Gradient Descent Practical Demonstration

Programmed By : Rithik Tripathi

Connect with me on Linkedin (https://www.linkedin.com/in/rithik-tripathi-data-scientist/)

In [1]:

```
import numpy as np
import matplotlib.pyplot as plt
```

In [2]:

```
x = np.arange(10)
x
```

Out[2]:

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

In [3]:

```
# creating y which is a function of x

# y = f(x - 5)^2

y = (x-5)**2
y
```

Out[3]:

```
array([25, 16,  9,  4,  1,  0,  1,  4,  9, 16])
```

Goal of Gradient Descent

Given a function f(x) , we want to find the value of x that Minimizes f

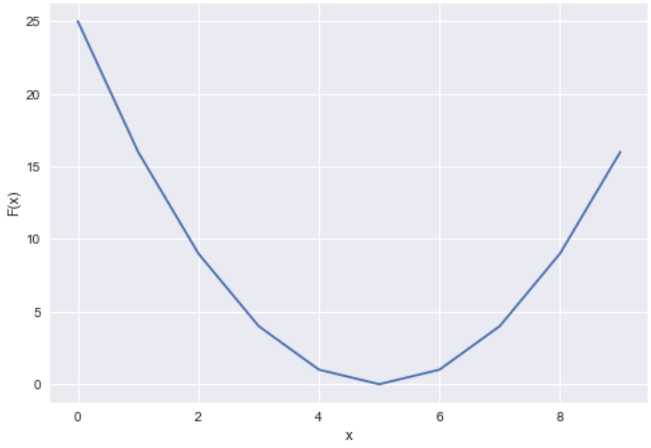
Visualizing X vs Y

In [4]:

```
plt.style.use("seaborn")
plt.plot(x,y)
plt.xlabel("x")
plt.ylabel("F(x)")
```

Out[4]:

```
Text(0, 0.5, 'F(x)')
```

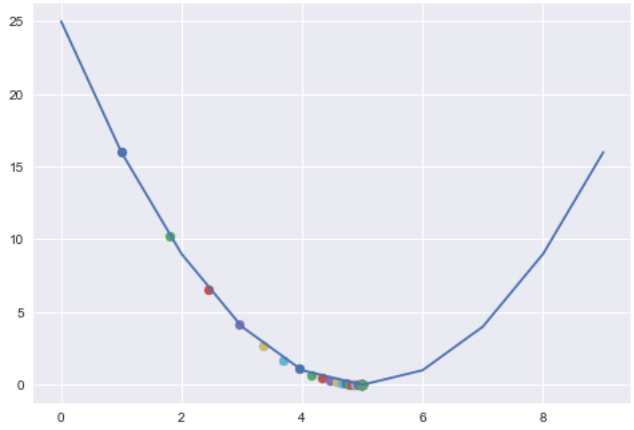



```
As y = f(x - 5)^2 => Gradient of this is => 2(x-5) which is simply the differentiation w.r.t x of the function
```

Implementing Gradient Descent

In [5]:

```
random_value = 0
Learning_rate = 0.1
error = []

plt.plot(x,y)

for i in range(50):
    gradient = 2*(random_value-5)
    random_value = random_value - (Learning_rate * gradient)

    Y = (random_value-5)**2
    error.append(Y)

    print("Random Value -> " ,random_value, "\nCost -> ", Y)

    plt.scatter(random_value,Y)

    print("\n=================")
```




```
we can observe that the step gradients are converging because the magnitude of (del f(x) / del x) is decreasing and can observe how our gradient descent is converging
```

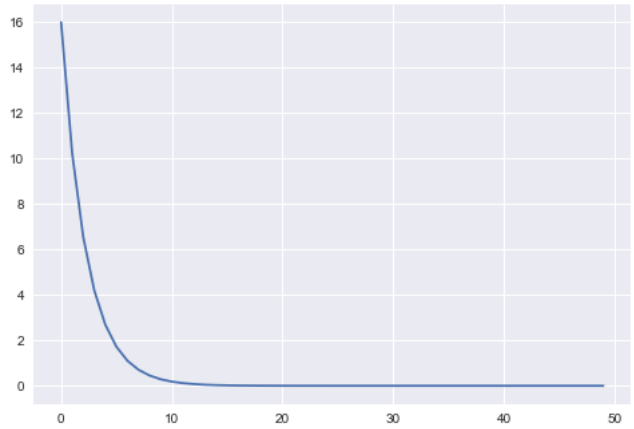
Plotting the errors

In [6]:

```
plt.plot(error)
```

Out[6]:

```
[<matplotlib.lines.Line2D at 0x1ca20387df0>]
```




```
this demonstrates that as we increase the number of steps, after around 10, the error is being saturated
```