2.8 Recursion and Advanced Algorithms: 10-minute Pack

Total Marks

AQA 7517 · Calculators not allowed · about 10 minutes

Name: _____ Date: ____ / ____ / _____

Answer ALL questions. Show all your working.

1 A programmer named Aisha is writing a recursive function to add up the first n elements of an array called list (indexed from 0).

(a) Using AQA pseudocode conventions, write a recursive function SumList(list, n) that returns the sum of the first n elements of list (i.e. list[0] up to list[n-1]). Your function should return 0 if n = 0. **(4)**

(b) State the time complexity of SumList in Big O notation, in terms of n. **(1)**

(c) State the space complexity of SumList in Big O notation, giving a reason related to the call stack. **(2)**

(d) An iterative version of SumList uses a single WHILE loop and one accumulator variable instead of recursion. State one advantage of the iterative version over the recursive version for a very large array. **(1)**

(Total for Question 1 is 8 marks)

© Revision Library | revisionlibrary.co.uk

Mark scheme · 2.8 Recursion and Advanced Algorithms: 10-minute Pack

Question 1

(a) function header FUNCTION SumList(list, n) defined **M1**

(a) base case IF n = 0 THEN RETURN 0 **M1**

(a) recursive case makes exactly one further call, SumList(list, n - 1) **M1**

(a) recursive case correctly adds the (n-1)th element, e.g. RETURN SumList(list, n - 1) + list[n - 1] cao **M1**

(a) Answer:

```
FUNCTION SumList(list, n)
   IF n = 0 THEN
      RETURN 0
   ELSE
      RETURN SumList(list, n - 1) + list[n - 1]
   ENDIF
ENDFUNCTION
```

(b) O(n) oe **B1**

(b) Answer: O(n)

(c) O(n) oe **B1**

(c) because a new stack frame is created for every recursive call, and up to n calls are in progress (not yet returned) at the deepest point of the recursion, before any of them return **B1**

(c) Answer: O(n), because up to n stack frames (one per recursive call) exist at the same time at the deepest point of the recursion, before any calls have returned.

(d) the iterative version uses O(1) (constant) additional memory rather than O(n), since it does not create a new stack frame for every element, avoiding the risk of a stack overflow error for very large n (oe) **B1**

(d) Answer: The iterative version only needs a fixed, small amount of extra memory (O(1)), regardless of the size of the array, because it does not create a new stack frame for every element; this avoids the risk of a stack overflow error that the recursive version could hit for a very large array.

© Revision Library | revisionlibrary.co.uk