General rules

- You have 2 hours to complete the test. People with special facilities have 2h20 minutes in total.
- The exam is "closed book", meaning that you can only make use of the material given to you.
- You are supposed to write the codes in Python language, but syntactic errors are allowed as far as the written algorithm can be well understood.
- For the loops, feel free to use for or whiles.
- The grade will be computed as the number of obtained points, plus 1.
- Keep the names of the variables and functions as stated in the question.
- You are only allowed to use simple arithmetic operations (including modulo). Do not use any mathematical libraries or modules.
- In all questions, you should always test your program by inputing some value where you know the result beforehand.
- **If you do not follow these instructions you will not receive any points in the respective question.**

Question 1 (2.5 points)

Write a function that receives a natural number n and returns the value of the sum of the squares of all integers from 1 to n using a loop. Then, write another function that returns the same quantity but using a recursion.

Question 2 (3 points)

Given a natural number n, the following algorithm can be defined:

- In case that n is even, then redefine n by halving it.
- In case that n is odd, then redefine n by tripling it and then adding one to the result.

Lothar Collatz conjectured that this algorithm would eventually reach 1 no matter the starting value of n. Write a program which verifies this conjecture for all starting values $n < 100$. If the algorithm does not terminate in 1000 steps, you may consider to have found a counter example to Collatz's conjecture, for the purpose of the question. The program should print a message with the result of test every time you end the test for a specific value of n, telling if the algorithm converge to 1 or not. The programme should use three different logical statements.

Question 3 (3.5 points)

The binary representation of an natural number n can be written as another integer, where each digit represents a coefficient for base 2. For example, the binary representation of 13 is 1101, since $1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 13$. You are requested to:

(a) Write a function which receives a natural number n and returns the largest power of 2, denoted by k, such that 2^k is smaller than n, e.g. largestPowerOfTwo(13) = 3.

(b) Write a function which receives as input a positive integer n and returns its binary representation. Assume given the function in (a).