

Chapter 5.2: Searching and Sorting - Insertion Sort

Refresh

In the last lesson we looked at a Bubble Sort and you might want to refresh your memory by watching [this video](#). We stated that the Bubble Sort is not an efficient algorithm can you remember why?

Objectives

- You should be able to explain how an Insertion Sort works.
- You should be able to write a simple program in Python that performs an Insertion Sort on a simple set of items.

Introduction

An Insertion Sort works by examining each item in turn and inserting it into the right position.

The steps of the Insertion Sort are:

1. IF there is only one item STOP
2. Look at the second item and compare it with the first
3. IF it is greater than the first then leave it in place ELSE IF it is less than swap the two numbers
4. Check the third number. If it is smaller than the second one then compare it with the first one. IF it is smaller than the first, place it in this position by moving the first two number along.
5. REPEAT this procedure with all the numbers by comparing them with the numbers to the left until a smaller number is found. IF a smaller number is found then place the number in the position to the right of it by moving the other numbers along.

WORKED EXAMPLE

Using insertion sort, put this list in ascending order.

6 3 1 2 7 4 5

6	3	1	2	7	4	5	Second number – smaller than first – swap.
3	6	1	2	7	4	5	Third number – smaller than second, smaller than first: move third number to first position and move others to the right.
1	3	6	2	7	4	5	Fourth number – smaller than third, smaller than second, bigger than first: move fourth number to second position and move second and third to the right.
1	2	3	6	7	4	5	Now compare the next number with those to the left. As they are all smaller this number is left in position.
1	2	3	6	7	4	5	Repeat the procedure until the end of the list is reached and the numbers are in the correct order.
1	2	3	4	6	7	5	
1	2	3	4	5	6	7	

Watch [this video](#) to see how the Insertion Sort works.

This [animation](#) goes through the algorithm step by step.

Pseudo Code Examples

Below is one way to program the algorithm in Python. It is just the **pseudo code** and not the actual Python code.

Chapter 5.2: Searching and Sorting - Insertion Sort

```
1  Procedure InsertionSort (List, First, Last)
2      For CurrentPointer <- First + 1 To Last
3          CurrentValue <- List[CurrentPointer]
4          Pointer <- CurrentPointer - 1
5          While List[Pointer] > CurrentValue AND Pointer > 0
6              List[Pointer+1] <- List[Pointer]
7              Pointer <- Pointer - 1
8          EndWhile
9          List[Pointer+1] <- CurrentValue
10     EndFor
11 EndProcedure
```

Task 1

Produce a table showing the steps needed to sort the following numbers into ascending order using the insertion sort method. 20, 15, 3, 13, 9, 2, 6.

Position 1	Position 2	Position 3	Position 4	Position 5	Position 6	Position 7
20	15	3	13	9	2	6
15	20	3	13	9	2	6
3	15	20	13	9	2	6
3	13	15	20	9	2	6
3	9	13	15	20	2	6
2	3	9	13	15	20	6
2	3	6	9	13	15	20

Bold represents the numbers moved into the sorted list.

Task 2

A teacher has sorted the test results for a particular class in an array. Write the code that would sort the list into ascending order using an insertion sort

Link to code or screen shot of annotated code

Insertion sort has a list of sorted and unsorted numbers if there is more than one number in a list. In the code you can see a lot about the position of elements because elements are moved into the sorted list and sorted until eventually everything is sorted.

Chapter 5.2: Searching and Sorting - Insertion Sort

```
def insertionSort(list):  
    for index in range(1, len(list)):  
        currentValue = list[index]  
        position = index  
        while position > 0 and list[position-1] > currentValue:  
            list[position] = list[position-1]  
            position = position - 1  
        list[position] = currentValue  
  
list = [ 20, 15, 3, 13, 9, 2, 6]  
print("Here is the list unsorted:", list)  
insertionSort(list)  
print("Here it is again after using Insertion Sort:", list)  
print("Insertion Sort is faster than Bubble Sort as it only has to pass through the list once  
      unlike Bubble Sort which passes through multiple times.")
```