COMP 4030/6030: Assignment 1
Due date: 09/08/2015

1. Algorithm *SlowSort* sorts an array A from smallest to largest. Give/describe the invariant that remains true after each outer loop (with index i). This invariant should show why the algorithm is correct at the end.

```
1: SlowSort(A)
2:     for i = 0 to len(A)-1 do
3:         m ← i
4:         for j = i + 1 to len(A)-1 do
5:             if A[j] < m then
6:                 m ← j
7:         Swap(A[i], A[m])
```

2. Algorithm *Process* takes a list A and returns another list. A few notations:

   - An empty list is represented by [ ].
   - $A.pop(0)$ removes and returns the first element of A.
   - As in Python, adding two lists (e.g. $x + y$) results in the concatenation of the two lists.

```
1: Process(A)
2:     if len(A) == 0 then
3:         return [ ]
4:     else
5:         first = A.pop(0)
6:         return Process(A) + [first]
```

   Questions/problems:

   (a) Trace algorithm *Process* on input $A = [1, 2, 3, 4]$. Show the intermediate steps.

   (b) Given any given list A, what does algorithm *Process* return?

   (c) Use mathematical induction to explain why algorithm *Process* correctly does what it is supposed to do.

Notes

The abstract framework of mathematical induction consists of two main steps:

1. Show the algorithm works correctly for the smallest input sizes.
2. Assuming the algorithm works correctly for input sizes $0, 1, 2, \cdots, k$, show the algorithm works correctly for input size $k + 1$. This means you can assume recursive calls are correct, if the recursive calls have smaller input sizes.