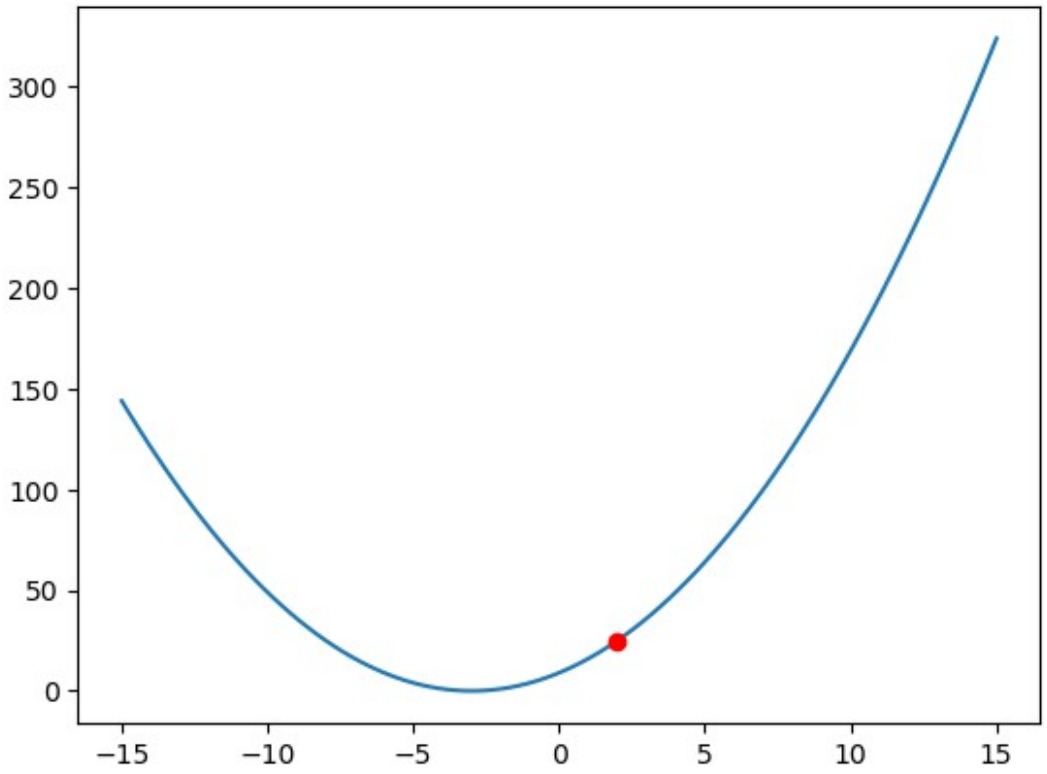
```
import numpy as np
import pandas as pd
import sympy as sym
import matplotlib as pyplot
from matplotlib import pyplot

def objective(x):
    return (x+3)**2

def derivative(x):
    return 2*(x+3)

def gradient(alpha,start,max_iter):
    x_list=list()
    x=start
    x_list.append(x)
    for i in range(max_iter):
        gradi=derivative(x)
        x=x-(alpha*gradi)
        x_list.append(x)
    return x_list
x=sym.symbols('x')
expr=(x+3)**2.0
grad=sym.Derivative(expr,x)
print("{}".format(grad.doit()))
grad.doit().subs(x,2)
```

```
2.0*(x + 3)**1.0
```

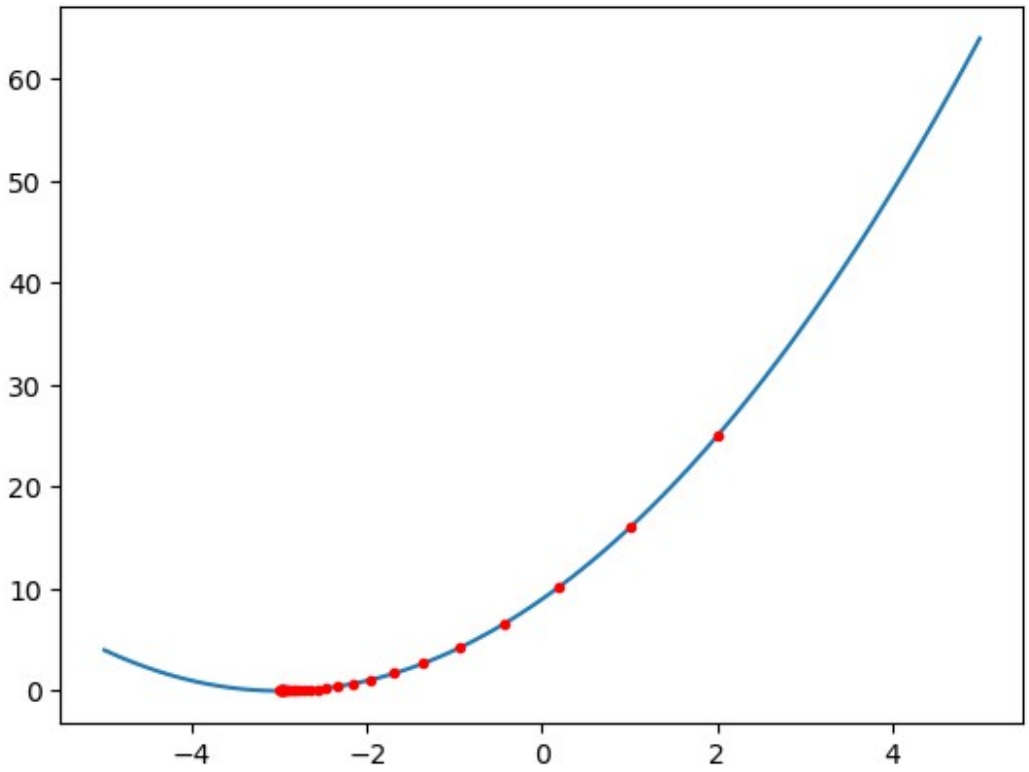
```
10.0000000000000
```

```
alpha=0.1
start=2
max_iter=30
x=sym.symbols('x')
expr=(x+3)**2

x_cor=np.linspace(-15,15,100)
pyplot.plot(x_cor,objective(x_cor))
pyplot.plot(2,objective(2),'ro')
```

```
[<matplotlib.lines.Line2D at 0x1ada72c8bf0>]
```




```
x=gradient(alpha,start,max_iter)
x_cor=np.linspace(-5,5,100)
pyplot.plot(x_cor,objective(x_cor))

x_arr=np.array(x)
pyplot.plot(x_arr,objective(x_arr),'.',color='red')
pyplot.show()
```




```
#Initialize Parameters
cur_x = 2
rate = 0.01
precision = 0.000001
previous_step_size = 1
max_iters = 1000
iters = 0
df = lambda x : 2 * (x + 3) #Gradient of our function i.e (x + 3)²

#Run a loop to perform gradient Descent
while previous_step_size > precision and iters < max_iters:
    prev_x = cur_x
    cur_x -= rate * df(prev_x)
    previous_step_size = abs(prev_x - cur_x)
    iters += 1
print("Local Minima Occurs at :",cur_x)
```

```
Local Minima Occurs at : -2.999951128099859
```