Exercises - Week 3

! FOR THE PROBLEMS BELOW YOU ARE NOT ALLOWED TO USE ITERATIVE STRUCTURES (for/while), THE range() FUNCTION.

Exercise 1: Arithmetic progression

Implement in Python a recursive function to calculate the value of the rank term $n \in N$
, for the arithmetic progression defined by the relation:

$$A_n = 2 \times A_{n-1} - 3, \forall$$

$n \in N$

Term value is considered $A_0 = 2$. The function will receive as a single parameter the natural number n.

Exercise 2: Fibonacci

Implement in Python a recursive function to calculate the rank term n from the Fibonacci sequence:

$fibonacci(n) = $ $fibonacci(n-1) + fibonacci(n-2)$ 1 0 for $n \neq 1$ sauna $\neq 0$ for $n = 1$ for $n = 0$

Exercise 3: The sum of the first N natural numbers

Implement in Python a recursive function to calculate the sum of primes N natural numbers.

Exercise 4: Digits of a number

a) Implement in Python a recursive function to calculate the product of the digits of a number given as a parameter.
b) Implement in Python a recursive function to count how many digits the given number has as a parameter.
c) Implement in Python a recursive function that returns the maximum digit of the number given as parameter.
d) Implement in Python a recursive function that returns the number of even digits of the number given as parameter.

Exercise 5: The exponent of a natural number

Implement in Python a recursive function that calculates $A^n, A \geq 1, n \geq 0, A, n \in N$.

Exercise 6: Prime number

Implement in Python a function that returns True if a number n is prime, otherwise False.

Exercise 7: Greatest Common Divisor

Implement in Python a recursive function that calculates the greatest common divisor.

Hint: $gCD(A,b) = gCD(b,amodb), for \neq 0$

Exercise 8: The my_reverse function
Implement in Python a recursive function that reverses a string.
Hint: You can extract parts of a string using the syntax name[start:end]. If you want to extract a substring from a certain position to the end of the original string you can use name[start:].

Exercise 9: Interval
Implement a recursive function in Python that receives two values, min_value and max_value, respectively, and prints all the natural numbers that fall within the range whose endpoints are the two values.

Exercise 10: Appearances
a) Implement in Python a recursive function that checks whether a digit is present or not in a number.
b) Implement in Python a recursive function that returns the number of occurrences of a digit in a number.

Exercise 11: Palindrome
Implement in Python a function that determines whether or not a number is a palindrome. A number is considered a palindrome if it is equal to its opposite.
Examples:121, 34543, 1111 are such numbers

Exercise 12: Composition of functions
Implement in Python a recursive function that takes a function as parameters f, a real value x and a natural number n greater than or equal to 2. The function will return the value resulting from the function's composition f of $n$or applied to the point x.

Exercise 13: Remarkable sums
Implement in Python recursive functions that calculate the term n of the following amounts:
a)$S=1+\frac{1}{2}+\frac{1}{3}+...+\frac{1}{n+1}$b)$S=e^x=1+\frac{x^1}{1!}+\frac{x^2}{2!}+...+\frac{x^n}{n!}+...(Taylor\ series)$
Observation: For the Taylor series, try to find a calculation method that does not require the unnecessary recalculation of the power or the factorial at each step (you can either find a recurring relation or use the fact that in Python you can return multiple values)c)$S=\sqrt{1+\sqrt{1+\sqrt{1+...+\sqrt{1+\sqrt{1}}}}}$ n times

Exercise 14: Decimal-binary conversion
Write a recursive function in Python that takes a natural number as a parameter and returns a string representing the binary conversion of the given number as a parameter.
Example: For n=5 "101" will be returned.
Process: To convert from the decimal system to the binary system, first divide the chosen number by 2; the remainder is the least significant (rightmost) digit of the conversion result. The quotient is redistributed by 2, the remainder is noted, and the procedure is repeated with the new

quotient. The operation ends when the quotient becomes null.

Exercise 15: Triangle
Print the following triangular pattern with n lines using recursive functions.
Example: For $n=5$

```
1
2 2
3 3 3
4 4 4 4
5 5 5 5 5
```

Exercise 16: Remainders modulo p
In mathematics we know that if p is a prime number, and A is not divisible by p, then the string$A$,A^2,A^3, ...will reach 1 by taking the numbers modulo p (ie the remainders when dividing by$p$).Write a function that takes an integer as a parameter A and a number p(assumed prime) and returns the lowest power n for which$A^n \equiv 1 (mode)$(or return 0 if $A$is divided by p).
Hint: Write an auxiliary function that also has the exponent as parameters k respectively the value A^k (mode p), and which is called recursively until$A^k \equiv 1 (mode)$.
Example: Whether $p=7$and$A=4$. Then$A^2=16 \equiv 2 (mode 7)$,
and$A^3=A^2 * A \equiv 2 * 4 \equiv 1 (mode 7)$. The function will return 3.