

Lista Doble

```
class NodoDoble:
    def __init__(self, valor, siguiente=None, anterior=None):
        self.valor = valor
        self.siguiente = siguiente
        self.anterior = anterior

    def get_valor(self):
        return self.valor

    def set_valor(self, valor):
        self.valor=valor

    def get_siguiente(self):
        return self.siguiente

    def set_siguiente(self, siguiente):
        self.siguiente = siguiente

    def get_anterior(self):
        return self.anterior

    def set_anterior(self, anterior):
        self.anterior =anterior
```

```
class DoubleLinkedList:

    def __init__(self):
        self.cabeza= None
        self cola= None

    def esta_vacia(self):
        if self.cabeza is None:
            vacia= True
        else:
            vacia= False
        return vacia

    def get_tamano(self):
        contador = 0
        actual = self.cabeza
        while actual is not None:
            contador += 1
            actual = actual.get_siguiente()
        return contador
```

```
def agregar_al_inicio(self,valor):
    nodo_nuevo = NodoDoble(valor)
    if self.cabeza is None:
        self.cabeza= self cola= nodo_nuevo
    else:
        nodo_nuevo.siguiente= self.cabeza
        self.cabeza= nodo_nuevo
```

```
def agregar_al_final(self, valor):
    nodo_nuevo = NodoDoble(valor)
    if self.cabeza is None:
        self.cabeza= self.cola= nodo_nuevo
    else:
        self.cola.siguiente= nodo_nuevo
        self.cola=nodo_nuevo
```

```
def agregar_despues_de(self, valor, referencia):
    nodo_nuevo = NodoDoble(valor)
    actual = self.cabeza
    while actual.dato != referencia:
        actual = actual.siguiente
    nodo_nuevo.siguiente = actual.siguiente
    actual.siguiente = nodo_nuevo
```

```
def posicion(self, posicion):
    contador=2
    actual= self.cabeza
    if posicion > 1:
        while contador < posicion + 1:
            actual = actual.siguiente
            contador += 1
        print(f' En la posicion {posicion} se encutra el valor {actual.siguiente}')
```

```
def eliminar_el_primero(self):
    actual = self.cabeza
    self.cabeza = actual.get_siguiente()
```

```

def eliminar_el_final(self):
    tamaño = self.get_tamaño()
    actual = self.cabeza
    contador = 2
    while contador < tamaño:
        actual = actual.get_siguiete()
        contador += 1
    actual.set_siguiete(None)
    self.cola=actual.get_siguiete()

def eliminar_posicion(self, posicion):
    actual = self.cabeza
    contador = 1
    if posicion == 1:
        self.cabeza = actual.get_siguiete()
    else:
        while contador < posicion - 1:
            actual = actual.get_siguiete()
            contador += 1
        actual.set_siguiete(actual.get_siguiete().get_siguiete())

def buscar_valor(self, valor):
    actual = self.cabeza
    contador = 1
    while actual.valor != valor:
        actual = actual.get_siguiete()
        contador += 1
    print(f' El valor {valor} se ubica en la posición {contador}')

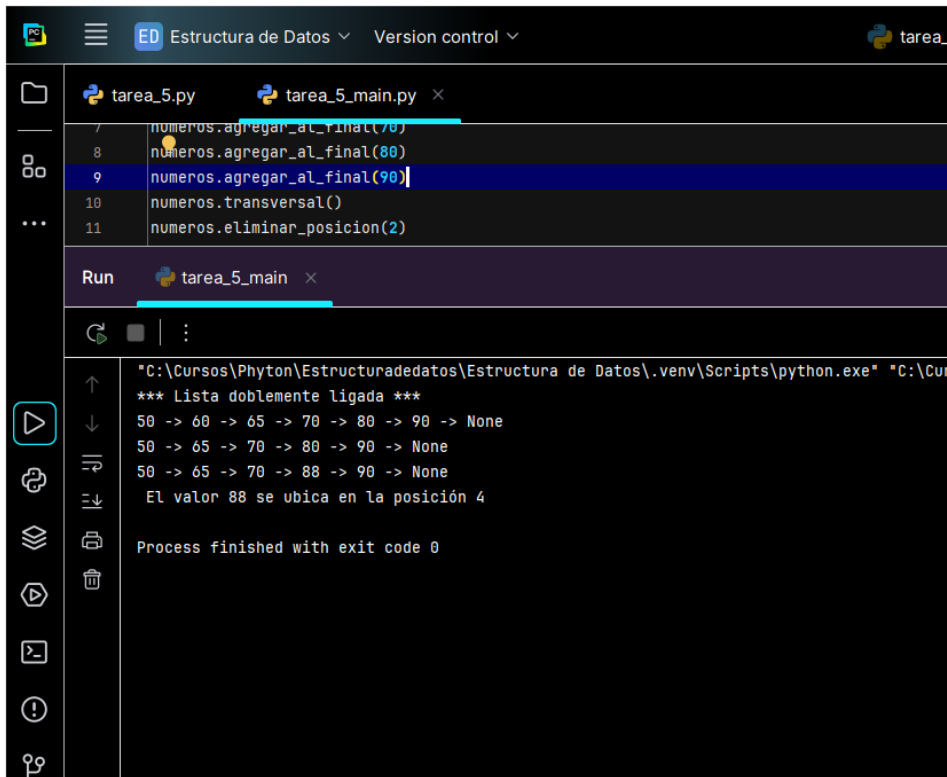
def actualizar(self,a_buscar, valor):
    contador=1
    actual=self.cabeza
    if a_buscar == 1:
        actual.set_valor(valor)
    else:
        while contador < a_buscar:
            actual = actual.get_siguiete()
            contador += 1
        actual.set_valor(valor)

def transversal(self):
    actual = self.cabeza
    while actual is not None:
        print(actual.valor, end=' -> ')
        actual = actual.get_siguiete()
    print('None')

```

Código de main

```
from tarea_5 import *
print('*** Lista doblemente ligada ***')
numeros = DoubleLinkedList()
numeros.agregar_al_inicio(50)
numeros.agregar_al_final(60)
numeros.agregar_al_final(65)
numeros.agregar_al_final(70)
numeros.agregar_al_final(80)
numeros.agregar_al_final(90)
numeros.transversal()
numeros.eliminar_posicion(2)
numeros.transversal()
numeros.actualizar(4, 88)
numeros.transversal()
numeros.buscar_valor(88)
```



```
7 numeros.agregar_al_final(70)
8 numeros.agregar_al_final(80)
9 numeros.agregar_al_final(90)
10 numeros.transversal()
11 numeros.eliminar_posicion(2)

Run tarea_5_main x

"C:\Cursos\Phyton\Estructuradedatos\Estructura de Datos\.venv\Scripts\python.exe" "C:\Cur
*** Lista doblemente ligada ***
50 -> 60 -> 65 -> 70 -> 80 -> 90 -> None
50 -> 65 -> 70 -> 80 -> 90 -> None
50 -> 65 -> 70 -> 88 -> 90 -> None
El valor 88 se ubica en la posición 4

Process finished with exit code 0
```