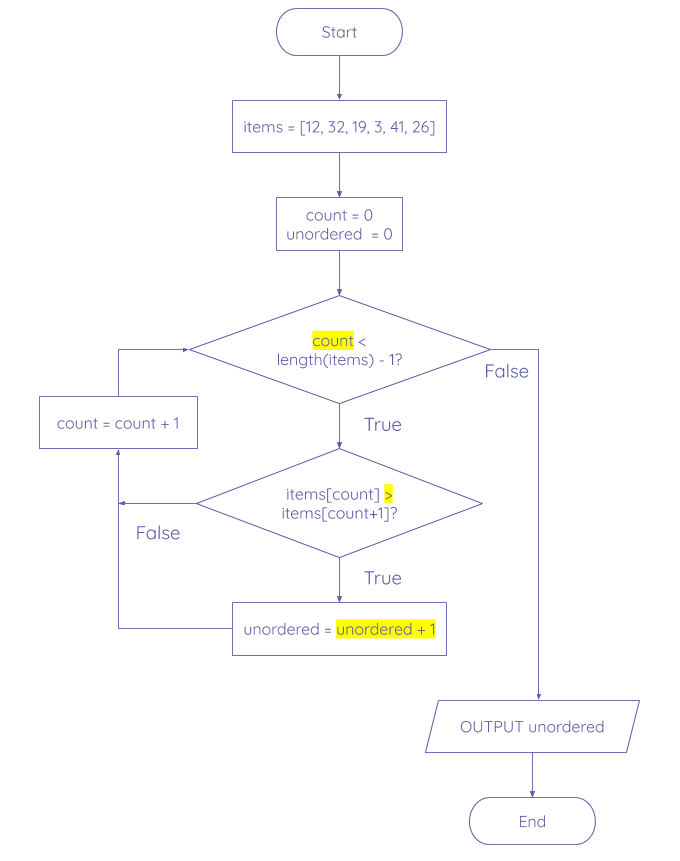
Counting unordered items - Solution

Task 1 . Items that are out-of-order

The purpose of the flowchart is to count the number of pairs of items that are not in the right order. The order that the items should be in is the lowest value to the highest.

Analyse the flowchart and then answer the questions that follow.



What is the missing part of the first condition?

| count < length(items) |
| --- |

Why is the first condition based on length(items) - 1?

| Each adjacent pair of items needs to be compared, so the number of pairs in a list of items is the total number of items - 1. |
| --- |

There are two errors in the flowchart. **Fix** the errors by listing the correct statements below. **State** the type of error it is and **explain** your reasoning for each correction.

| 1. Is items[count] > items[count+1]  Logic error.  Based on what the flowchart executes after the True and False scenarios, this condition should be checking if the item after count is larger than the item at count. Another fix could be to swap the True and False statements around.  2. unordered = unordered + 1  Logic error.  Setting the value or unordered to 1 each time that statement is executed does not record how many pairs of items are out-of-order. Instead, unordered should increment by 1 each time there are a pair of items that are not in the correct order. |
| --- |

Explorer .

Using the below code from the starter activity, create a flowchart for this algorithm that uses more meaningful identifiers than the existing ones.

| 1  2  3  4  5  6  7  8 | def linear\_search(item\_list, search\_item):  index = -1  num = 0  while num < len(item\_list):  if item\_list[num] == search\_item:  index = num  num = num + 1  return index |
| --- | --- |