jgromero@inf.uc3m.es

Lesson 4. Control Flow
Exercises III – Advanced

Exercise 1. Print numbers in groups
Write a program that prints on the screen integer numbers in {1, 2, …, 1000}. Numbers will be printed in 20-element groups –the program asks the user whether he/she wants to continue or not after visualizing a group.

Extend the program to let the user specify the lower bound of the interval, the upper bound of the interval, and the size of the groups.

Exercise 2. Repeated values
Write a program that reads from the keyboard integer numbers until 0 is entered. Next, the program prints on the screen the length of the largest sequence of equal numbers entered in a row, and the value of this repeated number.

Example:
Input: *8 8 8 4 5 6 6 6 7 7 7 7 2 0*
Output: *The largest sequence of equal numbers (7) in a row had length (4)*

Exercise 3. Factorial
Write a program that reads a positive integer number and prints out its factorial.

NOTE: The factorial of an integer *n* is the product of the integer values from 0 to *n*: n! = 1 * 2 * 3 * … * (n-1) * n

Exercise 4. Root
Write a program that calculates the square root of a number with (at most) *n* decimal digits by applying the method of successive square approximations.

Compare the obtained result with the value calculated with the *sqrt* function of the <math.h> library.

Note: *Successive square approximation method*

This method is based on a 'trial and error' strategy. The algorithm to obtain the result starts with an initial candidate solution, which is modified in successive steps to converge to the real solution.

To calculate the square root of a number *num*, 0 is taken as the initial candidate *sol*(ution). In the first stage, *sol* is incremented in 1 while *sol* * *sol* < *num*. The procedure is repeated until *sol* * *sol* == *num* (success, end algorithm) or *sol* * *sol* > *num* (proceed to second stage). In the second stage, the procedure discards the last increment to *sol*. Next, it reduces the increment value by a 0.1 factor to proceed with the calculation of the next decimal of the approximation. In addition, it increments in 1 the precision achieved. The procedure continues by executing

again stage 1 with the new increment value if the achieved precision is less or equal than precision value specified by the user.

Exercise 5. Divison

Write a program that calculates the integer division of two positive values *a/b* by applying the method of successive subtractions.

Note: *Successive subtractions approximation method*

$$a/b = c \Rightarrow a \leq b \times c \Rightarrow a \leq \overbrace{b + b + \cdots + b}^{c\ times}$$

(i.e., *c* is the number of times that *b* can be added before exceeding *a*.)

Exercise 6. Magic number

The *magic* number of a person is calculated from his/her birthdate as follows:

Birthdate: 05/02/1973

Magic number: 5 + 2 + 1973 = 1980 --> 1 + 9 + 8 + 0 = 18 --> 9

Write a program that reads the user birthday and prints out the corresponding magic number. The program must check that the birthday is correct: *year* > 0; 1 <= *month* <= 12; 1 <= *day* <= days of *month*. Do not consider leap years.