题目地址：<https://leetcode.com/problems/sort-list/>

解题思路：

1. 利用归并排序 (看代码)

2. 爬楼梯排序（C++ 列表排序算法）

解释一下第2种

1. 开一个数组 sortedList []\*ListNode{}，

|  |  |  |
| --- | --- | --- |
| 0 | nil |  |
| 1 | nil |  |
| 2 | nil |  |
| 3 | nil |  |
| … |  |  |

1. 遍历原链表
2. 对sortedList进行从0开始遍历，如果有数据则与之合并，然后遍历sortedList[i+1] …，否则 把当前合并的链表存在sortedList[i], 中转2.
3. 合并所有sortedList中的链表。

举例说明：

Head :

|  |  |
| --- | --- |
| 0 | nil |
| 1 | nil |
| 2 | nil |
| 3 | nil |

遍历-1 ，查看sortedList[0] 没有数据则直接放入

|  |  |
| --- | --- |
| 0 | -1 |
| 1 | nil |
| 2 | nil |
| 3 | nil |

遍历12， 查看sortedList[0] 有数据则合并（-1->12->nil），查看sortedList[1]没有直接放入。

|  |  |
| --- | --- |
| 0 | Nil |
| 1 | -1->12->nil |
| 2 | nil |
| 3 | nil |

遍历10， 查看sortedList[0] 没有数直接放入（10->nil）。

|  |  |
| --- | --- |
| 0 | 10->nil |
| 1 | -1->12->nil |
| 2 | nil |
| 3 | nil |

遍历1，查看sortedList[0] 有数据则合并（1->10->nil），

|  |  |
| --- | --- |
| 0 | 1->10->nil |
| 1 | -1->12->nil |
| 2 | nil |
| 3 | nil |

查看sortedList[1]有数据继续合并（-1->1->10->12->nil）

|  |  |
| --- | --- |
| 0 | nil |
| 1 | -1->1->10->12->nil |
| 2 | nil |
| 3 | nil |

查看sortedList[2]没有数据直接放入（-1->1->10->12->nil）

|  |  |
| --- | --- |
| 0 | nil |
| 1 | Nil |
| 2 | -1->1->10->12->nil |
| 3 | nil |

遍历0 查看sortedList[0] 没有数据则直接放入

|  |  |
| --- | --- |
| 0 | 0->nil |
| 1 | Nil |
| 2 | -1->1->10->12->nil |
| 3 | nil |

最后合并所有链表得到 -1->0->1->10->12->nil .（关于合并链表可以参考 题目 <https://leetcode.com/problems/merge-k-sorted-lists/>

）