Algorithm design

A. Assignment instructions and input/output instructions

Solve the following questions using flowcharts and/or pseudocode.

1. Build an algorithm that performs the following actions (in the order indicated):
   - enter two integer numbers,
   - calculate the sum and difference between them,
   - show both results.
2. Build an algorithm that performs the following actions (in the order indicated):
   - enter an integer number and a float number,
   - calculate the product between these two numbers,
   - show the result obtained.
3. Build an algorithm that performs the following actions (in the order indicated):
   - insert two float numbers, corresponding to the sides of a rectangle,
   - calculate the area of this ratangle,
   - show the value of the area obtained.
4. Build an algorithm that performs the following actions (in the order indicated):
   - enter two float number (x and y),
   - calculate the value of the function F(x,y) = = 2 sin(x) + cos(y) – tg(x+y),
   - Show the value of F(x,y) obtained.
5. Build an algorithm that performs the following actions (in the order indicated):
   - enter the base price of a product (float number) and a IVA rate (positive integer number),
   - calculate the final price of the product (base price plus IVA rate),
   - show the final price obtained.
6. Build an algorithm that performs the following actions (in the order indicated):
   - enter two integer numbers (associated with a student's grades for two curricular units),
   - calculate the average of these grades,
   - show the average obtained.
7. Build an algorithm that performs the following actions (in the order indicated):
   - enter a float number (corresponding to the temperature in Celsius degrees),
   - convert this number into the same temperature in Fahreneit degrees,
   - show the number in Fahreneit.

   The conversion formula is as follows:

   F = 9/5 x C + 32, where C is the value in Celsius and F is the value in Fahreneit.

8. Build an algorithm that performs the following actions (in the order indicated):
   - enter a float number (corresponding to the weight in kg),
   - convert this number into the same weight in grams,
   - show both numbers (in kg and grams).
9. Build an algorithm that performs the following actions (in the order indicated):
   - enter an integer number (associated with a time in seconds),
   - convert this number into the format HH:MM:SS (HH hours, MM minutes and SS seconds),
   - show the three numbers separately and in the order indicated (HH, MM and SS).
10. Build an algorithm that performs the following actions (in the order indicated):
    - enter a positive integer with three digits,
    - determine the digits that make up this number,
    - show these digits by the order indicated (hundreds, tens and units).

    Example: 937 is made up of the 9 (hundreds digit), 3 (tens digit) and 7 (units digit).

B. Conditional statements

Solve the following questions using flowcharts and/or pseudocode.

1. Build an algorithm that performs the following actions (in the order indicated):
   - ask the user to enter a non-zero positive integer number (>0),
   - check if this number is an even number or an odd number,
   - show a message with this information.
2. Build an algorithm that performs the following actions (in the order indicated):
   - ask the user to enter three integer numbers,
   - determine the largest of them,
   - show the highest obtained.
3. Build an algorithm that performs the following actions (in the order indicated):
   - ask the user to enter two non-zero positive integer numbers (> 0),
   - calculate the remainder of the integer division of the larger number by the smaller number,
   - show the result obtained.
4. Build an algorithm that performs the following actions (in the order indicated):
   - ask the user to enter two float numbers (X and Y),
   - show an ERROR message (if Y = 0) or the float number |X / Y| (otherwise).
5. Build an algorithm that performs the following actions (in the order indicated):
   - ask the user to enter two non-zero positive integer numbers (> 0), M and N,
   - check if M is a multiple of N,
   - show a message informing this situation (ex: 20 is a multiple of 5).

6. Build an algorithm that performs the following actions (in the order indicated):
   - ask the user to enter two flloat numbers (a and b) and one character ('+', '-', '*' or '/'),
   - calculate the result of the expression (ex: a+b),
   - show the result obtained.

7. A cash register records the price of products accompanied by a code that designates the type of product. Each product is subject to an additional fee distributed as follows:

product	code	fee
household electrical appliances	10010	5%
clothes	10020	0%
furniture	10030	10%
tools	10040	15%
perfumes	10050	30%

   Build a algorithm that receives the code for a product and its price without fees, calculates the final price of that product and shows the result obtained. If the product code entered does not correspond to any of the predicted situations, the algorithm should show the message: "Unknown product".

8. Build an algorithm that, given a product code (6-digit integer) and its base price (real number), determines the final price. The final price is obtained by adding the respective ICMS rate to the base price, which is associated with the last two digits of the product code, as shown in the following table:

cod	IVA
xxxx10	6%
xxxx20	13%
xxxx30	23%

   The algorithm must perform the following actions (in the order indicated): ask the user for a 6-digit integer (product code) and a real number (base price), determine the final price of the product and display a message indicating the product code product, the ICMS rate applied and the final price. If the product code entered does not correspond to any of the predicted situations, the algorithm should show the message: "Unknown product".

C. Repeat instructions (loops)

Solve the following questions using flowcharts and/or pseudocode.

1. Build an algorithm to show your name 20 times.
2. Build an algorithm to show the first 100 natural numbers.
3. Build an algorithm to calculate and show the sum of the first 100 natural numbers.
4. Build an algorithm to calculate the sum and the product of the odd natural numbers up to 1000. Finally, you show the results obtained.
5. Build an algorithm to read a sequence of positive integer numbers (ends by entering a negative value), calculate the sum between them and show the result.
6. Build an algorithm to determine the sum of N ($N \geq 2$) numbers. The algorithm should show an ERROR message if N is less than 2 and ask for a new number until it is valid.
7. Build an algorithm to determine the largest number in a sequence of N ($N \geq 1$) integer numbers given by the user. If $N < 1$ it must be requested again until a valid number is obtained. Number entry should end when N numbers have been entered. Finally, the algorithm should show the result obtained (highest value reported).
8. Build an algorithm to determine the product of integer numbers between N1 and N2. The algorithm must show an error message if $N2 \leq N1$ and ask for new numbers.
9. Build an algorithm to determine the product and sum of the even integer numbers (> 0) between N1 and N2. The algorithm must show an error message if $N2 \leq N1$ and ask for new numbers. At the end, it must show the values of the product and the sum.
10. Build an algorithm to determine the largest number and smallest number from a sequence of N integer numbers ($N \geq 2$) given by the user. If $N < 2$ it must be requested again until a valid number is obtained. Number entry should end when N numbers have been entered. At this point, the algorithm should show a message with the results obtained (highest and lowest values reported).
11. Build an algorithm to calculate the factorial of a positive integer number (> 0).
12. Build an algorithm to determine the largest and smallest number from a sequence of integer numbers entered by the user. Number entry should end when the zero number is entered. Finally, it should show the largest and smallest of the numbers entered.
13. Construct an algorithm to determine the arithmetic mean of a sequence of N ($N \geq 2$) integer numbers. If $N < 2$ then it must be requested again until a valid number is obtained. Number entry should end when N numbers have been entered. Finally, it should show a message with the calculated average value.

14. Build an algorithm to determine the arithmetic mean of a sequence of integer numbers. Entering numbers must end when the zero number is entered, and at least two numbers must be entered; otherwise an error message should be displayed. Finally, it should show a message with the average value.
15. Build an algorithm to determine the sum of the digits of a positive integer number.
16. Build an algorithm to show the first N numbers of the Fibonacci sequence. The Fibonacci sequence is as follows: 1, 1, 2, 3, 5, 8, 13, 21, … (after the first 1 each number in the sequence is the sum of the two previous numbers).
17. Build an algorithm to convert a number in binary format to a number indecimal format. Example: $10010_2 = 1 \times 2^4 + 0 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 = 18_{10}$.
18. Build an algorithm to convert a number in decimal format entered by the user, into the same number in binary format.
19. Build an algorithm to determine whether a given positive integer (> 0) is prime. A number is prime if it is divisible only by itself and the unit (for example: 13). If the number entered is not positive number, the application must ask for another number until a valid number is entered.
20. Build an algorithm to check whether a positive integer number is capicua. A number is capicua if it is the same number when read from left to right or vice-versa (ex: 202).