

Algoritmos y Estructuras de Datos I

Algoritmos de Ordenamiento Básico

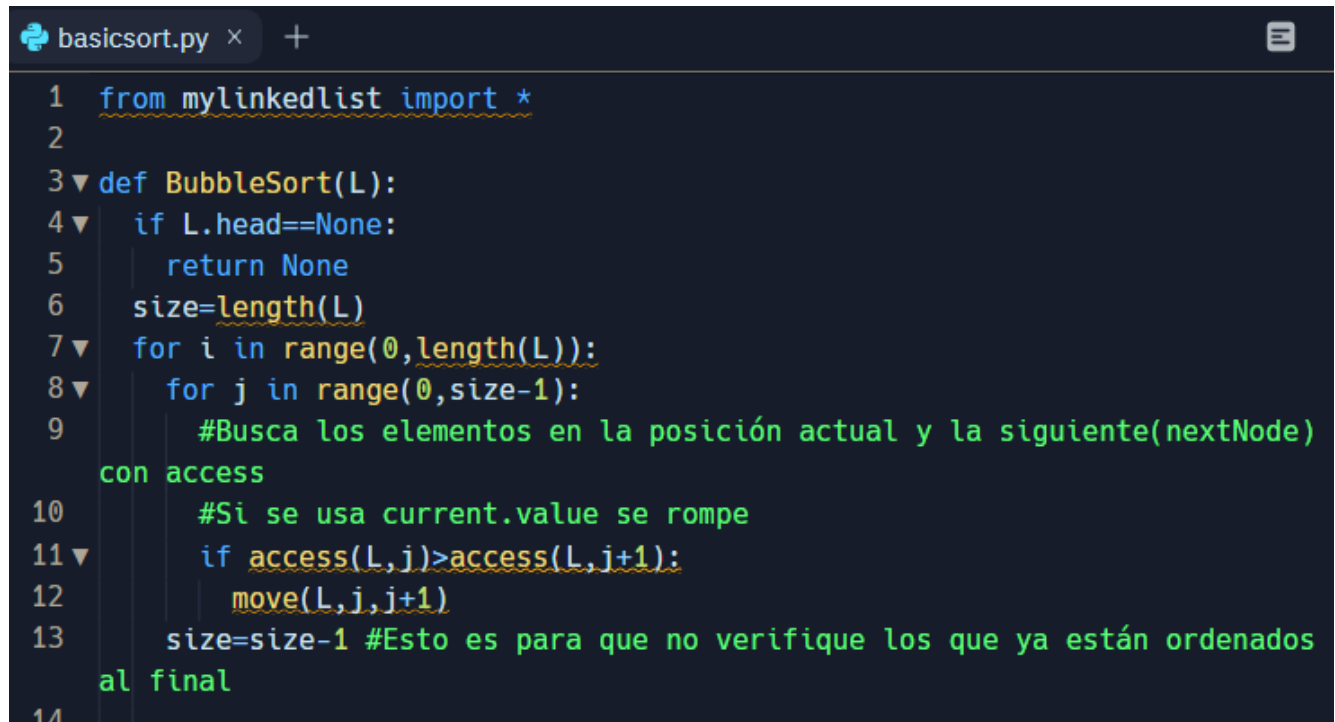
Link Repl.it:

<https://repl.it.com/@Paulonia/Martinez13866Ordenamiento-Basico#basicsort.py>

Ejercicio 1:

Implementar los algoritmos de ordenamiento básico:

• BubbleSort

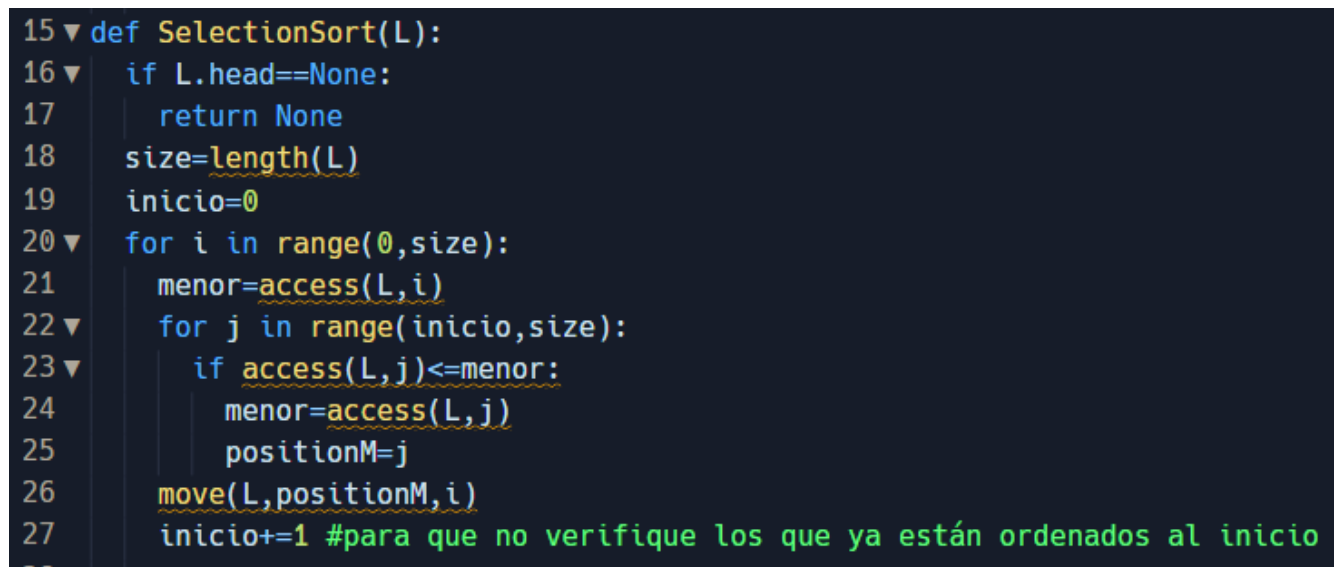


```

1  from mylinkedlist import *
2
3  def BubbleSort(L):
4      if L.head==None:
5          return None
6          size=length(L)
7      for i in range(0,length(L)):
8          for j in range(0,size-1):
9              #Busca los elementos en la posición actual y la siguiente(nextNode)
con access
10             #Si se usa current.value se rompe
11             if access(L,j)>access(L,j+1):
12                 move(L,j,j+1)
13             size=size-1 #Esto es para que no verifique los que ya están ordenados
al final
14

```

• SelectionSort



```

15 def SelectionSort(L):
16     if L.head==None:
17         return None
18     size=length(L)
19     inicio=0
20     for i in range(0,size):
21         menor=access(L,i)
22         for j in range(inicio,size):
23             if access(L,j)<=menor:
24                 menor=access(L,j)
25                 positionM=j
26         move(L,positionM,i)
27         inicio+=1 #para que no verifique los que ya están ordenados al inicio
28

```

• InsertionSort

```

30 ▼ def InsertionSort(L):
31 ▼     if L.head==None:
32         return None
33     size=length(L)
34 ▼     for i in range(1,size):
35 ▼         for j in range(0,i):
36 ▼             if access(L,i)<access(L,j):
37                 elementDel=deletePosition(L,i) #Esta función retorna el elemento
que se eliminó
38                 insert(L,elementDel,j)
39

```

Función utilizada de mylinkedlist.py:

```

171 ▼ def deletePosition(L,position):
172     #Elimina un elemento de la lista de acuerdo a su posición
173     #Retorna el elemento eliminado
174     element=access(L,position)
175     current=L.head
176 ▼     if (position==None):
177         return None
178 ▼     if (position==0):
179         L.head=current.nextNode
180 ▼     else:
181 ▼         for i in range (0,position-1):
182             current=current.nextNode
183             current.nextNode=current.nextNode.nextNode
184     return element

```

Ejercicio 2:

	Estabilidad	Posibilidad de trabajo inPlace	Online	Rendimiento de acuerdo al Mejor Caso	Rendimiento de acuerdo al Peor Caso	Rendimiento de acuerdo al Caso Promedio
Bubble Sort	Si	Si	No	$O(n)$	$O(n^2)$	$O(n^2)$
Selection Sort	Si	Si	No	$O(n^2)$	$O(n^2)$	$O(n^2)$
Insertion Sort	No	Si	Si	$O(n)$	$O(n^2)$	$O(n^2)$

Ejercicio 3:

Dada la siguiente lista de elementos aplique el método de ordenamiento que piense que es el más conveniente según su criterio, justifique la respuesta explicando sus conclusiones.

[99, 78, 64, 62, 45, 34, 23, 13, 01]

Podría utilizar cualquiera de los 3 métodos, ya que esta lista es el peor caso y todos tienen una complejidad de $O(n^2)$ en esa situación.