In [ ]:

Practical No:3
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]:
```
x=2 #start
lr=0.01 #learning rate
precision = 0.000001
previous_step_size = 1
max_iter = 10000
iters =0
gf = lambda x: (x + 3) ** 2
```

In [2]:
```
import matplotlib.pyplot as plt
```

In [3]:
```
gd=[]
```

In [4]:
```
while precision < previous_step_size and iters < max_iter: prev = x
    x = x- lr * gf(prev)
    previous_step_size = abs(x - prev) iters += 1
    print('Iteration',iters,'Value',x) gd.append(x)
```

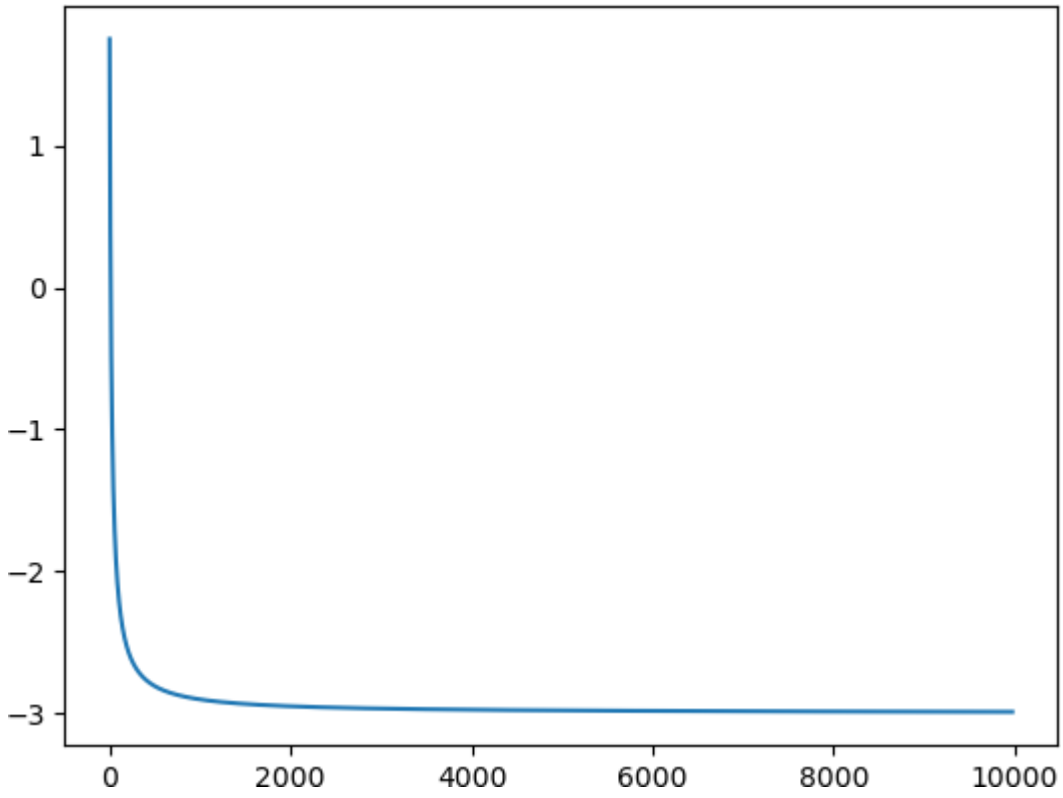
```
Iteration 1 Value 1.75
Iteration 2 Value 1.524375
Iteration 3 Value 1.31967530859375
Iteration 4 Value 1.133079360877005
Iteration 5 Value 0.9622559108439301
Iteration 6 Value 0.8052611918137536
Iteration 7 Value 0.6604610644345152
Iteration 8 Value 0.5264713123921045
Iteration 9 Value 0.4021113132208596
Iteration 10 Value 0.28636769934540596
Iteration 11 Value 0.1783655727923978
Iteration 12 Value 0.07734549564927831
Iteration 13 Value -0.017355057346650715
Iteration 14 Value -0.10631676588600673
Iteration 15 Value -0.19005079247993095
Iteration 16 Value -0.26900893796835756
Iteration 17 Value -0.34359205977732477
Iteration 18 Value -0.41415709122610556
Iteration 19 Value -0.4810229267146679
```

In [5]:
```
print('Local Minima:',x)
```

```
Local Minima: -2.990001240409911
```

In [6]:
```
plt.plot(range(len(gd)), gd)
```

Out[6]: [<matplotlib.lines.Line2D at 0x24ccf0cf2d0>]



In [ ]:
```

```