

Programming Assignment 3

1. Recursive Maximum

Write a function `rec_max(nums)` that returns the maximum element of the list using recursion. Your code should not use any loops.

Sample 1 Input

[1, 2, 3]

Sample 1 Output

3

Sample 2 Input

[5, 1, 9, 4, 1, 7, 3, 0]

Sample 2 Output

9

2. Sum to K

Write a function `sum_to_k(K)` that prints all the lists containing positive integers whose sum is K.

Hint: Use a backtracking approach similar to the DNA strings example from class

Sample 1 Input

3

Sample 1 Output

[1, 1, 1]

[1, 2]

[2, 1]

[3]

3. Imbalance

Write a function `imbalance(nums)` that returns the smallest possible absolute difference between the sums of two subsets obtained by partitioning the list `nums`.

More precisely:

- Split `nums` into two disjoint sublists whose union is the entire list (every element appears in exactly one of the two sublists; either sublist may be empty).
- Compute the absolute difference between the sums of these two sublists.
- The imbalance of `nums` is the minimum such difference over all possible partitions.
- `imbalance(nums)` should return this minimum value.

As an example, if the input is `[1, 2, 3, 6]`, then the minimum difference is 0, given by `[1, 2, 3]` and `[6]`.

Sample 1 Input

`[1, 2, 3, 6]`

Sample 1 Output

0

Sample 2 Input

`[1, 2, 3, 4, 5, 6, 7, 8, 50]`

Sample 2 Output

14

Sample 3 Input

`[30, 19, 44, 31, 57, 11, 39]`

Sample 3 Output

1