

Selection Sort

SELECTION SORT is a simple sorting algorithm, based on a subprocess (called a *pass*) which **selects** the leading element of a list. The method used to select the leader is to run through the list, keeping track (with a pointer) of the earliest element found so far (called the temporary leader). The following is a brief description.

The initial (temporary) leader is compared with the second element and the earlier one becomes the new temporary leader. No swapping takes place, but the new temporary leader can be marked with a pointer. The new temporary leader is compared with the third element and again the earlier one becomes the temporary leader. The process is continued until the final element is reached, when the final temporary leader is known to be the overall earliest element. It is then swapped with the initial temporary leader.

To obtain List 2 from List 1 a pass is applied to List 1 to select a new leader. The first element of List 2 is therefore in its correct place. The process is then applied to the remaining elements of List 2 to select their new leader. The first element of List 2 is left unchanged, so the new leader selected at Pass 2 becomes the second element of List 3. The process is then repeated until the list is completely sorted.

The following example illustrates the first few lists when the process is used to sort List 1 into alphabetical order. The questions following the lists should make the method clear.

<u>List 1</u>	<u>List 2</u>	<u>List 3</u>	<u>List 4</u>
NSW	<u>ACT</u>	ACT	ACT
TAS	TAS	<u>NSW</u>	NSW
ACT	NSW	TAS	<u>NT</u>
QLD	QLD	QLD	QLD
VIC	VIC	VIC	VIC
SA	SA	SA	SA
NT	NT	NT	TAS
WA	WA	WA	WA

- What is the initial temporary leader of List 1?
 - Which element is it compared with?
 - What is the temporary leader after one comparison?
 - Which element is it compared with?
 - What is the temporary leader after two comparisons?
 - Which element is it compared with?
 - What is the temporary leader after three comparisons?
 - Which element is it compared with?
 - How many comparisons are needed to determine the earliest element in List 1?
 - What is the final temporary leader in List 1?
 - Which two elements are swapped to create List 2?
- Which element of List 2 is not involved in Pass 2?
 - What is the initial temporary leader in Pass 2?
 - Which element is it compared with?
 - How many comparisons are needed in Pass 2?
 - How many comparisons are needed in Pass 3?
 - How many comparisons are needed for the entire sorting process? (Give your answer as a sum.)
- Write down List 5.

4. (a) If $S(n)$ is the number of comparisons needed to sort the list:

$$n, n-1, n-2, \dots, 3, 2, 1$$

into numerical order using selection sort, write down a formula for $S(n)$ as a sum.

- (b) Use the result of Sorting Practice Class 1 to write down a formula for $S(n)$.
If you have forgotten it, the result used the following trick:

$$\begin{array}{r} S(n) = 1 + 2 + \dots + (n-2) + (n-1) \\ S(n) = (n-1) + (n-2) + \dots + 2 + 1 \\ \hline 2S(n) = n + n + \dots + n + n \end{array}$$

- (c) Which Big O class is $S(n)$ in?
5. (a) What is the number of comparisons needed to sort out the list $1, 2, \dots, n$, which is already in the correct order?
- (b) Is there any list which takes fewer than $\frac{1}{2}n(n-1)$ comparisons?
- (c) Is there any list which takes more than $\frac{1}{2}n(n-1)$ comparisons?
- (d) Is this a good property for a sorting algorithm?
6. (a) Calculate $S(n) - S(n-1)$.
- (b) What does $S(1)$ equal?

Comment: Using the theory of difference equations it is possible give an alternative derivation of the formula for $S(n)$.