Exercise Sheet $n°2$: Recursion & Recursive Functions

Note: Trace the execution of the first two functions using the call stack.

Exercise 1. Write a recursive function to compute the *binomial coefficient* $C(n, k)$. This coefficient represents the number of ways to choose k elements from a set of n elements. It is defined as:

$$C(n,k) = \frac{n!}{k!(n-k)!}$$

It can be computed using Pascal's Triangle relation:

$$C(n,k) = \begin{cases} 1, & \text{if } k = 0 \text{ or } k = n \\ C(n-1,k-1) + C(n-1,k), & \text{otherwise} \end{cases}$$

Exercise 2. Write a recursive function to convert a given decimal number into its binary representation. The function should take an integer as input and print its binary equivalent. The binary number should be printed in the correct order.

Exercise 3. The Tower of Hanoi is a puzzle where disks are moved between rods, following specific rules. The goal is to transfer all disks without placing a larger one on a smaller one. Write a recursive function to solve the Tower of Hanoi problem for n disks. The function should print moves required to transfer all disks from Source (A) to Destination (C) using Auxiliary (B).
Example:

```
"Move disk 1 from A to C"
"Move disk 2 from A to B"
"...etc."
```

Exercise 4. Convert the iterative *binarySearch* function studied in the previous exercise sheet to a recursive function.

Exercise 5. Write a recursive function to check if a given word is a palindrome (reads the same forward and backward). For example, "level" is a palindrome.

Exercise 6. Write a recursive function that removes consecutive duplicate characters in a string.
Example:

```
Input: "aaaabccaadeeee"
Output: "abcade"
```