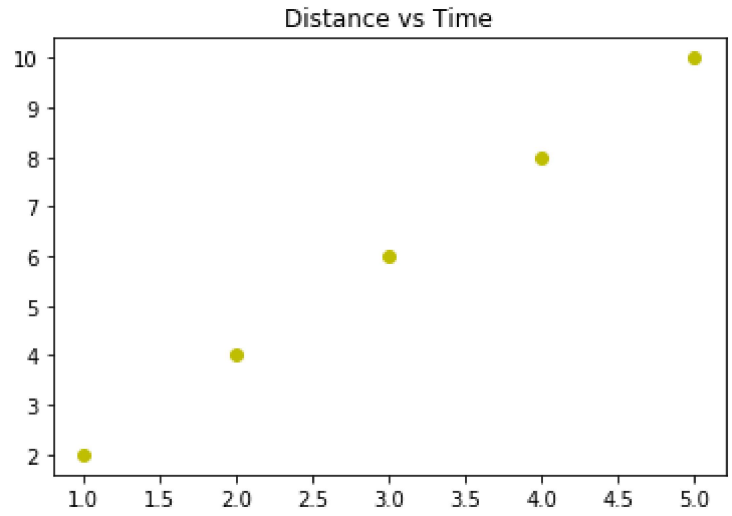


In [14]:

```
# Graph to plot distance and time with y colour dots

distance = list(range(2,11,2))
time = list(range(1,6))
#distance
#time
#d=2*t

plt.title("Distance vs Time")
plt.plot(time,distance,'bo')
plt.show()
```

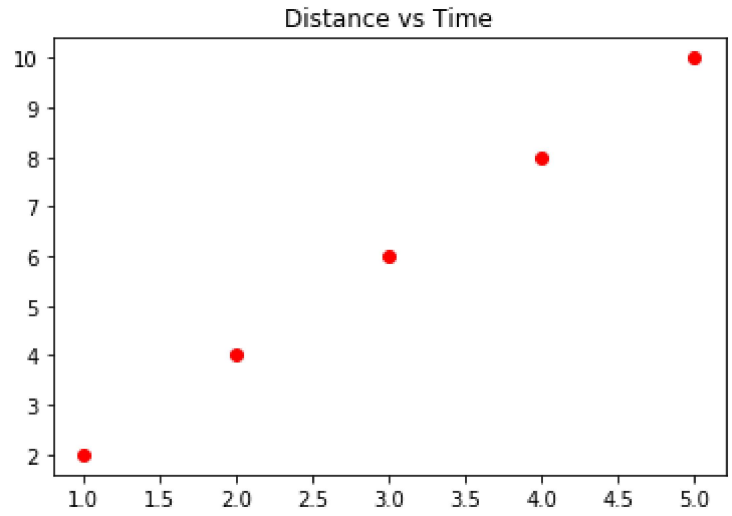


In [15]:
```
# Graph to plot distance and time with y colour dots

distance = list(range(2,11,2))
time = list(range(1,6))
#distance
#time
#d=2*t

plt.title("Distance vs Time")
plt.plot(time,distance,'yo')
plt.show()
```



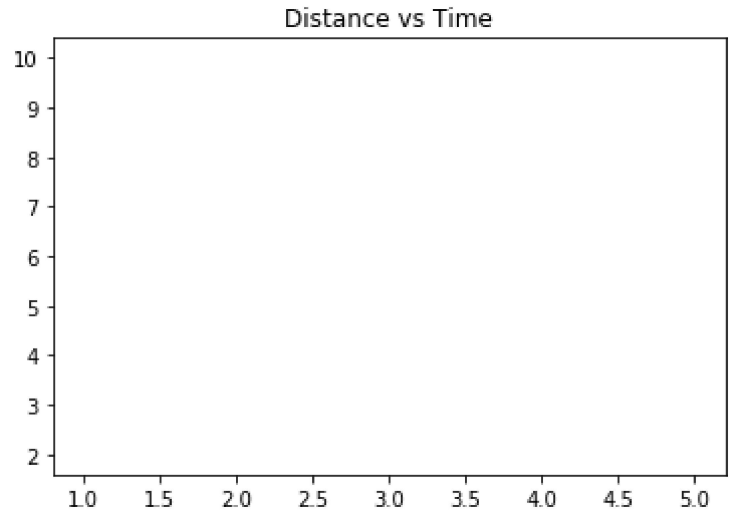
In [16]:

```
# Graph to plot distance and time with r colour dots

distance = list(range(2,11,2))
time = list(range(1,6))
#distance
#time
#d=2*t

plt.title("Distance vs Time")
plt.plot(time,distance,'ro')
plt.show()
```



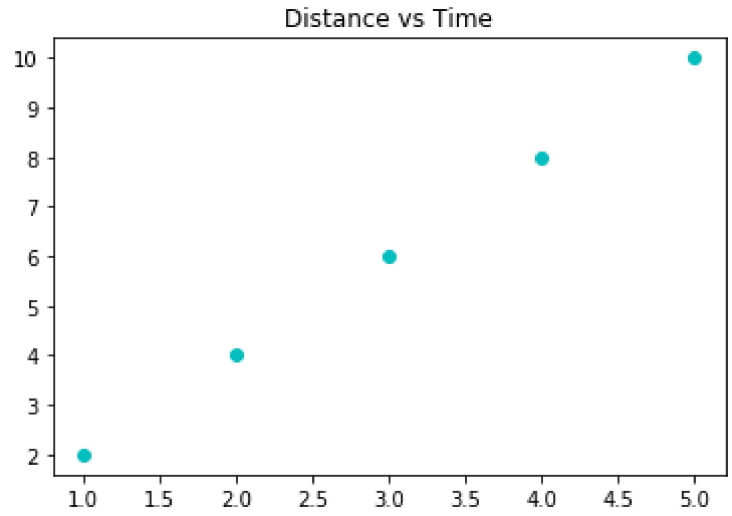
In [17]:

```
# Graph to plot distance and time with w colour dots

distance = list(range(2,11,2))
time = list(range(1,6))
#distance
#time
#d=2*t

plt.title("Distance vs Time")
plt.plot(time,distance,'wo')
plt.show()
```



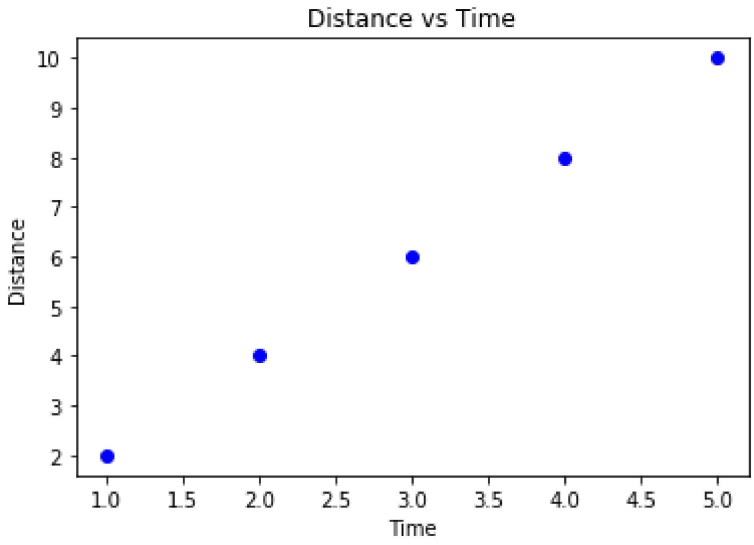
In [18]:

```
# Graph to plot distance and time with y colour dots

distance = list(range(2,11,2))
time = list(range(1,6))
#distance
#time
#d=2*t

plt.title("Distance vs Time")
plt.plot(time,distance,'co')
plt.show()
```

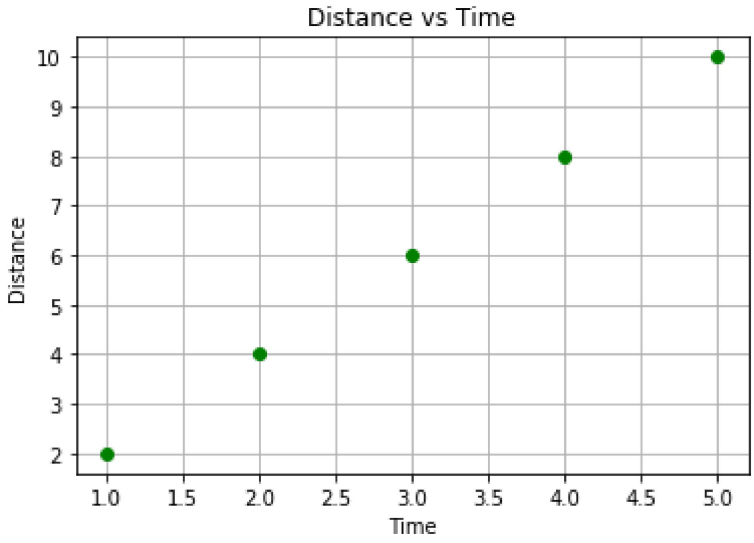


In [22]:

```
# Graph to plot distance and time with y colour dots

distance = list(range(2,11,2))
time = list(range(1,6))
#distance
#time
#d=2*t

plt.title("Distance vs Time")
plt.xlabel('Time')
plt.ylabel('Distance')
plt.plot(time,distance,'bo')
plt.show()
```



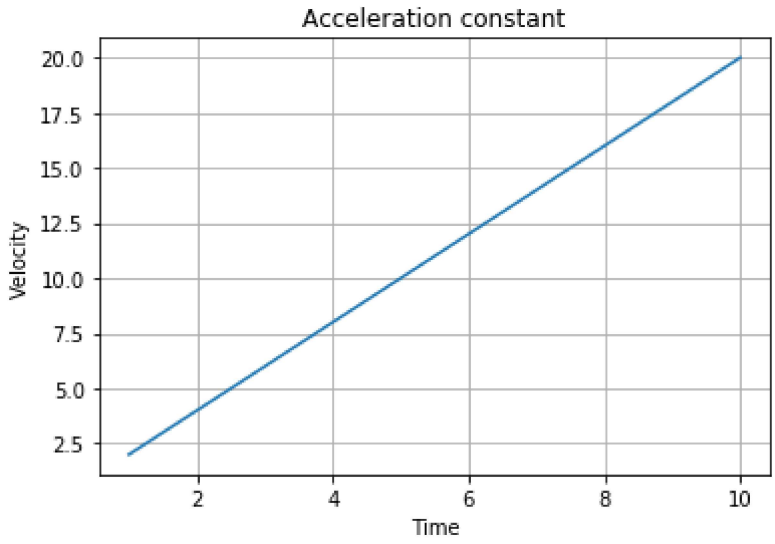
In [23]:
```
# Graph to plot distance and time with y colour dots

distance = list(range(2,11,2))
time = list(range(1,6))
#distance
#time
#d=2*t

plt.title("Distance vs Time")
plt.xlabel('Time')
plt.ylabel('Distance')
plt.plot(time,distance,'go')
plt.grid()
plt.show()
```



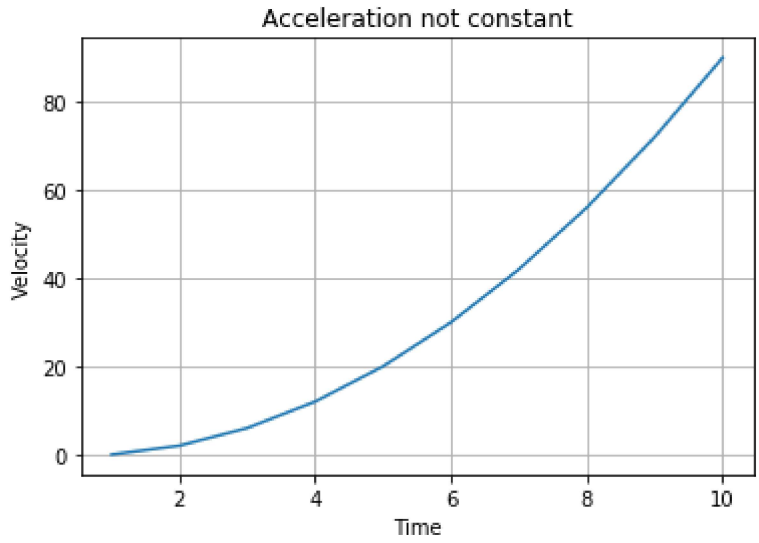
In [24]:
```
# v = a t

v = [2 * i for i in range(1,11)]
v
```

Out[24]:
```
[2, 4, 6, 8, 10, 12, 14, 16, 18, 20]
```

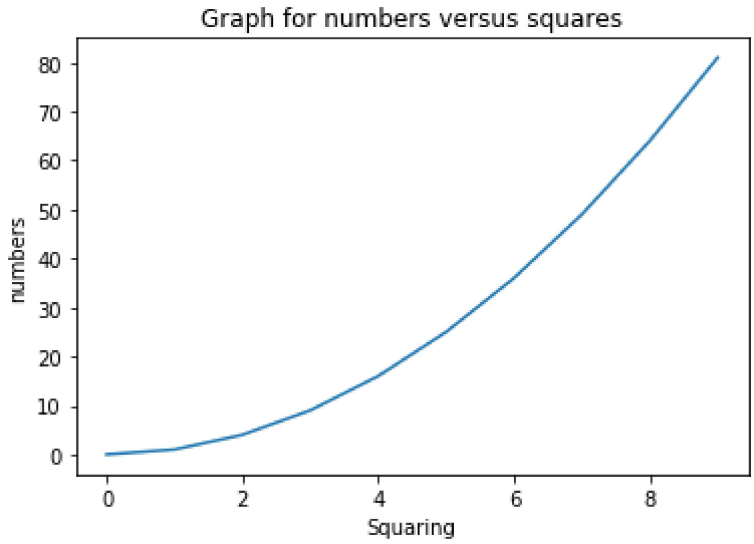
In [28]:
```
# when acceleration is constant while time changes

t=list(range(1,11))
v=[2*i for i in t]
plt.title('Acceleration constant')
plt.xlabel('Time')
plt.ylabel('Velocity')
plt.plot(t,v)
plt.grid()
plt.show()
```



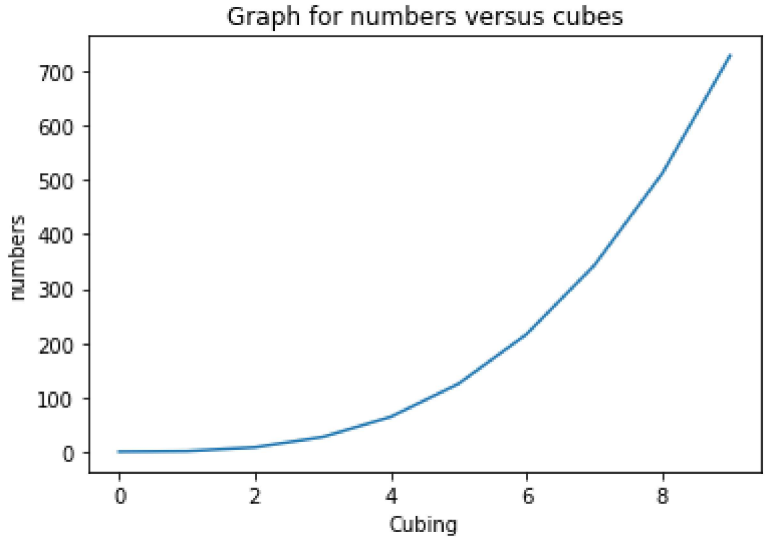
In [32]:
```
# when acceleration is not constant while time changes

t=list(range(1,11))
a=list(range(1,len(t)+1))

v=[a[i]*i for i in range( len(t))]
plt.title('Acceleration not constant')
plt.xlabel('Time')
plt.ylabel('Velocity')
plt.plot(t,v)
plt.grid()
plt.show()
```



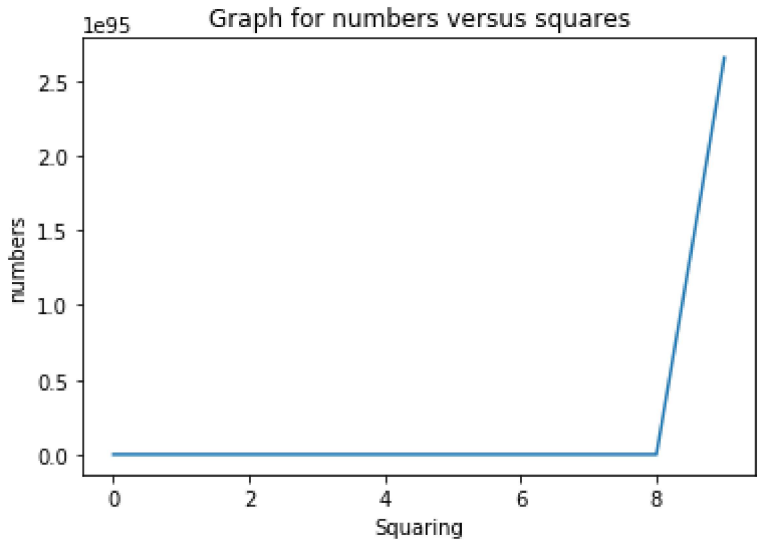
In [47]:
```
# Graph for numbers versus squares
n=list(range(10))

sqn = [i**2 for i in n]

plt.plot(n,sqn)
plt.title("Graph for numbers versus squares")
plt.xlabel('Squaring')
plt.ylabel('numbers')
plt.show()
```



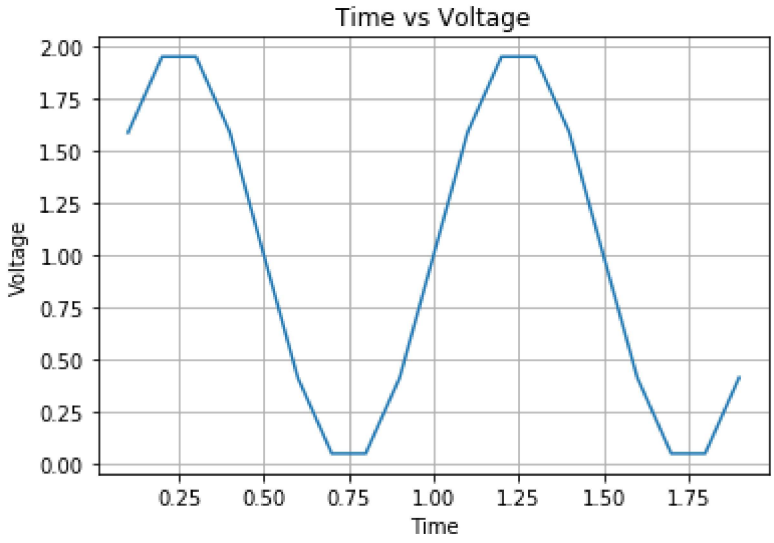
In [46]:
```
# Graph for numbers versus cubes
n=list(range(10))

sqn = [i**3 for i in n]

plt.plot(n,sqn)
plt.title("Graph for numbers versus cubes")
plt.xlabel('Cubing')
plt.ylabel('numbers')
plt.show()
```



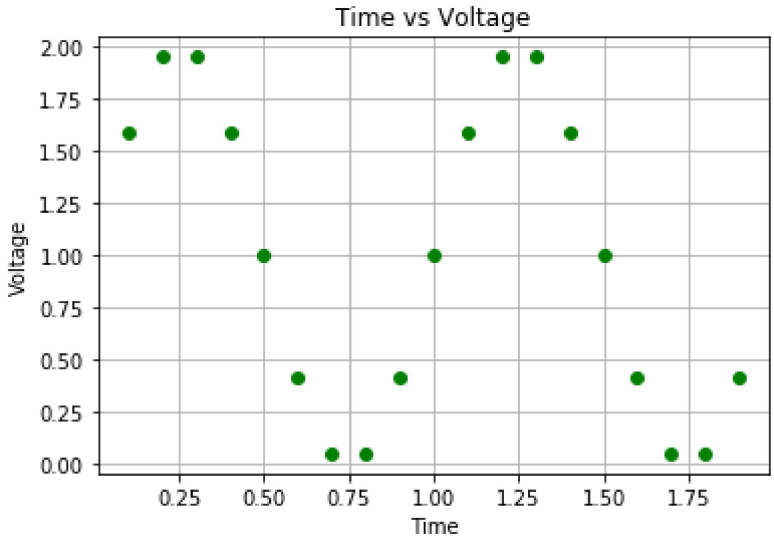
In [50]:
```
# Graph for numbers versus squares
n=list(range(10))

sqn = [i**100 for i in n]

plt.plot(n,sqn)
plt.title("Graph for numbers versus squares")
plt.xlabel('Squaring')
plt.ylabel('numbers')
plt.show()
```



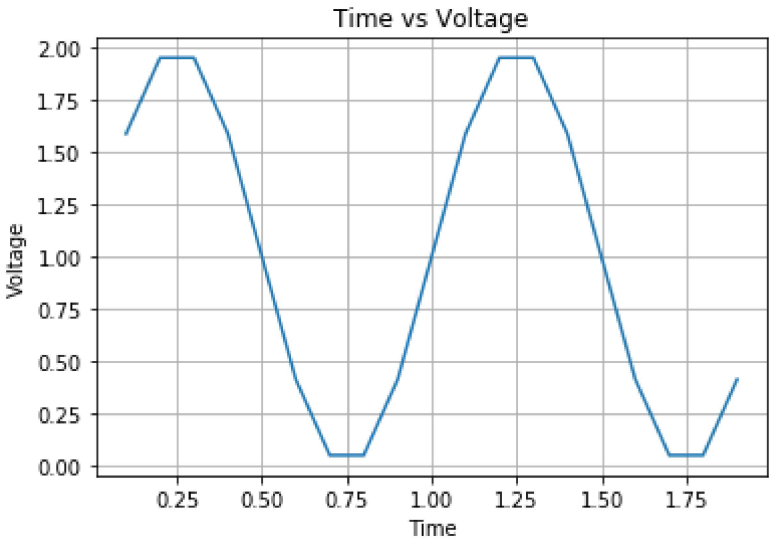
In [69]:
```
# Voltage  -Time Method 1

import numpy as np

t = [i/10 for i in range(1,20)]

v =[ 1 + np.sin(2*np.pi*i) for i in t]
plt.title('Time vs Voltage')
plt.plot(t,v)
plt.grid()
plt.xlabel('Time')
plt.ylabel('Voltage')
plt.savefig('Images\Voltage-Time.png')
plt.show()
```



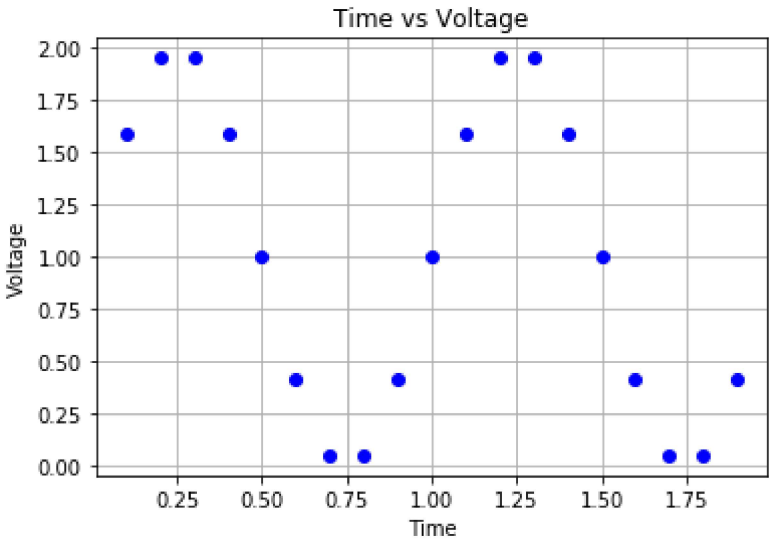
In [72]:
```
# Voltage  -Time Metho1

import numpy as np

t = [i/10 for i in range(1,20)]

v =[ 1 + np.sin(2*np.pi*i) for i in t]
plt.title('Time vs Voltage')
plt.plot(t,v,'go')
plt.grid()
plt.xlabel('Time')
plt.ylabel('Voltage')
plt.savefig('Images\Voltage.png')
plt.show()
```



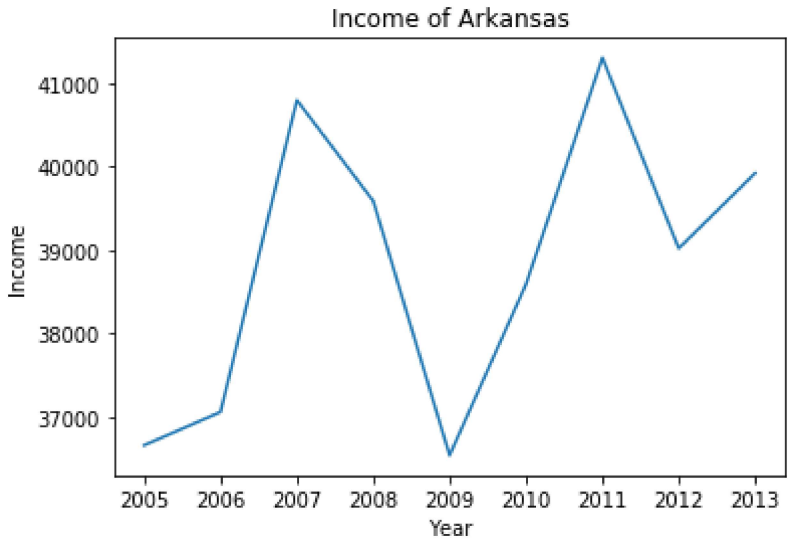
In [63]:
```
# Voltage  -Time Method 2

import numpy as np

t=np.arange(0.1,2.0,0.1)
v =[ 1 + np.sin(2*np.pi*i) for i in t]
plt.title('Time vs Voltage')
plt.plot(t,v)
plt.grid()
plt.xlabel('Time')
plt.ylabel('Voltage')
plt.show()
```



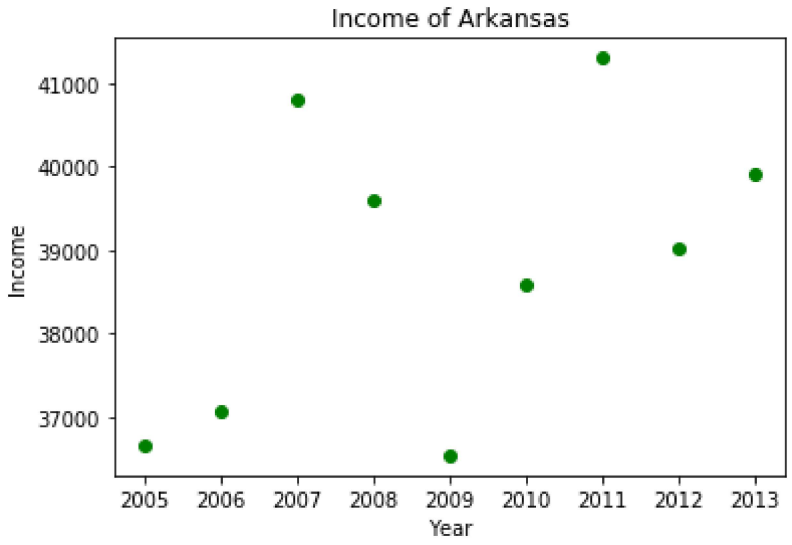
In [62]:
```
# Voltage  -Time Method 2

import numpy as np

t=np.arange(0.1,2.0,0.1)
v =[ 1 + np.sin(2*np.pi*i) for i in t]
plt.title('Time vs Voltage')
plt.plot(t,v,'bo')
plt.grid()
plt.xlabel('Time')
plt.ylabel('Voltage')
plt.show()
```



In [75]:
```
# Plot Income of Arkansas from  2005 to 2013
import pandas as pd

df=pd.read_csv('DataFiles\income.csv')
years = df.columns[2:]

incomeArkansas = df.values[3,2:]

plt.title('Income of Arkansas')
plt.xlabel('Year')
plt.ylabel('Income')
plt.plot(years,incomeArkansas)
plt.show()
```

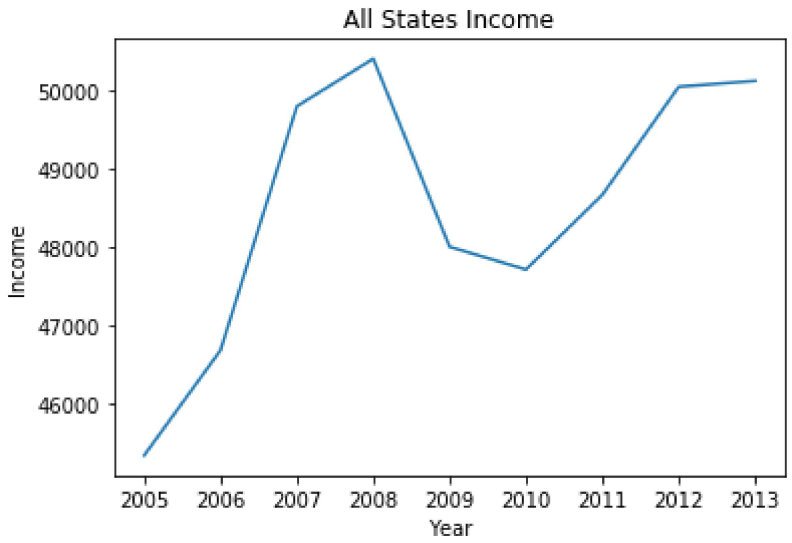


In [76]:
```
# Plot Income of Arkansas from  2005 to 2013
import pandas as pd

df=pd.read_csv('DataFiles\income.csv')
years = df.columns[2:]

incomeArkansas = df.values[3,2:]

plt.title('Income of Arkansas')
plt.xlabel('Year')
plt.ylabel('Income')
plt.plot(years,incomeArkansas,'go')
plt.show()
```



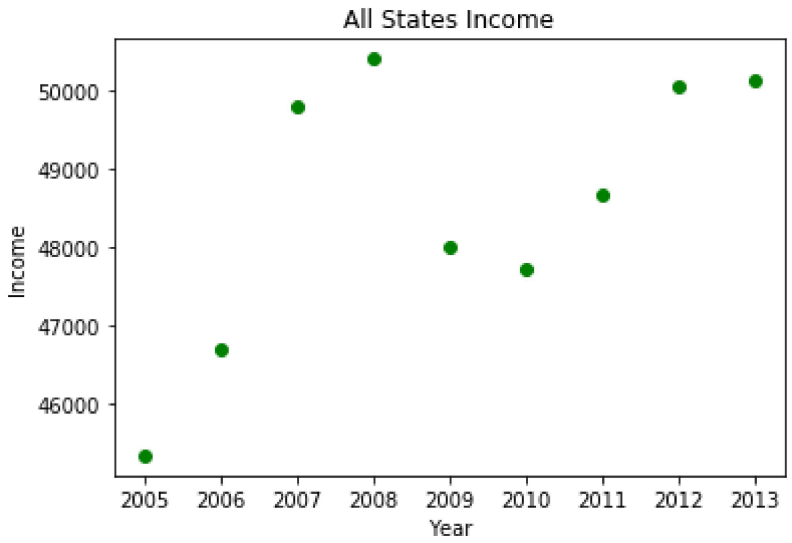
In [87]:
```
# Income of all the states in 2005 to 2013



incomeallstates =  []

for i in df.columns[2:]:
    incomeallstates.append(df[i].mean())

print(incomeallstates)

plt.plot(years,incomeallstates)
plt.title("All States Income")
plt.xlabel('Year')
plt.ylabel('Income')
plt.show()
```

```
[45339.8, 46680.6, 49789.8, 50395.8, 47999.0, 47709.4, 48662.2, 50038.8, 50113.
4]
```



In [88]:
```
# Income of all the states in 2005 to 2013



incomeallstates =  []

for i in df.columns[2:]:
    incomeallstates.append(df[i].mean())

print(incomeallstates)

plt.plot(years,incomeallstates,'go')
plt.title("All States Income")
plt.xlabel('Year')
plt.ylabel('Income')
plt.show()
```

```
[45339.8, 46680.6, 49789.8, 50395.8, 47999.0, 47709.4, 48662.2, 50038.8, 50113.
4]
```



In [ ]:
```

```