Calculando factoriales de números grandes

El día de ayer por medio del perfil de línea de tiempo del Doctor Juan González observamos el interesante artículo tutorial sobre Python publicado por Juan J. Merelo Guervós el cual incluye retos de programación. Ni tardo ni perezoso, a pesar de llegar cansado a mi hogar, y tras una vigorizante taza de café negro con leche de avena sin azúcar nos dimos a la tarea de practicar algunas cositas para mantener el cerebro en forma y al día con las novedades en programación.

https://twitter.com/jjmerelo/status/882846481010569216

Calculando el factorial de un número.

Definición.

No vamos a hacer esta entrada muy profunda, ya que debemos ir a trabajar para ganarnos el pan de cada día, directo y rápido: el factorial de un número viene expresado por la multiplicación sucesiva de sus antecesores, uno a uno, hasta llegar a uno y se denota por " ***n!*** " ; así, por ejemplo 5!= 5 x 4 x 3 x 2 x 1 = 120 (las equis indican multiplicación).

El reto.

Pues arroja el guante el sr. Merelo con una función recursiva escrita, por supuesto, en lenguaje Python para calcular un factorial y tras ensayo y error llegamos a una satisfactoria *y muy parecida* a la ya famosa y escrita por todos lados en internet (para ser honestos en algún momento vimos esta función pero no recordabamos... hasta que el buscador DuckDuckGo acudió en nuestra ayuda, un asistente tan capaz -en memoria- como todas las asistentes que tuvo Albert Einstein -que en paz descansen, Valentine Bargmann y el Doctor Einstein-).

Para ilustraros cómo funciona, aprovechando nuestro mundo tecnológico *-y añorando el gis y el pizarrón-* os colocamos un gif animado tomado de penjee.com, un sitio web dedicado también a la enseñanza de Python:

```
def factorial(num):    if num == 1:      return 1    else:      return nu
m * factorial(num - 1)
```

Calculando el factorial de un número grande.

Hasta acá todo muy bien, pero el reto no finalizaba allí, una segunda pregunta era ¿qué tal calcula los números grandes? Como bien podéis imaginar, el multiplicar y multiplicar en ordenador pues en algún momento se nos agota la memoria *suponíamos nosotros...* pues que aquí está la pega del asunto, tras calcular el factorial 120! de manera fácil (en Wikipedia en inglés denotan que normalmente el máximo factorial que se puede calcular con un procesador de 64 bits en una variable de tipo entero es 20! -ya nosotros hemos comentado el poder de cómputo de los procesadores y sistemas operativos de 64 bits-) *pues nos envalentonamos a ir "**al infinito y más allá".***

Siendo así la cosa le establecemos a 1200 (un mil doscientos) a la función de marras y nos arrojó este lindísimo mensaje de error que os mostramos:

Pero leyendo nos percatamos que el problema no es el desbordamiento de memoria, no, ***el desbordamiento es un error en la profundidad de recursión:***

```
RecursionError: maximum recursion depth exceeded in comparison
```

El contrareto.

Ya esto se salía del reto impuesto, *es un contrareto que nos presenta Python* pues claro, en un entorno de programación uno debe siempre imponer límites considerando que aún no ha llegado la computación cuántica (y aún así ésta tiene límites *en este universo*). Indagando, y hay bastante material sobre el tema, llegamos a la librería ***sys*** encargada de manejar tales asuntos y de hecho el manual en línea de Pyton nos advierte:

> Set the maximum depth of the Python interpreter stack to limit. This limit prevents infinite recursion from causing an overflow of the C stack and crashing Python. The highest possible limit is platform-dependent. A user may need to set the limit higher when she has a program that requires deep recursion and a platform that supports a higher limit. This should be done with care, because a too-high limit can lead to a crash.

Que traducimos al castellano:

> Establece el límite de la profundidad máxima de la pila del interprete Python. Este límite previene una recursión infinita ocasionando un desbordamiento en la pila (del lenguaje) C y la consecuente falla de Python (*Dios nos ampare que alguien crea que Python es falible*†). El límite más alto posible es dependiente de la plataforma empleada. Un usuario puede necesitar ajustar el límite superior cuando tenga un programa que requiera una mayor recursión y la plataforma que lo soporta lo permita. Esto debería hacerse con cuidado porque un límite muy alto puede conducir a un fallo (*Dios nos ampare que alguien crea que Python es falible†*).
>
> †(Nota del traductor en tono de sarcasmo).

O inventamos o erramos.

Como nuestro insigne Don Simón Rordríguez nos legó, debemos inventar y a lo sumo fallar (y reintentar y así sucesivamente hasta lograr inventar) nosotros pensamos ***¿Y si establecemos el límite superior de la pila de procedimientos (límite de recursión) precisamente de manera recursiva? ?***

Pues acá vamos entonces:

```
import sys def factorial(num): '''Return large factorial by Jimmy Olano
 Sayago (GNU General Public License v3.0) ''' if num
```