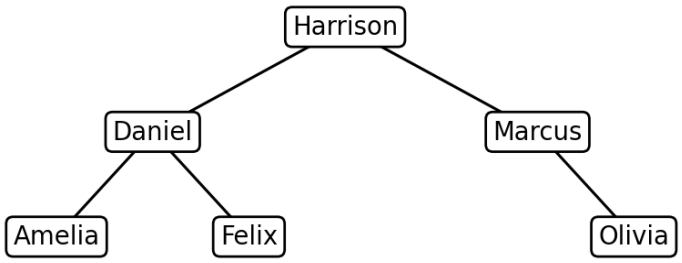
4.1.1 Recursion

1) The binary search tree in Figure 1 contains the following names. Figure 2 shows a recursive subroutine called `SearchTree` that searches the tree for a specified name.

Figure 1



`HasChild` returns `True` if a node has a child in the specified direction, and `False` otherwise. `node.left` and `node.right` refer to the corresponding child nodes.

Figure 2

```
FUNCTION SearchTree(target, node)
    OUTPUT "Visited ", node
    IF target == node THEN
        RETURN True
    ELSE IF target > node AND HasChild(node, right) THEN
        RETURN SearchTree(target, node.right)
    ELSE IF target < node AND HasChild(node, left) THEN
        RETURN SearchTree(target, node.left)
    ENDIF
    RETURN False
ENDFUNCTION
```

a) Explain what is meant by a recursive subroutine.

[1 mark]

- **A subroutine that calls itself;**

b) The `SearchTree` subroutine has two possible base cases. State one of the base cases.

[1 mark]

- **When target equals node // node == target // (When target does not equal node and) node is a leaf //;**

c) The following function call is made: `SearchTree(Grace, Harrison)`. Complete the table to show the function calls and output produced as the algorithm searches for Grace. You may not need to use every row.

[3 marks]

Function call	Output
`SearchTree(Grace, Harrison)`	**(Visited) Harisson;**
`SearchTree(Grace, Daniel)`	**(Visited) Daniel;**
`SearchTree(Grace, Felix)`	**(Visited) Felix;**

d) State the return value after the function call `SearchTree(Grace, Harrison)`.

[1 mark]

- **False;**

---

2) The function in Figure 3 is a recursive function.

Figure 3

```
FUNCTION Mystery(n)
    IF n <= 1 THEN
        RETURN 1
    ELSE
        RETURN n * Mystery(n - 2)
    ENDIF
ENDFUNCTION
```

a) Write down the sequence of function calls made when Mystery(7) is called.

[2 marks]

- **Mystery(7);**
- **Mystery(5);**
- **Mystery(3);**
- **Mystery(1);**

Marking points:
1 mark for first two bullet points, 1 mark for the second two.

b) Trace the function and state the value returned by Mystery(7). Show your working.

[3 marks]

- **Mystery(7);**
  n <= 1 → False
  return 7 * Mystery(5)
  Mystery(5);
  n <= 1 → False
  return 5 * Mystery(3)
- **Mystery(3);**
  n <= 1 → False
  return 3 * Mystery(1)
  Mystery(1);
  n <= 1 → True
  return 1
- **Final return value is 105**

---

3) The algorithm in Figure 4 is intended to calculate the factorial of a positive integer.

Figure 4

```
FUNCTION Factorial(n)
    IF __________ THEN
        RETURN 1
    ELSE
        RETURN ________________________
    ENDIF
ENDFUNCTION
```

a) Complete the condition for the base case.

[1 mark]

- **n = 1 // n = 0;**

b) Complete the recursive return statement.

[1 mark]

- **n * factorial(n-1);**

---

4) A function called PowerOfTwo should calculate 2 raised to the power of n.

For example:
`PowerOfTwo(0)` returns 1
`PowerOfTwo(4)` returns 16

a) Write a recursive algorithm for PowerOfTwo(n). Your algorithm must include a suitable base case and recursive case.

[4 marks]

- **FUNCTION PowerOfTwo(n)**
  IF n = 0 THEN
  RETURN 1
  ELSE
  RETURN 2 * PowerOfTwo(n - 1)
  ENDIF
  ENDFUNCTION

Marking points:
Correct base case, such as n = 0, returning 1. **[1]**
Function calls itself in the recursive case. **[1]**
Recursive call uses n - 1, moving towards the base case. **[1]**
Correct multiplication by 2 and return of the result. **[1]**