Computer Programming I

Sorting Algorithms, Part 1

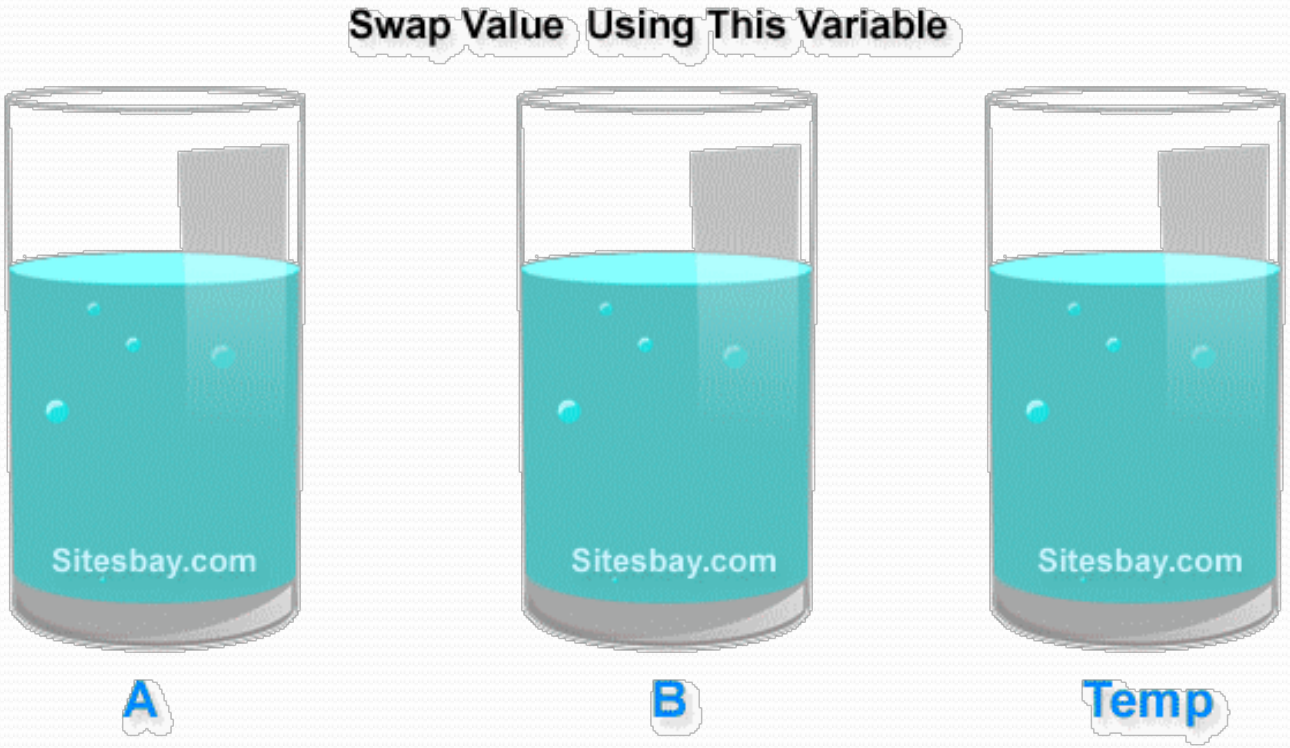
Sorting

- Sorting is the process of arranging a list of items into a well-defined order.
  - Alphabetical order
  - Ascending numeric order
  - Etc.
- Many different sorting algorithms exist:
  - Bubble Sort
  - Selection Sort
  - Insertion Sort
  - Merge Sort
  - Quick Sort
  - Radix Sort
  - Etc.

Sorting

- Algorithms tradeoff performance for simplicity.
- The challenge of sorting is that a program can't "see" the entire list to know where to move an element.
- Instead, a program is limited to simpler steps, typically observing or swapping just two elements at a time.
- So sorting just by swapping values is an important part of sorting algorithms

Swapping values



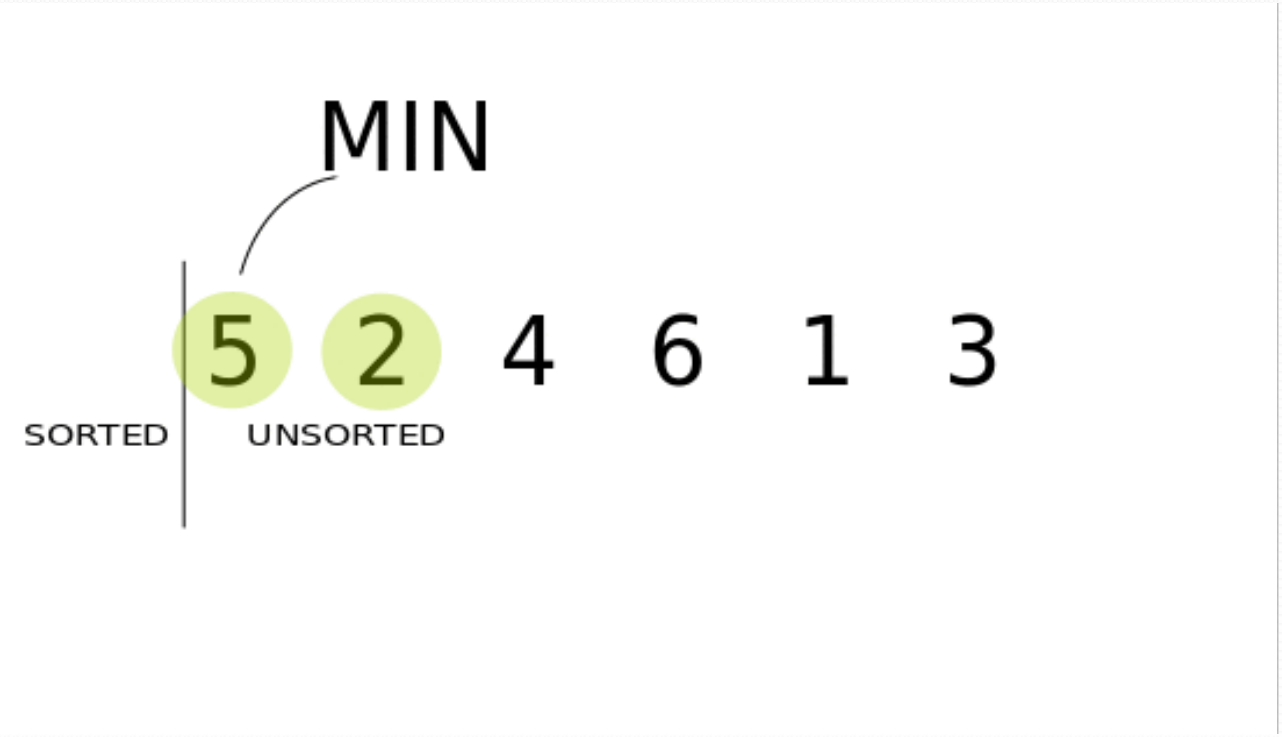
A swapping function

```
#values is a list
#i and j are the indices of the items to be
#swapped
def swap(values, i, j):
    temp = values[i]
    values[i] = values[j]
    values[j] = temp
```

Selection Sort

- One of the most intuitive approaches is selection sort
- Algorithm to sort in ascending order:
  - find the smallest value in the list
  - switch it with the value in the first position
  - find the next smallest value in the list
  - switch it with the value in the second position
  - repeat until all values are in their proper places
- https://www.youtube.com/watch?v=LriMvv9qDrk

Selection Sort Example



Selection Sort Example

5	2	4	6	1	3	original
1	2	4	6	5	3	After 1st pass
1	2	4	6	5	3	After 2nd pass
1	2	3	6	5	4	After 3rd pass
1	2	3	4	5	6	After 4th pass
1	2	3	4	5	6	After 5th pass
1	2	3	4	5	6	Sorted

Selection Sort Implementation

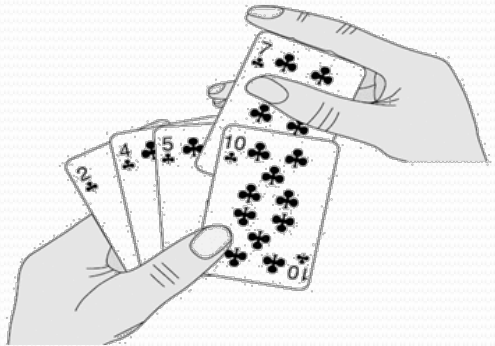
```
def selection_sort(values):
    for i in range(len(values) - 1):
        # Find index of smallest remaining element
        index_smallest = i
        for j in range(i + 1, len(values)):
            if values[j] < values[index_smallest]:
                index_smallest = j

        # Swap values[i] and values[index_smallest]
        swap(values, i, index_smallest)
```

Insertion Sort Algorithm

- Consider the first item to be a sorted sub-list (of one item)
- Insert the second element, by exchanging if necessary:
  - So the first and second element are in the proper place
- Insert the 3rd element in the appropriate position relative to the first two.
- Insert, the 4th element in the appropriate position relative to the first three.
- and so on…
- Similar to sorting your hand playing cards
- https://www.youtube.com/watch?v=ROalU379l3U&feature=youtu.be

Insertion Sort Example



Insertion Sort Example

5	2	4	6	1	3	original
2	5	4	6	1	3	After 1st pass
2	4	5	6	1	3	After 2nd pass
2	4	5	6	1	3	After 3rd pass
1	2	4	5	6	3	After 4th pass
1	2	3	4	5	6	Sorted

Insertion Sort Implementation

```
def insertion_sort(values):
    for i in range(1, len(values)):
        j = i
        # Insert values[i] into sorted part
        # stopping once values[i] in correct position
        while j > 0 and values[j] < values[j - 1]:
            # Swap values[j] and values[j - 1]
            swap(values, j, j - 1)
            j = j - 1
```