17 Feb 2020

Sorting Algorithm

CPE111: Programming with Data Structure

Reference

- Rance D. Necaise. Data Structures and algorithms using python. Chapter2. John Wiley&Sons,Inc., 2011
- Michael T.Goodrich, Roberto Tamassia, Michael H. Goodwasser. Data Structures and Algorithms in python. Chapter5. John Wiley&Sons,Inc. 2013

Sorting is the process of arranging or ordering a collection of items such that each item and its successor satisfy a prescribed relationship.

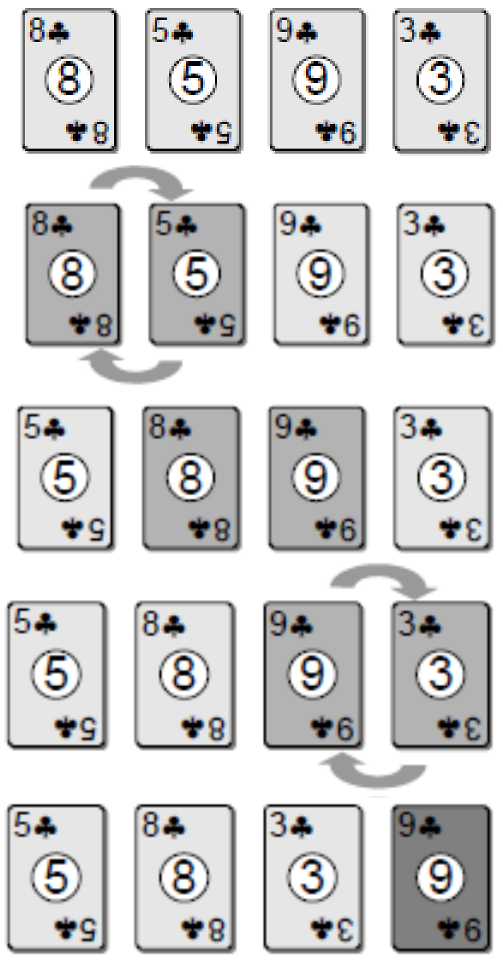
- Bubble Sort
- Selection Sort
- Insertion Sort
- Count Sort
- Heap Sort
- Merge Sort
- Quick Sort
- Radix Sort
- Bucket Sort

Quadratic

Linear

Loglinear

Bubble Sort



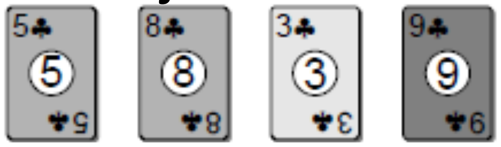
```
# Sorts a sequence in ascending order using the bubble sort algorithm.
def bubbleSort( theSeq ):
    n = len( theSeq ) - 1
    # Perform n-1 bubble operations on the sequence
    for i in range( n , 0 , -1) :
    # Bubble the largest item to the end
        for j in range(i) :
         if theSeq[j] > theSeq[j + 1] : # swap the j and j+1 items
            tmp = theSeq[j]
            theSeq[j] = theSeq[j + 1]
            theSeq[j + 1] = tmp
```

n = 3

i = 3

j = 0..2

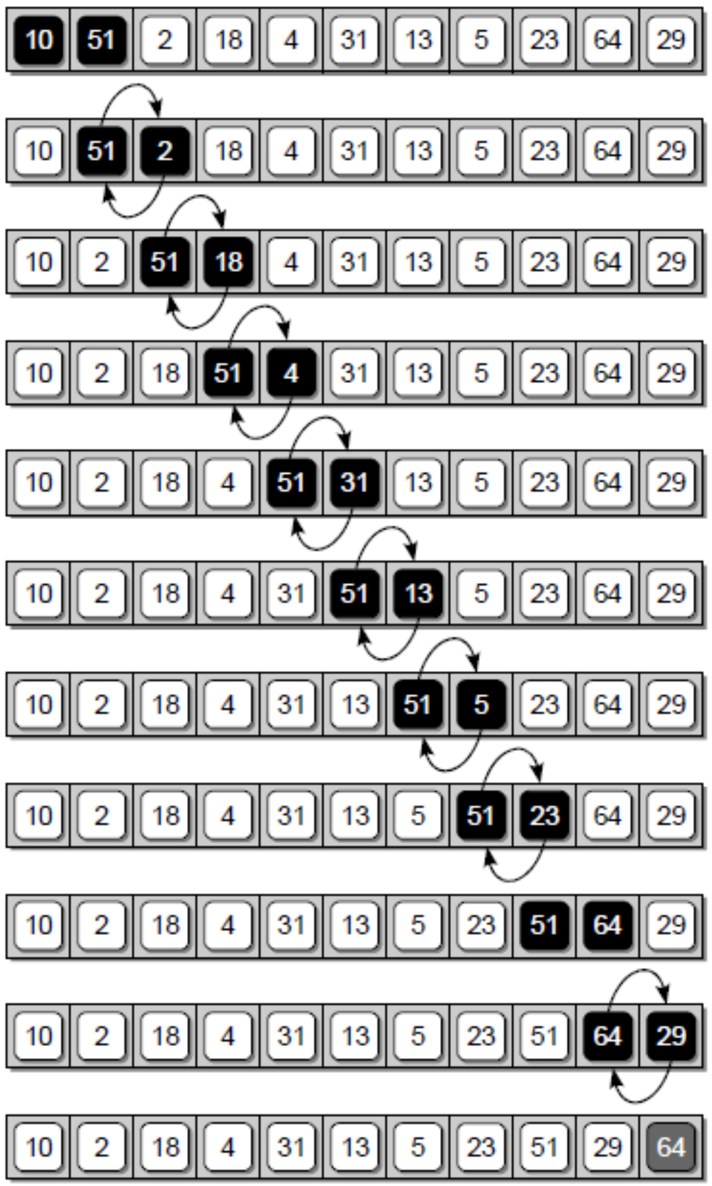
i = 2   j = 0..1



```
# Sorts a sequence in ascending order using the bubble sort algorithm.
def bubbleSort( theSeq ):
        n = len( theSeq ) - 1
        # Perform n-1 bubble operations on the sequence
        for i in range( n , 0 , -1) :
        # Bubble the largest item to the end
                for j in range(i) :
                  if theSeq[j] > theSeq[j + 1] : # swap the j and j+1 items
                          tmp = theSeq[j]
                          theSeq[j] = theSeq[j + 1]
                          theSeq[j + 1] = tmp
```

i = i   j = 0

Bubble sort is considered one of the most inefficient sorting algorithms due to the total number of swaps required.



n = 10, i = 10, j = 0..9

n = 10, i = 9, j = 0..8

n = 10, i = 8, j = 0..7

n = 10, i = 7, j = 0..6

n = 10, i = 6, j = 0..5

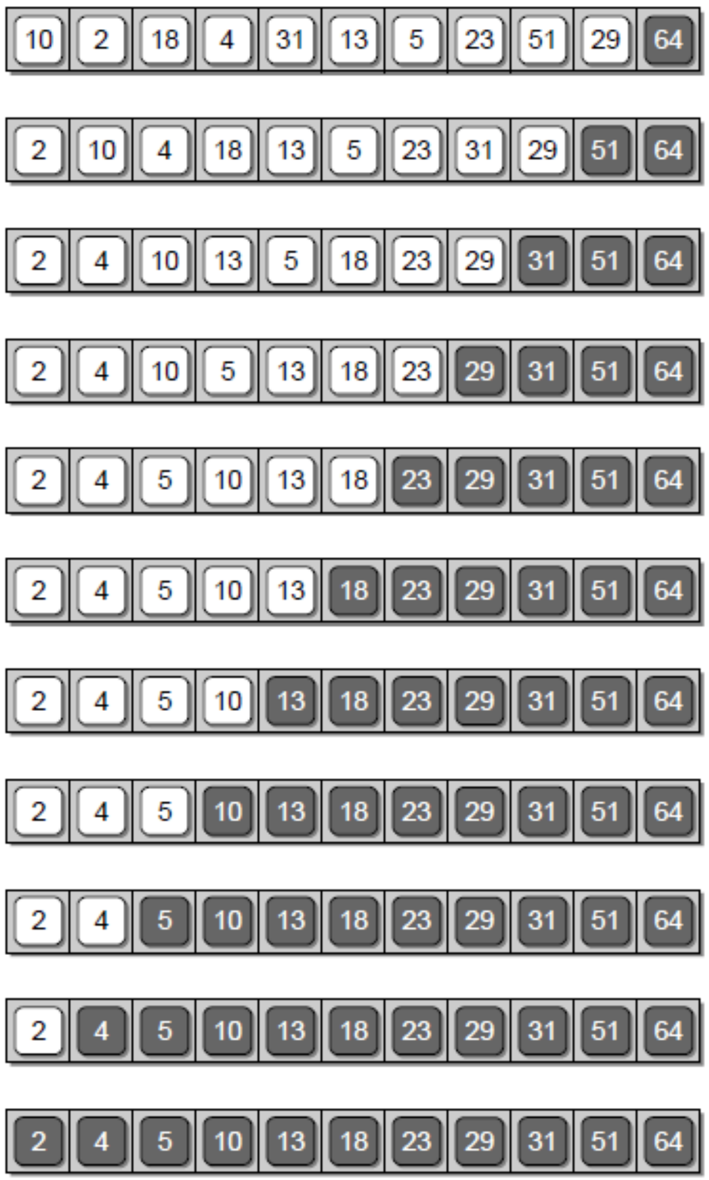
n = 10, i = 5, j = 0..4

n = 10, i = 4, j = 0..3

n = 10, i = 3, j = 0..2

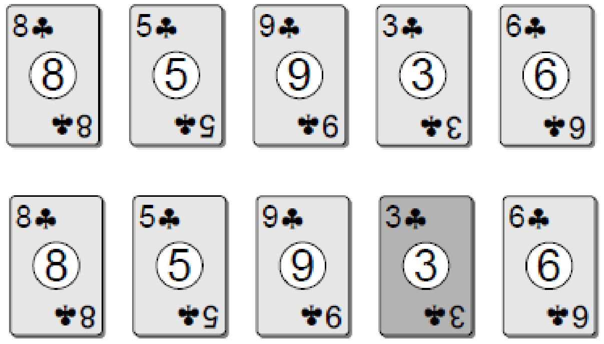
n = 10, i = 2, j = 0..1

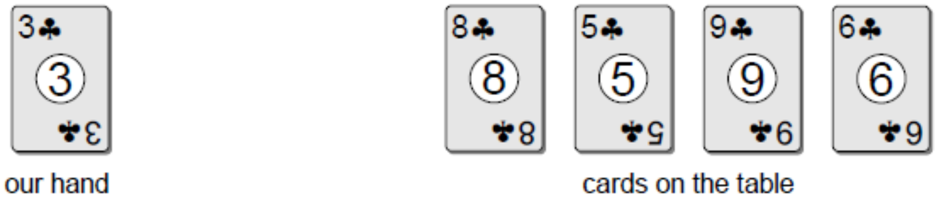
n = 10, i = 1, j = 0

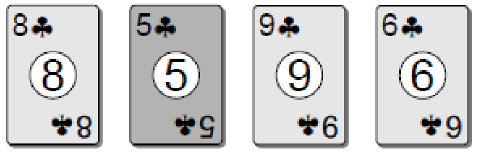


Selection Sort

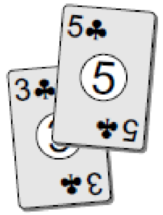
Selection Sort improves on the bubble sort and works in a fashion similar to what a human may use to sort a list of values



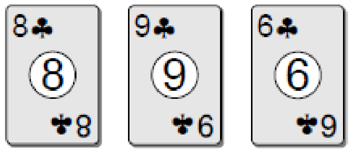




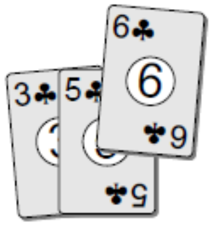
cards on the table



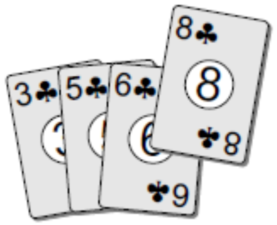
our hand



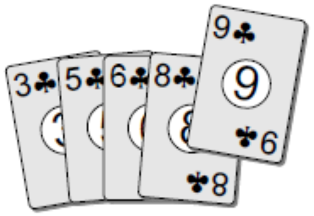
cards on the table



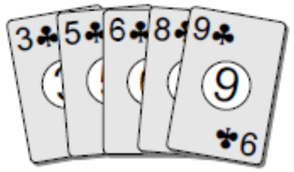
pick up the next smallest card (6)



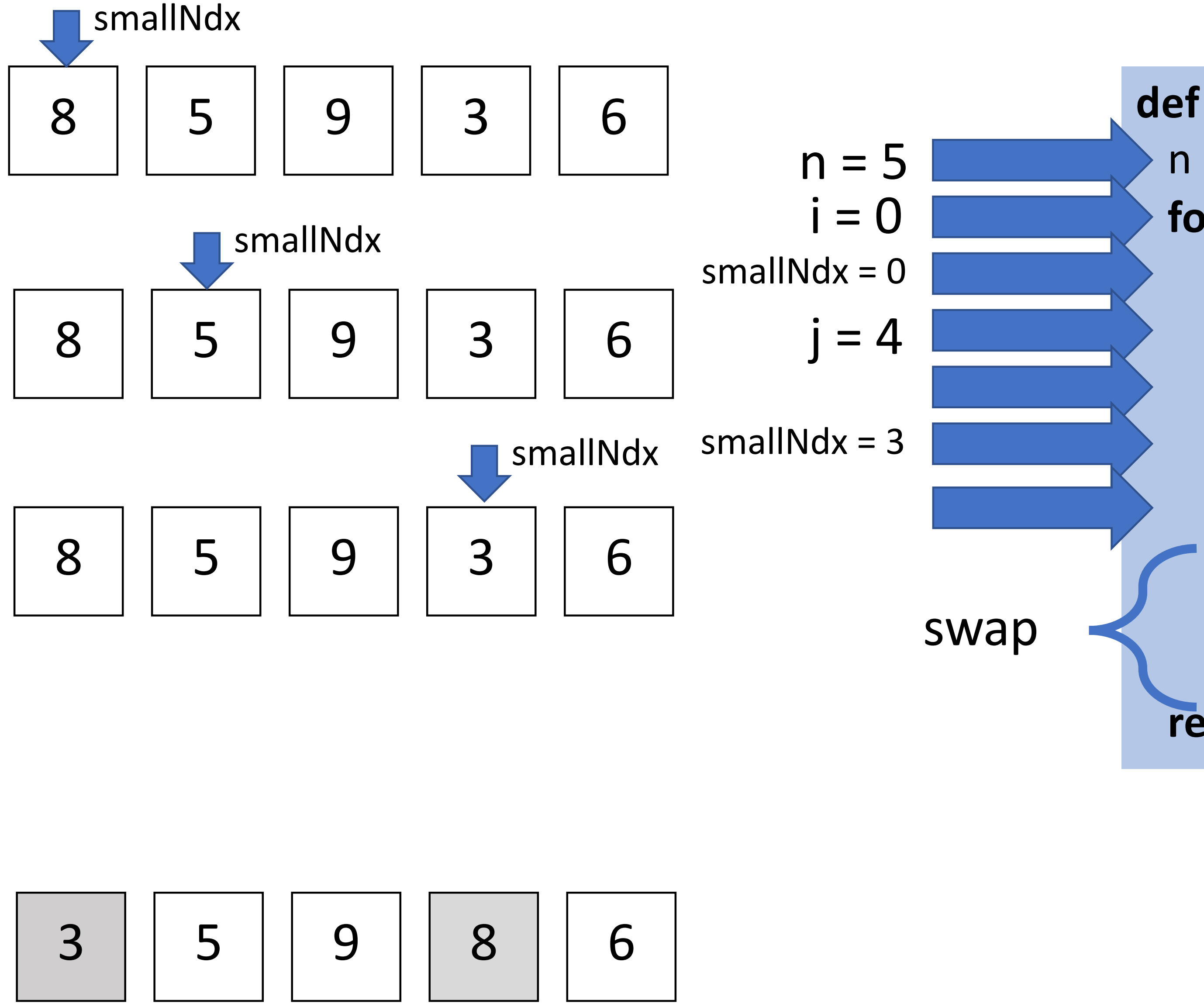
pick up the next smallest card (8)



pickup the last card (9)

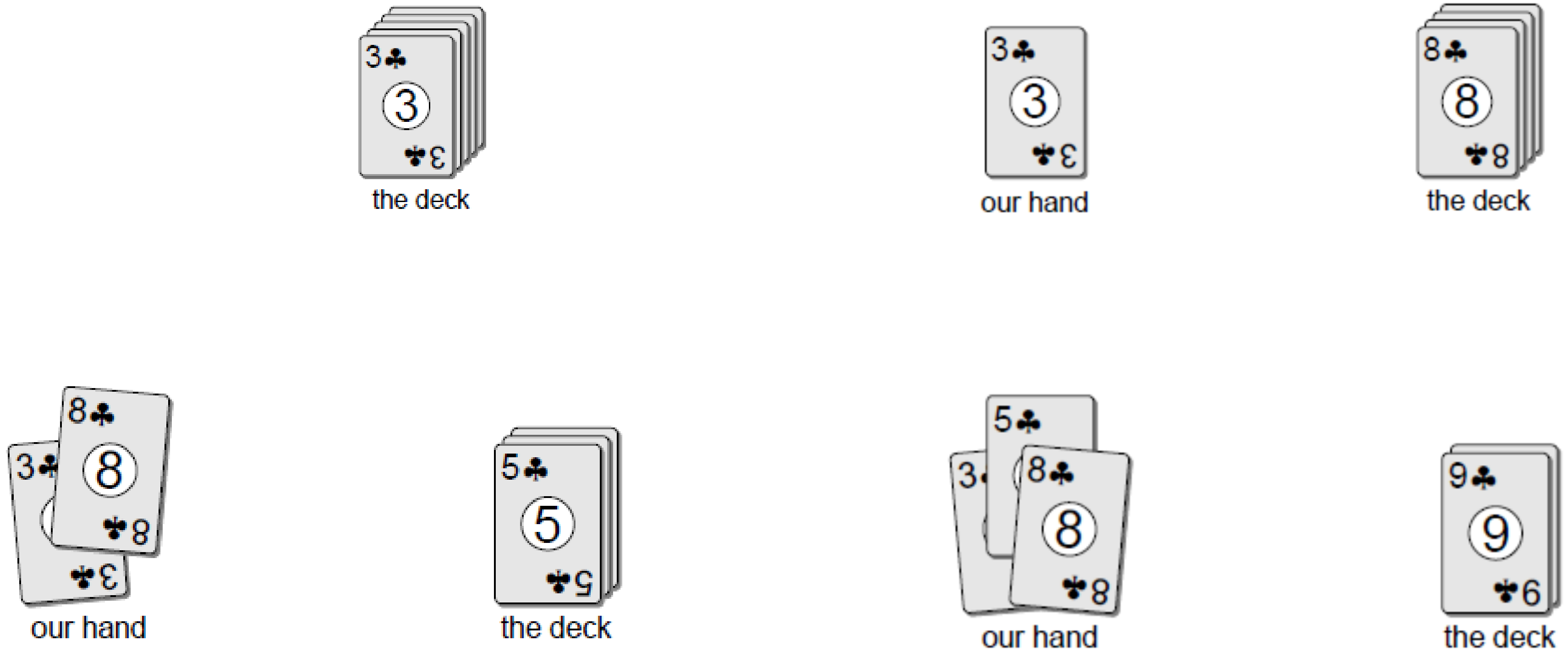


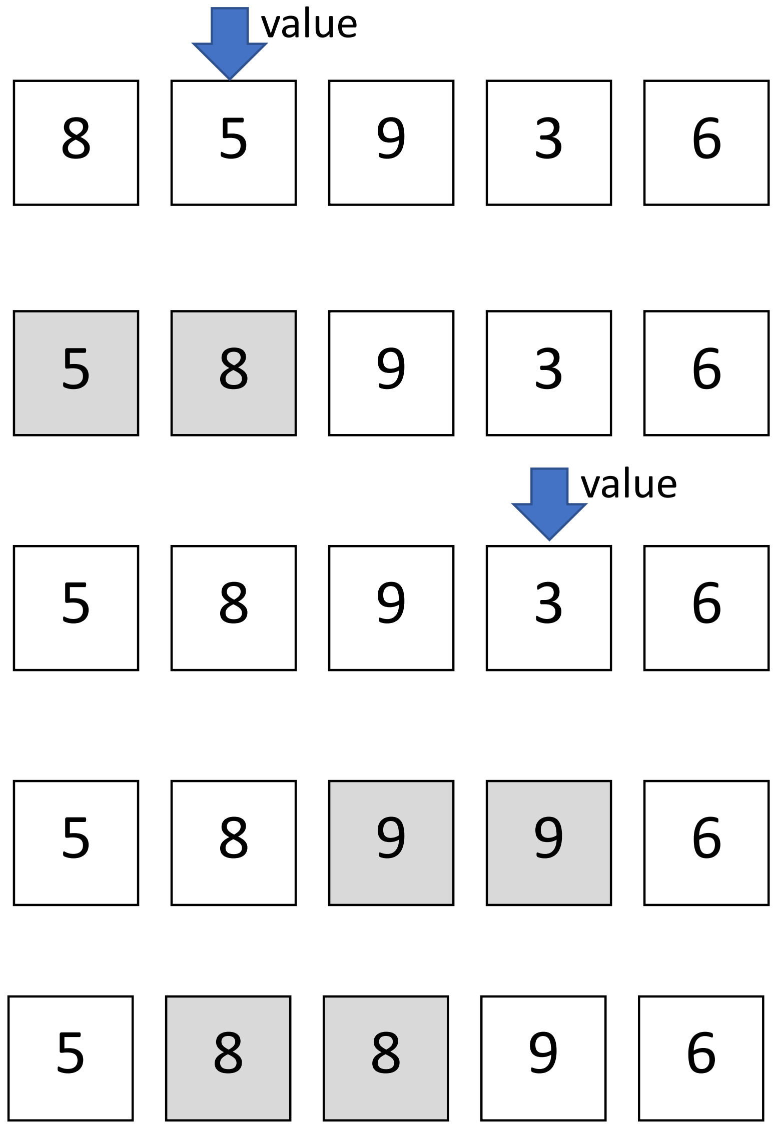
the resulting hand



```
def selectionSort( Seq ):
  n = len( Seq )
  for i in range( n - 1 ):
    smallNdx = i
    for j in range( i + 1, n ):
      if Seq[j] < Seq[smallNdx] :
        smallNdx = j
    if smallNdx != i :
      tmp = Seq[i]
      Seq[i] = Seq[smallNdx]
      Seq[smallNdx] = tmp
  return Seq
```

Insertion Sort

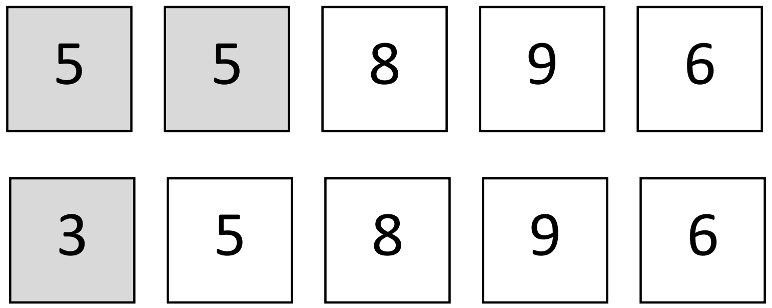


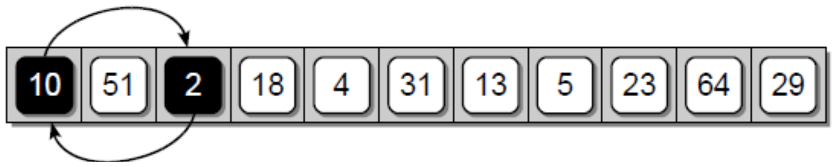


n = 5
i = 3
value = 3
pos = 3

pos = 0

```
def insertionSort(theSeq):
    n = len(theSeq)
    for i in range(1,n):
        value = theSeq[i]
        pos = i
        while pos > 0 and value < theSeq[pos -1]:
            theSeq[pos] = theSeq[pos - 1]
            pos -= 1
        theSeq[pos] = value
    return theSeq
```




10
51
2
18
4
31
13
5
23
64
29