Sequential Search: Ordered List

In [7]:
```
def seq_search(arr, ele):

    position = 0
    found = False

    while position<len(arr) and not found:

        if arr[position]==ele:
            found=True

        else:
            position+=1

    return found
```

In [8]:
```
arr = [1,2,3,4,5,6]
```

In [9]:
```
seq_search(arr, 3)
```

Out[9]:
```
True
```

In [10]:
```
seq_search(arr, 9)
```

Out[10]:
```
False
```

In [ ]:

In [11]:
```
def ordered_seq_search(arr, ele):

    position = 0
    found = False
    stopped = False

    while position<len(arr) and not found and not stopped:

        if arr[position]==ele:
            found=True

        else:

            if arr[position]>ele:
                stopped = True
            else:
                position+=1

    return found
```

In [12]:
```
arr1 = [1,2,3,4,5,6]
ordered_seq_search(arr1,5)
```

Out[12]:
```
True
```

```
In [14]: ordered_seq_search(arr1,7)

Out[14]: False
```

Binary Search:

```
In [10]: def binary_search(arr,ele):

             first = 0
             last = len(arr)-1
             found = False
             length = first+last

             while first<=last and not found:

                 if length%2 == 0:
                     mid = (first+last)/2
                 else:
                     mid = int((first+last)/2)

                 if arr[mid]== ele:
                     found = True
                 else:
                     if ele<arr[mid]:
                         last = mid-1
                     else:
                         first = mid+1

             return found
```

```
In [11]: arr = [1,2,3,4,5,6,7,8,9,10]
```

```
In [13]: binary_search(arr,11)

Out[13]: False
```

Recursive Binary Search:

```
In [20]: def rec_binary_search(arr,ele):

             first = 0
             last = len(arr)-1
             length = first + last
             found = False

             while first<=last and not found:

                 mid = int((first+last)/2)

                 if ele == arr[mid]:
                     found = True

                 else:
                     if ele<arr[mid]:
                         last = mid-1
                         return rec_binary_search(arr[:(last+1)],ele)
```

```
            else:
                first = mid+1
                return rec_binary_search(arr[first:],ele)
    return found
```

In [21]:
```
arr2 = [1,2,3,4,5,6,7,8,9,10]
```

In [22]:
```
rec_binary_search(arr2,10)
```

Out[22]:
```
True
```

In [24]:
```
rec_binary_search(arr2,9)
```

Out[24]:
```
True
```

In [25]:
```
rec_binary_search(arr2,19)
```

Out[25]:
```
False
```

Hashing:

In [ ]: