Improving bubble sort

Task 1 . Reducing the number of comparisons

One improvement that could be made to the bubble sort algorithm is to change the range of the inner loop on line 5 from num\_items - 1 to num\_items - passes.

| 1  2  3  4  ✎5  6  7  8  9  10 | def bubble\_sort(items):  # Initialise the variables  num\_items = len(items)  passes = 1  # Repeat while the maximum numbers of passes has not been made  while passes < num\_items:  # Repeat for the range num\_items - passes  for current in range(num\_items - passes):  # Compare the item at the current position with the next item  if items[current] > items[current+1]:  # Swap the out-of-order items  temp = items[current]  items[current] = items[current+1]  items[current+1] = temp  # Increase the number of passes by 1  passes = passes + 1 |
| --- | --- |

**Figure 2**

**Complete** the table below for tracing the two expressions num\_items - 1 and num\_items - passes when items is a list of **8 items**.

| passes | num\_items - 1 | num\_items - passes |
| --- | --- | --- |
| 1 |  |  |
| 2 |  |  |
| 3 |  |  |
| 4 |  |  |
| 5 |  |  |
| 6 |  |  |
| 7 |  |  |

**Explain** how changing the range of the inner loop to num\_items - passes increases the efficiency of the bubble sort algorithm compared to num\_items - 1.

|  |
| --- |

Task 2 . Stopping when no swaps were made

Now you are going to make a second improvement to the bubble sort algorithm in **Figure 2** by following the instructions below:

* Insert the statements `swapped = False` and `swapped = True` in the algorithm so that `swapped` is reset to False in the beginning of each pass and set to True only when a swap occurs.
* Modify the while condition so that the iteration continues only as long as `swapped` has been set to True in the previous pass, i.e. if at least a pair of elements was swapped.
* Add comments to the code to explain the changes you made.

| 1  2  3  4  5  6  7  8  9  10 | def bubble\_sort(items):  # Initialise the variables  num\_items = len(items)  passes = 1  # Repeat while the maximum numbers of passes has not been made  while passes < num\_items:  # Repeat for each pair of items, reducing the number of  # comparisons by the number of passes that have been completed  for current in range(num\_items - passes):  # Compare the item at the current position with the next item  if items[current] > items[current+1]:  # Swap the out-of-order items  temp = items[current]  items[current] = items[current+1]  items[current+1] = temp  # Increase the number of passes by 1  passes = passes + 1 |
| --- | --- |