Implement Gradient Descent Algorithm to find the local minima of a function. For example, find the local minima of the function y=(x+3)2 starting from the point x=2.

In [1]:
```
import numpy as np
import sympy as sym
import matplotlib.pyplot as plt
```

In [2]:
```
x = sym.symbols('x')
exp = ((x+3.0)**2.0)
grad = sym.Derivative(exp,x).doit()
grad.subs(x,2)
```

Out[2]: 10.0

In [3]:
```
start = 2
alpha = 0.1
exp = ((x+3.0)**2.0)
grad = sym.Derivative(exp,x).doit()
max_iter = 1000
```

In [4]:
```
def gradient_descent(exp, start, alpha, max_iter):
    precision = 0.000001
    iter = 0
    prev_step_size = 1

    x_list = list()
    curr_x = start
    x_list.append(curr_x)

    while prev_step_size > precision and iter < max_iter:
        prev_x = curr_x
        gradient = grad.subs(x,curr_x)
        curr_x = curr_x - (alpha * gradient)
        iter = iter + 1
        prev_step_size = abs(curr_x - prev_x)
        x_list.append(curr_x)
        print("Iteration No : ", iter, "\n Value of X : ", curr_x)

    return x_list
```

In [5]:
```
X = gradient_descent(exp, start, alpha, max_iter)
```

```
 Value of X :  -2.99603859187429
Iteration No :  33
 Value of X :  -2.99683087349943
Iteration No :  34
 Value of X :  -2.99746469879954
Iteration No :  35
 Value of X :  -2.99797175903963
Iteration No :  36
 Value of X :  -2.99837740723171
Iteration No :  37
 Value of X :  -2.99870192578537
Iteration No :  38
 Value of X :  -2.99896154062829
Iteration No :  39
 Value of X :  -2.99916923250263
Iteration No :  40
 Value of X :  -2.99933538600211
Iteration No :  41
 Value of X :  -2.99946830880169
Iteration No :  42
```

In [6]:
```
print("The local minima occurs at : ", X[len(X)-1])
```

```
The local minima occurs at :  -2.99999607681142
```

In [7]:
```
def objective(x):
    return (x+3)**2
```

In [8]:
```
x_cord = np.linspace(-4,4,10)
plt.plot(x_cord, objective(x_cord))

x_arr = np.array(X)
plt.plot(x_arr, objective(x_arr), '.-', c = "r")
```

Out[8]:
```
[<matplotlib.lines.Line2D at 0x10f858040>]
```