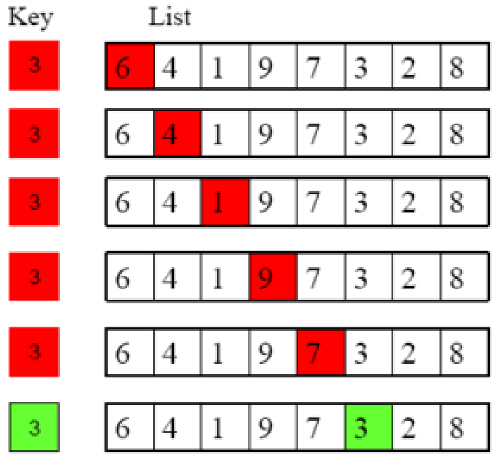
Computing GCSE – 2.1a

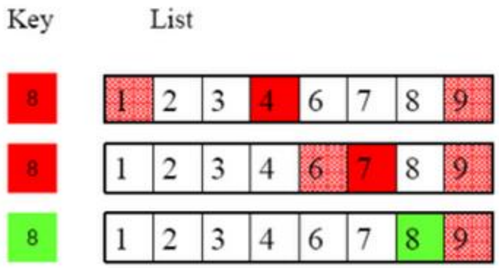
J276/02 – Algorithms: Searching and Sorting

KEY VOCABULARY	
Algorithm	An abstracted program which completes a given task, whatever the data provided
Search	Searching is looking through data, making comparisons with a search term, until the algorithm either finds the data, or identifies that it is not present.
Sort	Putting given sets of data into specified order – usually ascending (alphabetical) or descending (reverse alphabetical)
Linear Search	A type of search where the computer checks every variable, in order, until it finds the search term. Potentially very slow.
Binary Search	A search type based on repeatedly halving the searchable data, until the search term is found
Bubble Sort	A method of sorting data which looks at pairs of variable, and swaps them around if out of order. This continues until there are no more swaps to be made
Merge Sort	Splits the data into increasingly small segments, until single data points are reached, then reassembles the data structure one item at a time.
Insertion Sort	Checks through the data until finding the first incorrectly places item. The algorithm then checks all the previous places to see where the data fits, before inserting it into this slot.

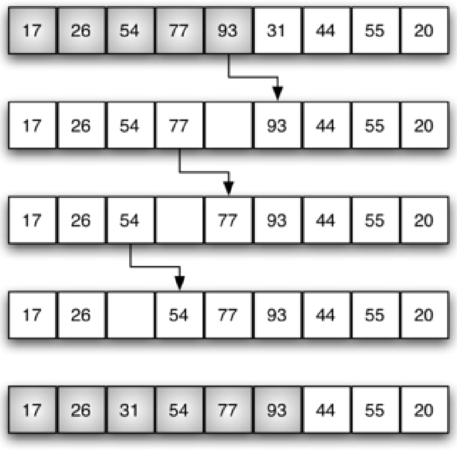
LINEAR SEARCH



BINARY SEARCH



INSERTION SORT



Need to insert 31 back into the sorted list

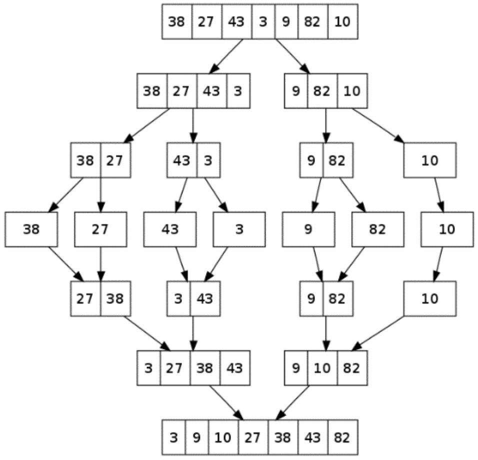
93>31 so shift it to the right

77>31 so shift it to the right

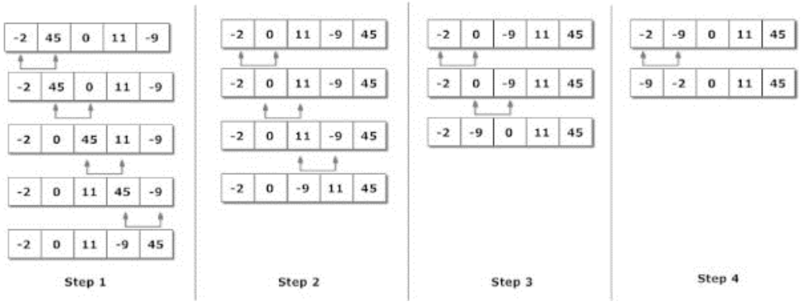
54>31 so shift it to the right

26<31 so insert 31 in this position

MERGE SORT



BUBBLE SORT



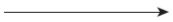
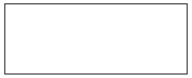
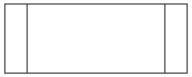
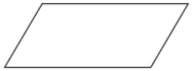
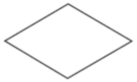
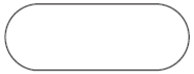
Computing GCSE – 2.1b

J276/02 – Algorithms: Pseudo Code & Flow Charts

KEY VOCABULARY	
Algorithm	An abstracted program which completes a given task, whatever the data provided
Abstraction	Abstraction is moving a problem out of the specific in order to create a general solution that would work in similar scenarios. Ignoring the gritty details to focus on the problem
Decomposition	Breaking a problem down into smaller, computational solvable chunks
Pseudo Code	A structured way of planning code, which is ‘computational’ in style (uses Boolean logic, variables, comparisons and arithmetic for example) but is not tied to a strict high-level language’s syntax
Flow Diagram	A diagram, made using specific shaped boxes, that mocks up the flow of a program through various stages, processes and decisions.
Program Control	Using Boolean logic to guide the computer through a program based on decisions
Comparison Operators	The symbols used to look at a variable or piece of data in relation to is similarity to another piece of data or variable
Arithmetic Operators	The symbols used to show the mathematics to be carried out on a piece of data

Flow charts

Flow charts like pseudocode are informal but the most common flow chart shapes are:

	**Line**	An arrow represents control passing between the connected shapes.
	**Process**	This shape represents something being performed or done.
	**Sub Routine**	This shape represents a subroutine call that will relate to a separate, non-linked flow chart
	**Input/Output**	This shape represents the input or output of something into or out of the flow chart.
	**Decision**	This shape represents a decision (Yes/No or True/False) that results in two lines representing the different possible outcomes.
	**Terminal**	This shape represents the “Start” and “End” of the process.

Comparison operators

==	Equal to
!=	Not equal to
<	Less than
<=	Less than or equal to
>	Greater than
>=	Greater than or equal to

Arithmetic operators

+	Addition e.g. x=6+5 gives 11
–	Subtraction e.g. x=6–5 gives 1
*	Multiplication e.g. x=12*2 gives 24
/	Division e.g. x=12/2 gives 6
MOD	Modulus e.g. 12MOD5 gives 2
DIV	Quotient e.g. 17DIV5 gives 3
^	Exponentiation e.g. 3^4 gives 81