19. Computational thinking and Problem-solving

Starter Quiz ·

A-Level Computer Science Name: _____ Score: _____

1. A linear search:

   ☐ checks each element in turn and works on any list

   ☐ always finds the target in one step

   ☐ requires the data to be sorted

   ☐ only works on numbers

2. Bubble sort works by:

   ☐ repeatedly swapping adjacent out-of-order pairs

   ☐ building a binary tree

   ☐ halving the list each time

   ☐ inserting each element into a sorted prefix

3. Which Big-O describes binary search?

   ☐ O(n)

   ☐ $O(n^2)$

   ☐ O(n log n)

   ☐ O(log n)

4. Every recursive algorithm must have a base case because:

   ☐ the compiler requires a comment

   ☐ without it the recursion never stops

   ☐ it speeds up the program

   ☐ it makes the code shorter

5. Linear search is the better choice when the data is:

   ☐ very large and sorted

   ☐ unsorted or only a small list

   ☐ in a database index

   ☐ already a binary tree

6. After the first pass of a bubble sort on 5, 3, 8, 1, what is the array? Write the four numbers separated by commas.

7. An O(n squared) algorithm takes 4 seconds on 1,000 items. Roughly how many seconds will it take on 2,000?

_____

8. Recursion is a natural fit for self-similar problems (trees, divide-and-conquer), but each call adds a stack frame – so without a base case it overflows the stack.

   ☐ True ☐ False

9. About how many comparisons does a binary search need for one million sorted items?

   _____

10. Insertion sort runs close to O(n) on small or nearly-sorted arrays, because few elements need to be shifted.

    ☐ True ☐ False

A-Level Computer Science

1. **checks each element in turn and works on any list**

   Linear search needs no preparation and works on any list, at worst O(n).

2. **repeatedly swapping adjacent out-of-order pairs**

   Each pass swaps adjacent pairs, ”bubbling” the largest element to the end.

3. **O(log n)**

   Halving the range each step is logarithmic – O(log n).

4. **without it the recursion never stops**

   The base case is the condition that ends the chain of calls; without it the recursion runs forever.

5. **unsorted or only a small list**

   With no order to exploit (or a tiny list), linear search avoids the cost of sorting first.

6. **3, 5, 1, 8**

   5 and 3 swap, 5 and 8 do not, 8 and 1 swap. The largest value has reached the end, so the next pass can be one comparison shorter.

7. **16**

   Doubling n quadruples an O(n squared) time. The same doubling would take an O(n) algorithm from 4 seconds to 8.

8. **True**

   A simple counting loop is cleaner for plain iteration; recursion shines when the problem contains smaller copies of itself.

9. **20**

   $\log_2(1\,000\,000) \approx 20$ – about 20 comparisons.

10. **True**

    On nearly-sorted data each new item is already almost in place – which is why insertion sort beats fancier sorts on small inputs.