Unit 2 · Recursion

WORKSHEET

Full marks 33

Name _____ Class _____ Roll no. ____ Date _____ Marks _____ / 33

1 Key words [4]

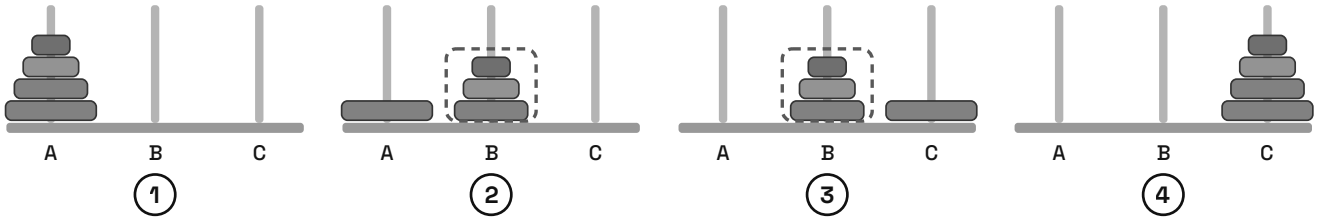
Fill in the blanks with words from the box. Two words in the box are not needed. Each blank carries 0.5 mark.

base case	**call stack**	**flood fill**	**fractal**	**loop**	**merge sort**	**queue**	**RecursionError**
recursive case	**recursive function**						

A function that calls itself is a _____. Its _____ gives an answer straight away, and its _____ calls the function again on a smaller copy of the problem. The calls still waiting pile up on the _____, and the last one made is the first to finish. Calls that never reach a base case stop the program with a _____. A picture made of smaller copies of itself is a _____. The paint bucket's _____ colors a square and then fills from each of its neighbors, and _____ sorts each half of a list, then merges the two halves.

2 Label the diagram [2]

The diagram shows the recursive way to move a tower of 4 disks from peg A to peg C. Write what each of 1 to 4 shows. Each carries 0.5 mark.



1. _____ 2. _____

3. _____ 4. _____

3 Multiple choice [5]

Choose the correct option for each of the following questions.

i. `def f(n): if n == 0: return 0; return n + f(n - 1)`. What is `f(4)`? [1]

a. 4 **b.** 8

c. 10 **d.** 24

ii. During a call of `factorial(6)` with the recursive function of this unit, how many calls are open at the most, at the same time? [1]

a. 1 **b.** 6

c. 7 **d.** 720

iii. A recursive function ends with `n * factorial(n - 1)` instead of `return n * factorial(n - 1)`. What does `factorial(3)` do? **[1]**

a. It gives back 6 as usual.

b. It gives back 0.

c. It stops with a TypeError, because a call gives back None and None can't be multiplied.

d. It runs for ever.

iv. How many moves does the Tower of Hanoi need for 8 disks? **[1]**

a. 16 **b.** 64

c. 128 **d.** 255

v. Which job is a loop clearly better suited to than recursion? **[1]**

a. finding the total size of a folder with folders inside

b. drawing a fractal tree

c. adding up the 100,000 numbers in a list

d. solving the Tower of Hanoi

4 What does it print? [5]

Write exactly what each program prints, in the box under it.

a. **[3]**

```
1  def show(n):
2      if n == 0:
3          return
4      print("down", n)
5      show(n - 1)
6      print("up", n)
7
8  show(3)
```

```
OUTPUT
```

b. **[2]**

```
1  def mystery(s):
2      if len(s) <= 1:
3          return s
4      return mystery(s[1:]) + s[0]
5
6  def digits(n):
7      if n < 10:
8          return n
9      return n % 10 + digits(n // 10)
10
11 print(mystery("Pokhara"), digits(2083))
```

```
OUTPUT
```

5 Trace the calls [4]

This function works out x to the power n by halving n. Fill in a row for each call of `power`, in the order the calls are made, then answer.

```
1  def power(x, n):
2      if n == 0:
3          return 1
4      half = power(x, n // 2)
5      if n % 2 == 0:
6          return half * half
7      return half * half * x
8
9  print(power(3, 5))
```

i. Fill in the table.

Call	n	half	Gives back
power(3, 5)	5		

ii. How many calls of `power` does `power(3, 1000)` make, counting the first?

6 Count the calls [2]

Read, then answer.

Climbing a stairway one step or two at a time, the number of ways to climb n steps is `ways(n)`. Fill in the table: what `ways(n)` gives back, and how many calls of `ways` it makes in all, counting the first.

```
1  def ways(n):
2      if n <= 1:
3          return 1
4      return ways(n - 1) + ways(n - 2)
```

n	ways(n)	Calls of ways
0	1	1
1	1	1

7 Find and fix the bug [3]

Read the program, then answer.

This flood fill should color the empty area of the little picture and print 5, but it stops with an error. (The picture has an outline of # all round, so it never reaches the edge.)

```
grid = [list("#####"), list("#..##"), list("#...#"), list("#####")]

def fill(grid, r, c, old, new):
    if grid[r][c] != old:
        return 0
    count = 1
    count = count + fill(grid, r - 1, c, old, new)
    count = count + fill(grid, r + 1, c, old, new)
    count = count + fill(grid, r, c - 1, old, new)
    count = count + fill(grid, r, c + 1, old, new)
    grid[r][c] = new
    return count

print(fill(grid, 1, 1, ".", "o"))
```

i. Which error stops the program?
ii. Why does it happen?
iii. How should the program be changed?

8 Write a program [4]

Write the program in Python on the lines. Any program that gives the right result gets the marks.

A list can hold numbers and other lists, which can hold lists too, as folders hold folders. Write a recursive function `total(items)` that gives back the sum of every number at any depth. `total([5, [10, [20, 3]], 2])` should give `40`, and `total([])` should give `0`. (`isinstance(x, list)` is True when x is a list.)

9 Short answer [2]

Write a short answer to the following question.

The fewest moves for a Tower of Hanoi of n disks follow the rule moves(n) = 2 × moves(n − 1) + 1. Explain where the rule comes from, and find moves(10).

10 Think [2]

Think, then answer.

`ways(40)`, with the function of **Count the calls**, makes 331,160,281 calls and takes minutes, though there are only 41 different questions, ways(0) to ways(40). Explain why it makes so many calls, and how the program could be changed to need far fewer.

11 Activity: Play the Tower of Hanoi (not marked)

At a computer, open the unit's page (the QR code) and find **Play the Tower of Hanoi.** Solve it for 3, 4 and 5 disks, and fill in the table.

Disks	My moves, first try	My fewest moves	The fewest possible
3			
4			
5			