Lec1

- dots -> vertices/nodes
- lines -> edges/links
  Bifercation results in a disconneted graph with 2 components

Lec2

- A disconnected graph with two components each of having 25 nodes is improbable(In case of friendship in a classroom)

Lec3

- ipython - Interactive Python

In [1]:

```
print range(10)
```

```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

In [5]:

```
import random as rd
print rd.randrange(1, 10) # Range : 1 <= x < 10
print rd.random() # Range from 0 to 1
print rd.randint(1, 10) # Range : 1 <= x <= 10
```

```
5
0.297706743891
4
```

In [7]:

```
list1 = [rd.randint(1, 10) for x in range(10)]
print list1
list1.sort()
print list1
list1.reverse()
print list1
print "Length : ", len(list1)
```

```
[7, 5, 8, 5, 3, 2, 9, 2, 10, 8]
[2, 2, 3, 5, 5, 7, 8, 8, 9, 10]
[10, 9, 8, 8, 7, 5, 5, 3, 2, 2]
Length :  10
```

In [8]:

```
#reload(packagename) after modifying it...
```

Dictionary Implemented as Hash tables -> faster

In [2]:

```
d = {'annie':20, 'yoyo':35, 'sid':65}
```

In [3]:

```
d
```

Out[3]:

```
{'annie': 20, 'sid': 65, 'yoyo': 35}
```

In [4]:

```
d['sid'] = 56
```

In [10]:

```
d['gok'] = 20
```

In [11]:

```
d
```

Out[11]:

```
{'annie': 20, 'gok': 20, 'sid': 56, 'yoyo': 35}
```

In [12]:

```
#deleting gok
del d['gok']
print d
```

```
{'yoyo': 35, 'annie': 20, 'sid': 56}
```

In [15]:

```
#d.clear()#It will remove only the elements
print d.has_key('annie')
print d.keys()
print d.values()
print d.items()
```

```
True
['yoyo', 'annie', 'sid']
[35, 20, 56]
[('yoyo', 35), ('annie', 20), ('sid', 56)]
```

In [16]:

```
for i in d:
    print i
```

```
yoyo
annie
sid
```

In [17]:

```
#Create square dic
d1 = {x:x**2 for x in range(100)}
print d1
```

```
{0: 0, 1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: 36, 7: 49, 8: 64, 9: 81, 10:
100, 11: 121, 12: 144, 13: 169, 14: 196, 15: 225, 16: 256, 17: 289, 1
8: 324, 19: 361, 20: 400, 21: 441, 22: 484, 23: 529, 24: 576, 25: 625,
26: 676, 27: 729, 28: 784, 29: 841, 30: 900, 31: 961, 32: 1024, 33: 10
89, 34: 1156, 35: 1225, 36: 1296, 37: 1369, 38: 1444, 39: 1521, 40: 16
00, 41: 1681, 42: 1764, 43: 1849, 44: 1936, 45: 2025, 46: 2116, 47: 22
09, 48: 2304, 49: 2401, 50: 2500, 51: 2601, 52: 2704, 53: 2809, 54: 29
16, 55: 3025, 56: 3136, 57: 3249, 58: 3364, 59: 3481, 60: 3600, 61: 37
21, 62: 3844, 63: 3969, 64: 4096, 65: 4225, 66: 4356, 67: 4489, 68: 46
24, 69: 4761, 70: 4900, 71: 5041, 72: 5184, 73: 5329, 74: 5476, 75: 56
25, 76: 5776, 77: 5929, 78: 6084, 79: 6241, 80: 6400, 81: 6561, 82: 67
24, 83: 6889, 84: 7056, 85: 7225, 86: 7396, 87: 7569, 88: 7744, 89: 79
21, 90: 8100, 91: 8281, 92: 8464, 93: 8649, 94: 8836, 95: 9025, 96: 92
16, 97: 9409, 98: 9604, 99: 9801}
```

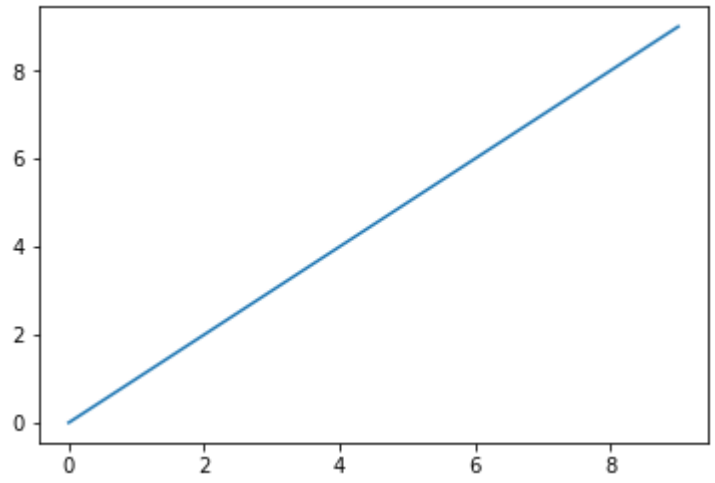
In [18]:

```
#Create square dic for even nos.
d1 = {x:x**2 for x in range(100) if x%2 == 0}
print d1
```

```
{0: 0, 2: 4, 4: 16, 6: 36, 8: 64, 10: 100, 12: 144, 14: 196, 16: 256,
 18: 324, 20: 400, 22: 484, 24: 576, 26: 676, 28: 784, 30: 900, 32: 10
24, 34: 1156, 36: 1296, 38: 1444, 40: 1600, 42: 1764, 44: 1936, 46: 21
16, 48: 2304, 50: 2500, 52: 2704, 54: 2916, 56: 3136, 58: 3364, 60: 36
00, 62: 3844, 64: 4096, 66: 4356, 68: 4624, 70: 4900, 72: 5184, 74: 54
76, 76: 5776, 78: 6084, 80: 6400, 82: 6724, 84: 7056, 86: 7396, 88: 77
44, 90: 8100, 92: 8464, 94: 8836, 96: 9216, 98: 9604}
```

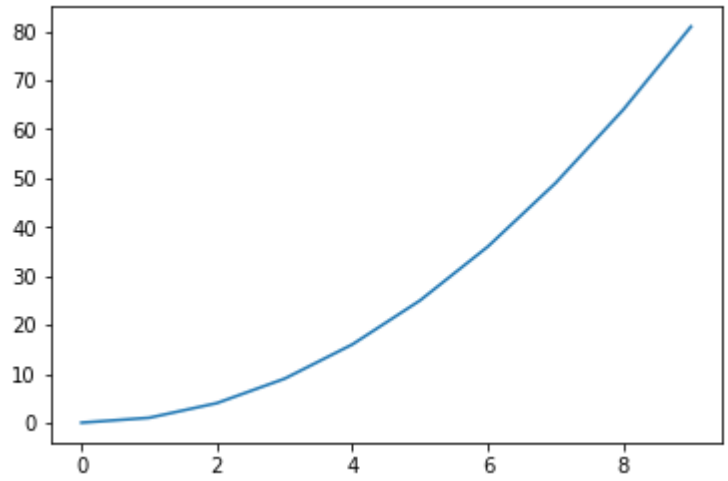
In [20]:

```
import matplotlib.pyplot as plt
plt.plot([x for x in range(10)])#default x values are used
plt.show()
```



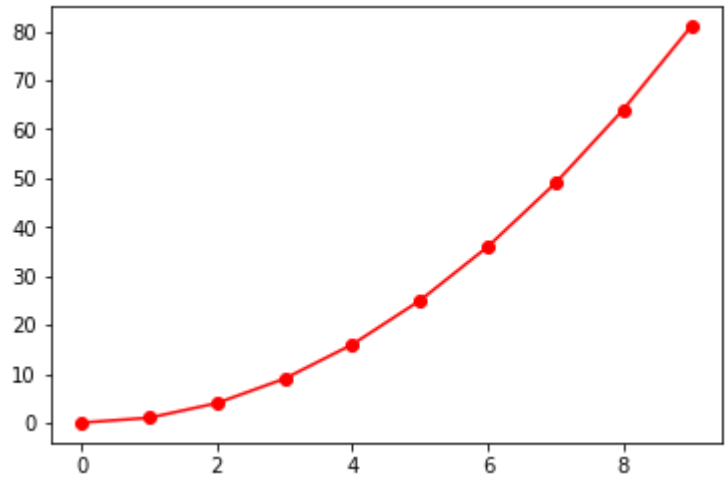
In [21]:

```
plt.plot([x for x in range(10)], [x**2 for x in range(10)])
plt.show()
```



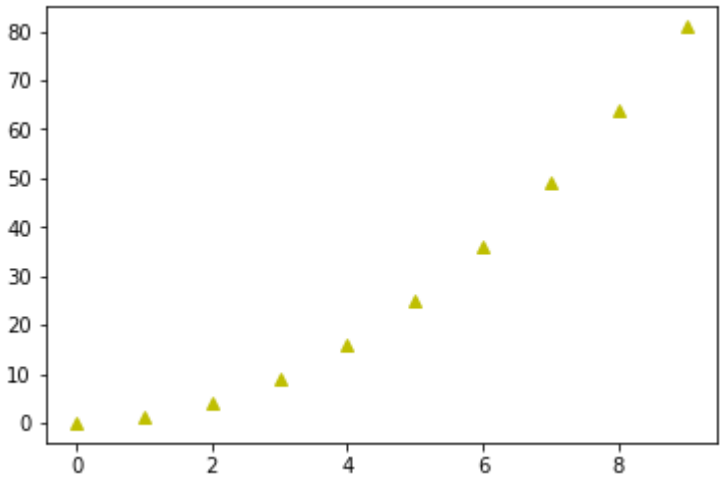
In [22]:

```
plt.plot([x for x in range(10)], [x**2 for x in range(10)], 'ro-')
plt.show()
```



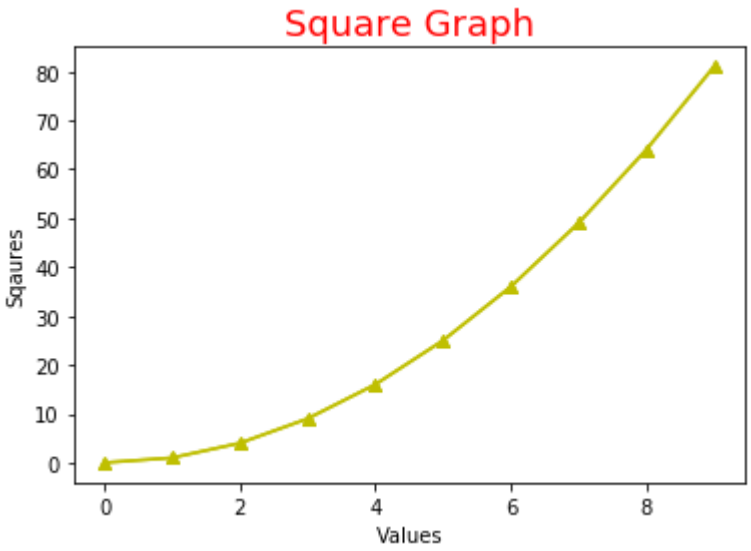
In [23]:

```
plt.plot([x for x in range(10)], [x**2 for x in range(10)], 'y^')
plt.show()
```



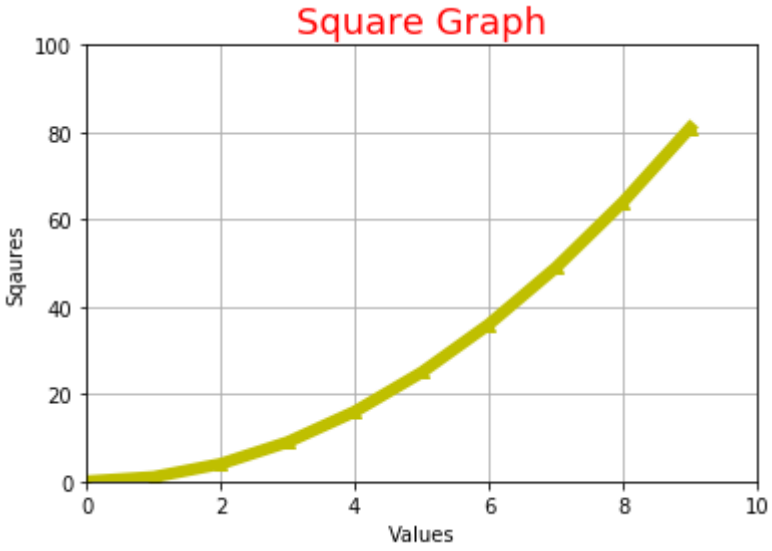
In [29]:

```
plt.plot([x for x in range(10)], [x**2 for x in range(10)], 'y^-')
plt.xlabel('Values')
plt.ylabel('Sqaures')
plt.title('Square Graph', fontsize = 18, color = 'red')
plt.show()
```



In [36]:

```
plt.plot([x for x in range(10)], [x**2 for x in range(10)], 'y^-', linewidth = 6.0)
plt.xlabel('Values')
plt.ylabel('Squares')
plt.title('Square Graph', fontsize = 18, color = 'red')
plt.axis([0,10,0,100])
plt.grid(True)
plt.show()
```



In [37]:

```
import numpy as np
x = [i for i in range(20)]
x = np.array(x)
plt.plot(x, x**2, 'r-', x, x**3, 'y-', x, x**4, 'b+')
plt.show()
```