Funciones - Actividades.

Los siguientes ejercicios están tomados del capítulo 6 del libro digital **"Introducción a la Programación con Python 3.**

Ejercicio 1:
Hacer el ejercicio 266

▶ 266 Define una función que devuelva el número de días que tiene un año determinado. Ten en cuenta que un año es bisiesto si es divisible por 4 y no divisible por 100, excepto si es también divisible por 400, en cuyo caso es bisiesto.

(Ejemplos: El número de días de 2002 es 365: el número 2002 no es divisible por 4, así que no es bisiesto. El año 2004 es bisiesto y tiene 366 días: el número 2004 es divisible por 4, pero no por 100, así que es bisiesto. El año 1900 es divisible por 4, pero no es bisiesto porque es divisible por 100 y no por 400. El año 2000 sí es bisiesto: el número 2000 es divisible por 4 y, aunque es divisible por 100, también lo es por 400).

Ejercicio 2:
Hacer el ejercicio 281

▶ 281 Diseña una función que devuelva la solución de la ecuación lineal $ax + b = 0$ dados a y b. Si la ecuación tiene infinitas soluciones o no tiene solución alguna, la función lo detectará y devolverá el valor None.

Ejercicio 3:
Hacer el ejercicio 286

▶ 286 Diseña una función que diga (mediante la devolución de True o False) si dos números son *amigos*. Dos números son amigos si la suma de los divisores del primero (excluído él) es igual al segundo y viceversa.

Ejercicio 4:
Hacer el ejercicio 288

▶ 288 En un programa que estamos diseñando preguntamos al usuario numerosas cuestiones que requieren una respuesta afirmativa o negativa. Diseña una función llamada *sí_o_no* que reciba una cadena (la pregunta). Dicha cadena se mostrará por pantalla y se solicitará al usuario que responda. Solo aceptaremos como respuestas válidas `'sí'`, `'s'`, `'Sí'`, `'SÍ'`, `'no'`, `'n'`, `'No'`, `'NO'`, las cuatro primeras para respuestas afirmativas y las cuatro últimas para respuestas negativas. Cada vez que el usuario se equivoque, en pantalla aparecerá un mensaje que le recuerde las respuestas aceptables. La función devolverá True si la respuesta es afirmativa, y False en caso contrario.

Ejercicio 5:
Hacer el ejercicio 292

▶ 292 Diseña un programa que, dado un número n, muestre por pantalla todas las parejas de números amigos menores que n. La impresión de los resultados debe hacerse desde un procedimiento.

Dos números amigos solo deberán aparecer una vez por pantalla. Por ejemplo, 220 y 284 son amigos: si aparece el mensaje «`220 y 284 son amigos`», no podrá aparecer el mensaje «`284 y 220 son amigos`», pues es redundante.

Debes diseñar una función que diga si dos números son amigos y un procedimiento que muestre la tabla.

Ejercicio 6:
Hacer el ejercicio 293

▶ 293 Implementa un procedimiento Python tal que, dado un número entero, muestre por pantalla sus cifras en orden inverso. Por ejemplo, si el procedimiento recibe el número 324, mostrará por pantalla el 4, el 2 y el 3 (en líneas diferentes).

Ejercicio 7:
Hacer el ejercicio 294

294 Diseña una función *es_primo* que determine si un número es primo (devolviendo True) o no (devolviendo False). Diseña a continuación un procedimiento *muestra_primos* que reciba un número y muestre por pantalla todos los números primos entre 1 y dicho número.

Ejercicio 8:

a) Diseña una función **es_perfecto** que determine si un número es perfecto (la suma de sus divisores es igual al número).
b) Pedir un número entero positivo al usuario hasta asegurarse que el número introducido es perfecto. A ese número lo llamaremos **p.**
Luego generar los primeros **p** números perfectos.

Ejercicio 9:

a) Diseña una función **mcd** que calcule el máximo común divisor de dos números.
b) Diseña una función **son_coprimos** que determine si dos números son coprimos. Dos números son coprimos cuando su máximo común divisor es 1.
c) Escribe un programa que pida al usuario dos números mayores que 1 hasta asegurarse que entre sí son coprimos.

Ejercicio 10:

a) Diseñar una función que obtenga el **factorial** de un número positivo.
b) Diseñar una función que calcule la **combinación** de m tomados de a n.
c) Diseñar una función que calcule las **variaciones** de m tomadas de a n.
d) Escribir un programa que le permita al usuario introducir un número y elegir alguna de esas tres operaciones:
    a. Permutación de n.
    b. Variación de m tomados de a n.
    c. Combinación de m tomados de a n.

Fórmulas:

- Permutación de n = n! = factorial (n)
- Variación de m tomados de a n = $\frac{m!}{(m-n)!}$
- Combinación de m tomados de a n = $\frac{m!}{(m-n)!n!}$

Ejercicio 11:

Diseña una función **mcm** que calcule el mínimo común múltiplo de dos números.

Ejercicio 12:

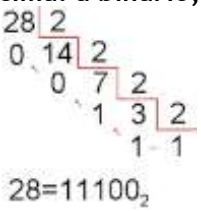
Escribe un programa que pida dos fracciones y luego:

a) las simplifique y muestre simplificadas
b) las sume y muestre su resultado simplificado.

Ejercicio 13:

a) Diseña una función **dectobin** que obtenga el número binario correspondiente a un número decimal.

Para convertir un número de decimal a binario, se divide sucesivamente por 2:



b) Diseña una función **bintodec** que obtenga el número decimal correspondiente a un número binario.

Para convertir un número de binario a decimal, se suman las potencias de 2: