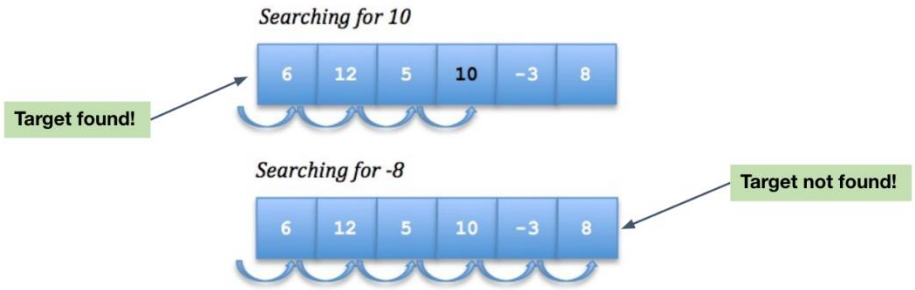
线性搜索 基本概念： 从集合中的第一项开始（例如列表） 每个项目依次与“目标”项目进行比较，直到： 找到“目标”项目；或者 到达集合的末尾（即“目标”项不存在）



Question 1SUBMITTED

Suppose you are doing a sequential search of the list [15, 18, 2, 19, 18, 0, 8, 14, 19, 14]. How many comparisons would you need to do in order to find the key 18?

>>2

Question 2SUBMITTED

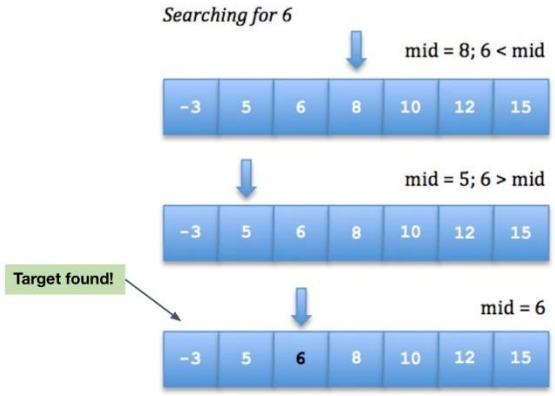
Suppose you are doing a sequential search of the ordered list [3, 5, 6, 8, 11, 12, 14, 15, 17, 18]. How many comparisons would you need to do in order to find the key 13?

>>7

Explanation

Since 14 is greater than the key value 13 you can stop

二进制搜索 基本概念： 首先选择一个将集合（列表）分成两半的项目“中间”项与“目标”项进行比较



Question 3SUBMITTED

Suppose you have the following sorted list [3, 5, 6, 8, 11, 12, 14, 15, 17, 18] and are using the recursive binary search algorithm. Which group of numbers correctly shows the sequence of comparisons used to find the key 8?

>>12,6,11,8

Explanation

Binary search starts at the midpoint and halves the list each time.

Question 4SUBMITTED

Suppose you have the following sorted list [3, 5, 6, 8, 11, 12, 14, 15, 17, 18] and are using the recursive binary search algorithm. Which group of numbers correctly shows the sequence of comparisons used to search for the key 16?

>>12,17,15

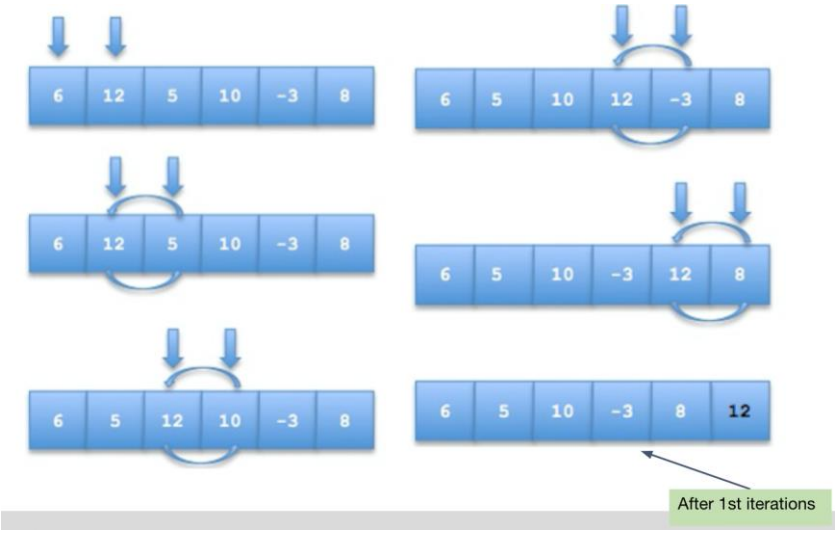
Explanation

A binary search starts at the midpoint and halves the list each time. It is done when the list is empty.

冒泡排序算法。此图说明了冒泡排序的第一次迭代中发生的情况。正在比较这对项目（由两个箭头表示）以确定它们的顺序是否正确。如果出现故障，它们将被交换。

时间复杂度： O(n2)

after first iteration



Question 1SUBMITTED

Suppose you have the following list of numbers to sort: [19, 1, 9, 7, 3, 10, 13, 15, 8, 12] which list represents the partially sorted list after three complete passes of bubble sort?

>>[1、3、7、9、10、8、12、13、15、19]

Question 4SUBMITTED

Consider the following list:

[14,11,43,27,77,41,45,21,70]

If we perform a **Bubble Sort** on this list, what will be the list after the first iteration?

>>

[11、14、27、43、41、45、21、70、77]

Explanation

In the first step: 14 swaps with 11, 43 swaps with 27, 77 swaps with 41, 77 swaps with 45, 77 swaps with 21, 77 swaps with 70.

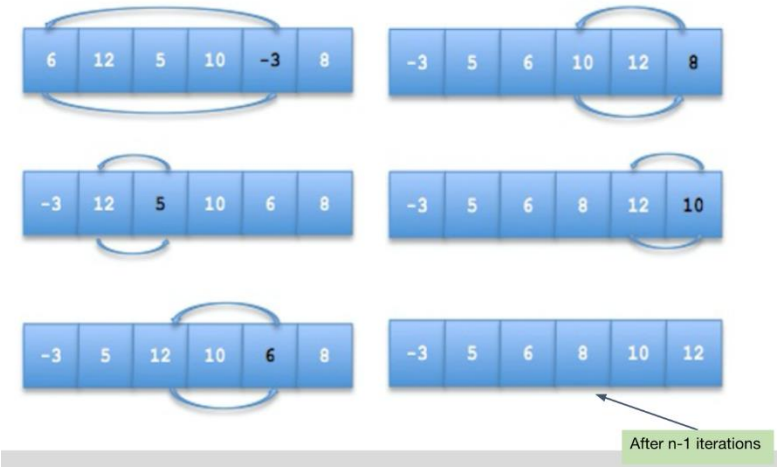
QuestionSAVED

Consider the list **[6, -1, 2, 10, 3, -4, 1]**. Perform bubble sort on this list and write down the list AFTER each element has been placed in its sorted position.

>>

[-1, 2, 6, 3, -4, 1, 10] [-1, 2, 3, -4, 1, 6, 10] [-1, 2, -4, 1, 3, 6, 10]
[-1, -4, 1, 2, 3, 6, 10] [-4, -1, 1, 2, 3, 6, 10] [-4, -1, 1, 2, 3, 6, 10]

选择排序 基本概念： 在每次迭代中查找下一个最小（或最大）的项目 在每次迭代结束时将最小（或最大）的项目放在正确的位置 每次迭代结束时只需要一次交换 集合中的 n 个项目需要 n-1 次迭代



QuestionSUBMITTED

Consider the list **[6, -1, 2, 10, 3, -4, 1]**. Perform selection sort on this list and write down the list AFTER each element has been placed in its sorted position.

>>

[-4,-1,2,10,3,6,1][-4,-1,2,10,3,6,1][-4,-1,1,10,3,6,2][-4,-1,1,2,3,6,10]
[-4,-1,1,2,3,6,10][-4,-1,1,2,3,6,10][-4,-1,1,2,3,6,10]

Question 2SUBMITTED

A list of items **[21, 6, 9, 33, 3]** is given in random order, and we would like to arrange the items using **selection sort.** Please write down the list after **each iteration**. E.g.,

>>

[3, 6, 9, 33, 21] [3, 6, 9, 33, 21] [3, 6, 9, 33, 21] [3, 6, 9, 21, 33]

Question 2SUBMITTED

Suppose you have the following list of numbers to sort: [11, 7, 12, 14, 19, 1, 6, 18, 8, 20] which list represents the partially sorted list after three complete passes of selection sort?

>>[1、6、7、14、19、11、12、18、8、20]

Explanation

Selection sort improves upon bubble sort by making fewer swaps. It is the simplest sorting algorithm that works by repeatedly finding the minimum element (considering ascending order) from the unsorted part and putting it at the beginning.

Question 6SUBMITTED

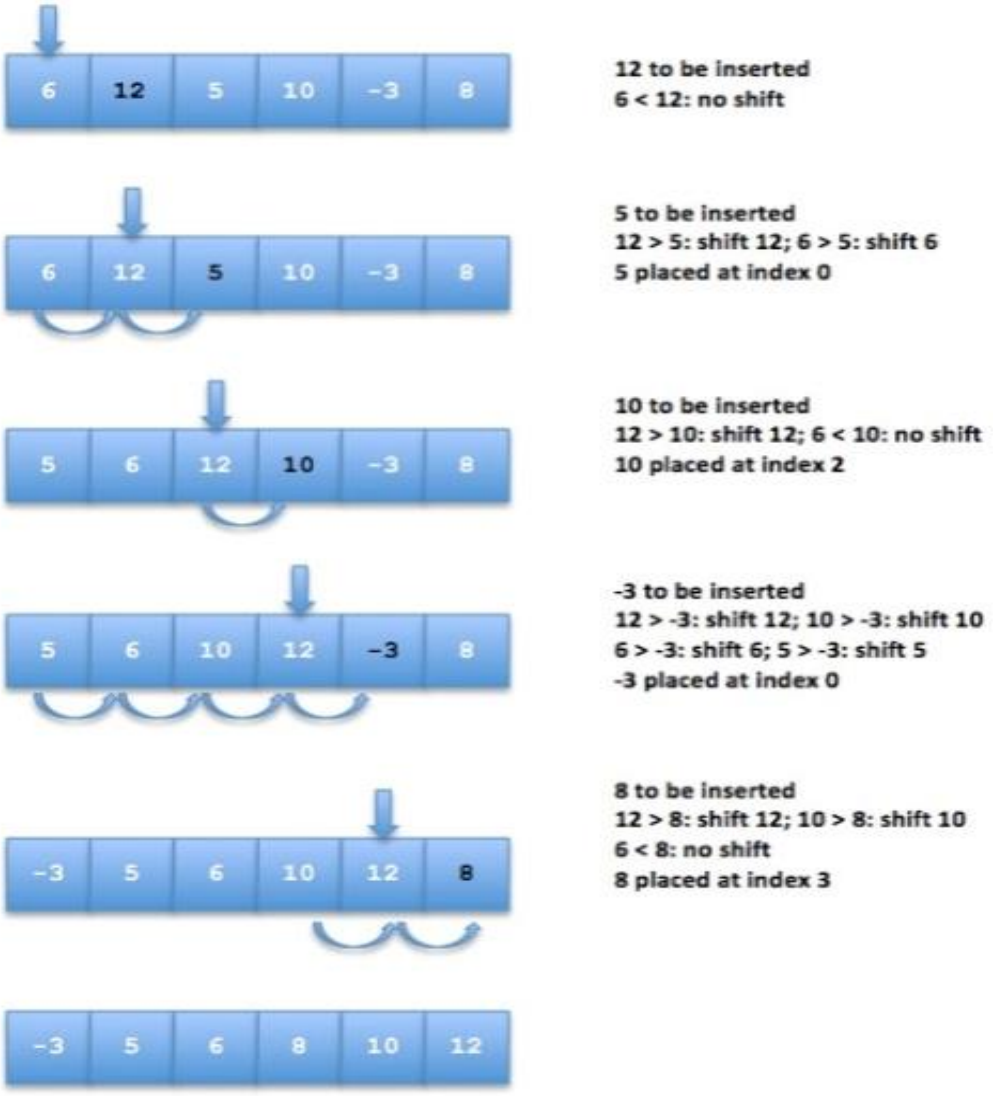
Consider the following list:
[25, 81, 34, 72, 73, 76, 19, 79, 41, 67]
If we perform a **Selection Sort** on this list, what will be the list after the first iteration?

>>[19、81、34、72、73、76、25、79、41、67]
Explanation
25 is swapped with 19 in the first step because 19 is the smallest number.

Question 8SUBMITTED
Consider the following list:
[10,9,8,7,6,5,4,3,2,1]
Which sorting algorithm requires the minimum number of swaps to sort this list in ascending order?

>> **Selection Sort**
Explanation
Only 5 swaps are needed when we use Selection Sort.

插入排序 基本概念： 为要排序的集合维护两个子列表 从“未排序”子列表中挑选每个项目 将这些项目中的每一个插入“排序”子列表中的正确位置 需要移位（重新排序）来为插入腾出“空间”



Question 3SUBMITTED

Suppose you have the following list of numbers to sort: [15, 5, 4, 18, 12, 19, 14, 10, 8, 20] which list represents the partially sorted list after three complete passes of insertion sort?

>> [4、5、15、18、12、19、14、10、8、20]

Explanation

Insertion sort works at the start of the list. Each pass produces a longer sorted list.

Question 5SUBMITTED

Consider the following list:

[11, 14, 27, 43, 41, 45, 21, 70, 77]

If we perform **Insertion Sort** on this list, what will be the list after the first iteration?

>> [11、14、27、43、41、45、21、70、77]

Explanation

In the first step, we pick 14 and insert it into the list before the current index (1). Since 14 is larger than 11, there will be no changes.

Question 7SUBMITTED

Consider the following sorting sequence:

[25, 79, 73, 76, 72, 67, 41, 19, 81, 34]
[25, 79, 73, 76, 72, 67, 41, 19, 81, 34]
[25, 73, 79, 76, 72, 67, 41, 19, 81, 34]
[25, 73, 76, 79, 72, 67, 41, 19, 81, 34]
[25, 72, 73, 76, 79, 67, 41, 19, 81, 34]
[25, 67, 72, 73, 76, 79, 41, 19, 81, 34]
[25, 41, 67, 72, 73, 76, 79, 19, 81, 34]
[19, 25, 41, 67, 72, 73, 76, 79, 81, 34]
[19, 25, 41, 67, 72, 73, 76, 79, 81, 34]
[19, 25, 34, 41, 67, 72, 73, 76, 79, 81]

>> **Insertion Sort**

Explanation

In each step, we insert the value at the current index into the suitable position on the left-hand side sublist and then shift the current index to the right.

QuestionSUBMITTED

Consider the list **[6, -1, 2, 10, 3, -4, 1]**. Perform insertion sort on this list and write down the list AFTER each element has been placed in its sorted position.

>>
[6,-1,2,10,3,-4,1]
[-1,6,2,10,3,-4,1]
[-1,2,6,10,3,-4,1]
[-1,2,6,10,3,-4,1]
[-1,2,3,6,10,-4,1]
[-4,-1,2,3,6,10,1]
[-4,-1,1,2,3,6,10]

Question 3SUBMITTED

A list of items **[21, 6, 9, 33, 3]** is given in random order, and we would like to arrange the items using **insertion sort.** Please write down **the list of each step** in the sorting. E.g.,
first step: []
second step: []

>>
[6, 21, 9, 33, 3]
[6, 9, 21, 33, 3]
[6, 9, 21, 33, 3]
[6, 9, 21, 3, 33]
[6, 9, 3, 21, 33]
[6, 3, 9, 21, 33]
[3, 6, 9, 21, 33]