Some Experiments with Slow Sorting

Here is a Mathematica implementation of Insertion Sort.

```
InsertionSort[Y_List] :=
 Module[{i, j, key, X = Y}, Do[(key = X[[i]]; j = i - 1; While[(j > 0) && (X[[j]] > key),
     (X[[j + 1]] = X[[j]]; j--)]; X[[j + 1]] = key), {i, 2, Length[X]}]; X]
```

Here is a Mathematica implementation of Selection Sort.

```
Unprotect[SelectionSort]
```

```
{SelectionSort}
```

```
SelectionSort[Y_List] := Module[{i, minIndex, X = Y, j},
  Do[(minIndex = i; Do[If[X[[j]] < X[[minIndex]], minIndex = j], {j, i + 1, Length[X]}];
    {X[[i]], X[[minIndex]]} = {X[[minIndex]], X[[i]]}), {i, Length[X] - 1}]; X]
```

Here random permutations of sizes 100, 200, 300,..., 1000 are constructed.

```
rps = Table[RandomPermutation[i], {i, 100, 1000, 100}];
```

Here are the running times (in seconds) of InsertionSort on the 10 permutations constructed above. It is quite slow −roughly 23 seconds for size−1000permutation.

```
ist = Table[ Timing[InsertionSort[ rps[[i]] ];][[1, 1]], {i, 10}]
```

```
{0.2, 0.87, 2.14, 3.54, 5.5, 8.36, 10.77, 15.11, 18.79, 23.}
```

Here are the running times (in seconds) of SelectionSort on the 10 permutations constructed above. It is also quite slow, however, as expected SelectionSort is a little faster than insertion sort.

```
sst = Table[ Timing[SelectionSort[ rps[[i]] ];][[1, 1]], {i, 10}]
```

```
{0.17, 0.67, 1.52, 2.73, 4.34, 6.26, 8.57, 11.23, 14.2, 17.47}
```

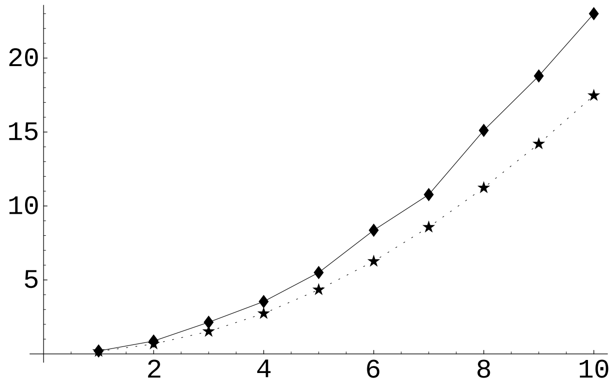
Here is further confirmation of the slowness of these functions. The in−builtsort function in Mathematica is so fast that even for a permutation of size 1000, its running time does not register!

```
st = Table[ Timing[Sort[ rps[[i]] ];][[1, 1]], {i, 10}]
```

```
{0., 0., 0., 0., 0., 0., 0., 0., 0., 0.}
```

The running times of the two functions, InsertionSort and SelectionSort are plotted. It is clear that the running times as a function of the sizes of the permutations are"super−linear"−that is, they grow at a rate that is faster than a linear function. In fact, the plots seem to indicate that these are growing quadratically.

```
MultipleListPlot[ist, sst, PlotJoined → True]
```




```
- Graphics -
```

In fact, further confirmation of this fact comes by computating the ratio: running time on a size−npermutation/ n^2. In both cases, the ratio seems to be a constant, telling us that the
running times are a constant times n^2, where n is the size of the array being sorted.

```
Table[ist[[i]] / (100 i) ^2, {i, 10}]
```

```
{0.00002, 0.00002175, 0.0000237778, 0.000022125, 0.000022,
 0.0000232222, 0.0000219796, 0.0000236094, 0.0000231975, 0.000023}
```

```
Table[sst[[i]] / (100 i) ^2, {i, 10}]
```

```
{0.000017, 0.00001675, 0.0000168889, 0.0000170625, 0.00001736,
 0.0000173889, 0.0000174898, 0.0000175469, 0.0000175309, 0.00001747}
```