COMP108 Algorithmic Foundations
Tutorial 5 (Suggested Solution and Feedback)
w/c 6th March 2017

1. Note that we are swapping TWO entries in the array. The way to call this method is to provide three paramesters: the name of the array (without square bracket [ ]), two indices/positions of the array where we want to swap the content.

```
// swap array[x] and array[y]
public void swap(int[] array, int x, int y) {
    int tmp;
    tmp = array[x];
    array[x] = array[y];
    array[y] = tmp;
}
```

For bubble sort, we need to use a nested loop.

The outer loop governs where the largest number should be moved to, when i is $count - 1$, the largest number should be moved to $data2[count - 1]$, when i is $count - 2$, the second largest number should be moved to $data2[count - 2]$, and so on.

For the inner loop, we want to go through the array from position 0 up to position i whatever i is for that iteration, and each time compare two adjacent values $data2[j]$ and $data2[j + 1]$.

```
// bubble sort ascendingly, using swap()
public void bsort() {
   int[] data2 = new int[count];  // data2.length = count

   copy(data, 0, data2, 0, count);  // copy data[] to data2[]

   // bubble sort on data2[], using swap()
   for (int i=count-1; i>0; i--) {
       for (int j=0; j < i; j++) {
           if (data2[j] > data2[j+1])
               swap(data2, j, j+1);
       }
   }
   printAll(LONG, data2, count);
}
```

For selection sort, we also need to use a nested loop.

The outer loop iterates to find the $(i+1)$-th smallest number and moves it to position i.

The inner loop iterates the array from $data2[i]$ to $data2[count-1]$ and finds the smallest number in this range.

```
// selection sort ascendingly, using swap()
public void ssort() {
   int minloc;
   int[] data2 = new int[count];  // data2.length = count

   copy(data, 0, data2, 0, count);  // copy data[] to data2[]

   // selection sort on data2[], using swap()
   for (int i=0; i<count-1; i++) {
       minloc = i;
       for (int j=i+1; j<count; j++) {
           if (data2[j] < data2[minloc])
              minloc = j;
       }
       swap(data2, i, minloc);
   }
   printAll(LONG, data2, count);
}
```