

```
metodo_main.py  class_smartphone.py ×
1  class Nodo:
2      def __init__(self, valor, siguiente= None):
3          self.valor = valor
4          self.siguiente = siguiente
5
6      def get_valor(self):
7          return self.valor
8
9      def set_valor(self, valor):
10         self.valor = valor
11
12     def get_siguiente(self):
13         return self.siguiente
14
15     def set_siguiente(self, siguiente):
16         self.siguiente = siguiente
17
18     class Smartphone:
19         def __init__(self):
20             self.cabeza = None
21
22         def esta_vacia(self):
23             if self.cabeza is None:
24                 print('Está vacía')
25             else:
26                 print('No está vacía')
27
```

```
metodo_main.py  class_smartphone.py ×  
18 class Smartphone:  
27  
28     def get_tamano(self):  
29         contador = 0  
30         actual = self.cabeza  
31         while actual is not None:  
32             contador += 1  
33             actual = actual.get_siguiete()  
34         return contador  
35  
36     def agregar_al_final(self, dato):  
37         nodo_nuevo = Nodo(dato)  
38         if self.cabeza is None:  
39             self.cabeza = nodo_nuevo  
40         else:  
41             actual = self.cabeza  
42             while actual.get_siguiete() is not None:  
43                 actual = actual.siguiete  
44             actual.set_siguiete(nodo_nuevo)  
45  
46     def agregar_al_inicio(self, valor):  
47         nodo_nuevo = Nodo(valor)  
48         nodo_nuevo.siguiete = self.cabeza  
49         self.cabeza = nodo_nuevo  
50  
51  
52
```

```
18 class Smartphone:
53     def agregar_despues_de(self, valor, referencia):
54         nodo_nuevo = Nodo(valor)
55         actual = self.cabeza
56         while actual.dato != referencia:
57             actual = actual.siguiente
58             nodo_nuevo.siguiente = actual.siguiente
59             actual.siguiente = nodo_nuevo
60
61     def eliminar_posicion(self, posicion):
62         contador = 1
63         actual = self.cabeza
64         if posicion == 1:
65             self.cabeza = actual.get_siguiente()
66         else:
67             while contador < posicion - 1:
68                 actual = actual.get_siguiente()
69                 contador += 1
70             actual.set_siguiente(actual.get_siguiente().get_siguiente())
71
72     def eliminar_primer(self):
73         actual = self.cabeza
74         self.cabeza = actual.get_siguiente()
75
76     def eliminar_el_final(self):
77         tamaño = self.get_tamaño()
```

```
75     def eliminar_el_final(self):
76         tamaño = self.get_tamaño()
77         actual = self.cabeza
78         contador = 2
79         while contador < tamaño:
80             actual = actual.get_siguiente()
81             contador += 1
82         actual.set_siguiente(None)
83
84     def buscar_valor(self, valor):
85         actual = self.cabeza
86         contador = 1
87         while actual.valor != valor:
88             actual = actual.get_siguiente()
89             contador += 1
90         print(f' El valor {valor} se ubica en la posición {contador} ')
91
92     def actualizar(self, a_buscar, valor):
93         actual = self.cabeza
94         while actual.valor != a_buscar:
95             actual = actual.siguiente
96         actual.set_valor(valor)
97
98     def transversal(self):
99         actual = self.cabeza
100
```

```
18 class Smartphone:
84
85     def buscar_valor(self, valor):
86         actual = self.cabeza
87         contador = 1
88         while actual.valor != valor:
89             actual = actual.get_siguiete()
90             contador += 1
91         print(f' El valor {valor} se ubica en la posición {contador}')
92
93     def actualizar(self, a_buscar, valor):
94         actual = self.cabeza
95         while actual.valor != a_buscar:
96             actual = actual.siguiete
97             actual.set_valor(valor)
98
99     def transversal(self):
100         actual = self.cabeza
101         while actual is not None:
102             print(actual.valor, end=' -> ')
103             actual = actual.get_siguiete()
104             print('None')
```

## Método main

```
ED Estructura de Datos Version control me
metodo_main.py x class_smartphone.py
1 from class_smartphone import *
2 smartphones = Smartphone()
3 smartphones.agregar_al_final('Iphone')
4 smartphones.agregar_al_final('Samsung')
5 smartphones.agregar_al_final('LG')
6 smartphones.agregar_al_final('Huawei')
7 smartphones.agregar_al_final('Motorola')
8 smartphones.transversal()
9 smartphones.eliminar_posicion(2)
10 smartphones.transversal()
11 smartphones.actualizar(a_buscar='Motorola', valor='Xiaomi')
12 smartphones.agregar_al_inicio('Nexus')
13 smartphones.agregar_al_final('Oppo')
14 smartphones.transversal()
15 smartphones.eliminar_primer()
16 smartphones.transversal()
17
```

Run metodo\_main

```

"C:\Cursos\Python\Estructuradedatos\Estructura de Datos\.env\Scripts\python.exe
Iphone -> Samsung -> LG -> Huawei -> Motorola -> None
Iphone -> LG -> Huawei -> Motorola -> None
Nexus -> Iphone -> LG -> Huawei -> Xiaomi -> Oppo -> None
Iphone -> LG -> Huawei -> Xiaomi -> Oppo -> None

Process finished with exit code 0

```