DSAII: Insertion Sort

Sattiraju Prabhakar
CS560: Class2
Wichita State University

Topics

- Insertion Sort

Insertion Sort

- Addresses the sorting problem:
  - Input: A sequence of numbers <a1, a2, …, an>
  - Output: An ordered sequence of input numbers <a1', a2', …an'> where a1' ≤ a2' ≤ … ≤ an'

Informal Algorithm

- Input is given as an array
- Assume you have another array (call it Temp array), to hold sorted items from Input
- Take the first element from Input, delete it from Input
- Insert it at right place in Temp.
- Repeat it until Input is empty
- Return Temp as Output

In Place Algorithm

- The previous algorithm uses two arrays
- It is useful for understanding the algorithm
- We can do it using a single array
  - We rearrange the elements in a single array without using another array – ***In Place Algorithm***

New Problem Instance

- Input: [9, 2, 6, 1, 8]
- Output: [1, 2, 6, 8, 9]

Let us observe the behavior

- Step1:
  - Input: [9, 2, 6, 1, 8]
  - Key: Nil
  - Temp: []
- Step2:
  - Input: [2, 6, 1, 8]
  - Key: 9
  - Temp: [9]
- Step3:
  - Input: [6,1,8]
  - Key: 2
  - Temp: [2, 9]
- Step4:
  - Input: [1, 8]
  - Key: 6
  - Temp: [2, 6, 9]
- Step5:
  - Input: [8]
  - Key: 1
  - Temp: [1, 2, 6, 9]
- Step6:
  - Input: []
  - Key: 8
  - Temp: [1, 2, 6, 8, 9]

Now you try it!

- Input: [10, 200, 500, 7, 3, 1]

In Place Algorithm

- The previous algorithm uses two arrays
- It is useful for understanding the algorithm
- We can do it using a single array
  - We rearrange the elements in a single array without using another array – ***In Place Algorithm***

Let us consider a modification to algorithm (Informally)

- Input is given as an array
- Instead of using another array (Temp), let us keep the sorted elements in the front of the array.
- Let us use two variables:
  - Position: It is a counter which tells us where the sorted array ends and the unsorted array begins
  - Key: The content of the Position
- Initialize the Position to first element in Input, Key holds the contents of the Position
- Step1: Insert Key in the array beginning in 1 and ending in (Position – 1)
- Increment Position, and Update Key
- If Position has reached end of Input, Stop
- Else Go to step1

Example Behavior

- Step1:Update Position
  - Input: [14, 12, 1, 10, 9]
  - Position: 0
  - Key: 14
- Step2: Insert
  - Input: [14, 12, 1, 10, 9]
  - Position: 0
  - Key: 14
- Step3: Update Position
  - Input: [14, 12, 1, 10, 9]
  - Position: 1
  - Key: 12
- Step4: Insert
  - Input: [12, 14,1, 10, 9]
  - Position: 1
  - Key: 12
- Step5:Update Position
  - Input: [12, 14, 1, 10, 9]
  - Position: 2
  - Key: 1

Example Behavior 2

- Step6: Insert
  - Input: [1, 12, 14, 10, 9]
  - Position: 2
  - Key: 1
- Step7: Update Position
  - Input: [1, 12, 14, 10, 9]
  - Position: 3
  - Key: 10
- Step8: Insert
  - Input:[1, 10, 12, 14, 9]
  - Position: 3
  - Key: 10
- Step9: Update Position
  - Input: :[1, 10, 12, 14, 9]
  - Position: 4
  - Key: 9
- Step10: Insert
  - Input: [1, 9, 10, 12, 14]
  - Position: 4
  - Key: 9
- Step11: Update Position
  - Position: 5
- Step11: Termination
  - Test: Position > length(Input)
  - True → Terminate

Now you try it!

- Input: [123, 76, 890, 21, 54, 12, 100]

Formal Algorithm: *Insertion Sort*

1. **for** j ← 2 to *length*[A]
    1. **do** *key* ← A [j]
        1. ;; Insert A [j] into the sorted sequence A[1...j-1]
        2. i ← j-1
        3. **while** i > 0 and A [i] > *key*
            1. **do** A[i+1] ← A [i]
                1. i ← i -1
        4. A [i + 1] ← *key*

Now try this!

- Input: [4, 3, 7, 5, 6]
- Input: [12, 34, 21, 5, 56, 78, 1, 3]

New Problem

- Rewrite Insertion-Sort procedure to sort into nonincreasing instead of nondecreasing order.
- Then try it on the following Input:
  - [4,3,7,5,6]