Searching (Linear)


```
def searchLinear(A, n, x):
  for i in range(0, n):
    if (A[i] == x):
      return i
  return -1

A = [77,12,8,39,27,21,44,18,6,47,11,37,60,56]
x = 39

searchResult = searchLinear(A, len(A), x)
if searchResult == -1:
  print ("Not Found")
else:
  print (x, "Found at Location ", searchResult)
```

Searching (Binary)


```
def searchBinary(A, x, start, end):
    while start <= end:
        middle = start + (end - start)//2
        if A[middle] == x:
            return middle
        elif A[middle] < x:
            start = middle + 1
        else:
            end = middle - 1
    return -1

A = [6,8,11,12,18,21,27,37,39,44,47,58,60,77]
x = 39
searchResult = searchBinary(A, x, 0, len(A))
if searchResult == -1:
  print ("Not Found")
else:
  print (x, "Found at Location ", searchResult)
```

Sorting (Bubble Sort)


```
def bubbleSort(A):
    n = len(A)
    flag = False
    for i in range(n-1):
        for j in range(0, n-i-1):
            if A[j] > A[j + 1]:
                flag = True
                A[j], A[j + 1] = A[j + 1], A[j]
        if not flag:
            return
A = [42, 23, 74, 11, 65, 58]
bubbleSort(A)
print(A)
```

Sorting (Selection Sort)


```
def selectionSort(numbers, size):
  for I in range(size):
    iMin = I
    for j in range(I + 1, size):
      if numbers[j] < numbers[iMin]:
        iMin = j
    (numbers[I], numbers[iMin]) = (numbers[iMin], numbers[I])
numbers = [42, 23, 74, 11, 65, 58]
size = len(numbers)
selectionSort(numbers, size)
print(numbers)
```

Sorting (Insertion Sort)


```
def insertionSort(numbers):
  for i in range(1, len(numbers)):
    value = numbers[i]
    hole = i-1
    while hole >=0 and value < numbers[hole] :
        numbers[hole+1] = numbers[hole]
        hole -= 1
    numbers[hole+1] = value
numbers = [42,23,74,11,65,58]
insertionSort(numbers)
print(numbers)
```

Sorting (Quick Sort)


```
def partition(A, start, end):
  pivot = A[end]
  pIndex = start - 1
  for i in range(start, end):
    if A[i] <= pivot:
      pIndex = pIndex + 1
      (A[pIndex], A[i]) = (A[i], A[pIndex])
  (A[pIndex + 1], A[end]) = (A[end], A[pIndex + 1])
  return pIndex + 1

def quickSort(A, start, end):
  if start < end:
    pIndex = partition(A, start, end)
    quickSort(A, start, pIndex - 1)
    quickSort(A, pIndex + 1, end)

numbers = [5,7,10,5,2,9,1,8,6,3]
size = len(numbers)
quickSort(numbers, 0, size - 1)
print(numbers)
```