Tarea 04

FISI2026-201901

Puede resolver esta tarea trabajando desde el enlace Binder del curso en https://github.com/ComputoCienciasUniandes/FISI2026-201910 o en el salón de CompuFísica Y-110B

Entrega:
Fecha: miércoles 27-feb-2019
Hora: 7:50 am
Mecanismo único de Entrega: Archivo comprimido en Sicuaplus
Puntuación Máxima: 100 puntos.

Lea atentamente toda la tarea antes de empezar a trabajar.

Cree una carpeta que se llame "`hw04-`" más su usuario de uniandes.
En mi caso se llamaría "`hw04-fl.gomez10`". Ingrese a ella y trabajaremos todo el tiempo dentro de este directorio.

```
:~$ mkdir hw04-username
```

Trabajo previo a Clase:

Revisar el vídeo correspondiente a funciones y recursividad. Los ejercicios 1, 2 y 3 se resuelven antes de llegar a clase.

Ejercicio 1

(10 puntos)

Cree un archivo "ejercicio01.py" que calcule **recursivamente** la suma de los N primeros números enteros. Se debe ejecutar desde la línea de comandos como:

```
~:$ python ejercicio01.py 5
```

En funciones recursivas es útil imprimir el paso a paso. Incluya un par de líneas que indiquen cuando la función que suma recursivamente está en el caso base o en el caso general.

Recuerde que para leer el argumento en la línea de entrada puede usar `sys.argv[]:` en el archivo ejercicio01.py

```
# coding utf-8
import sys
print( sys.argv[1] )
```

Si no suma recursivamente, no se obtiene puntuación por este ejercicio.

Ejercicio 2

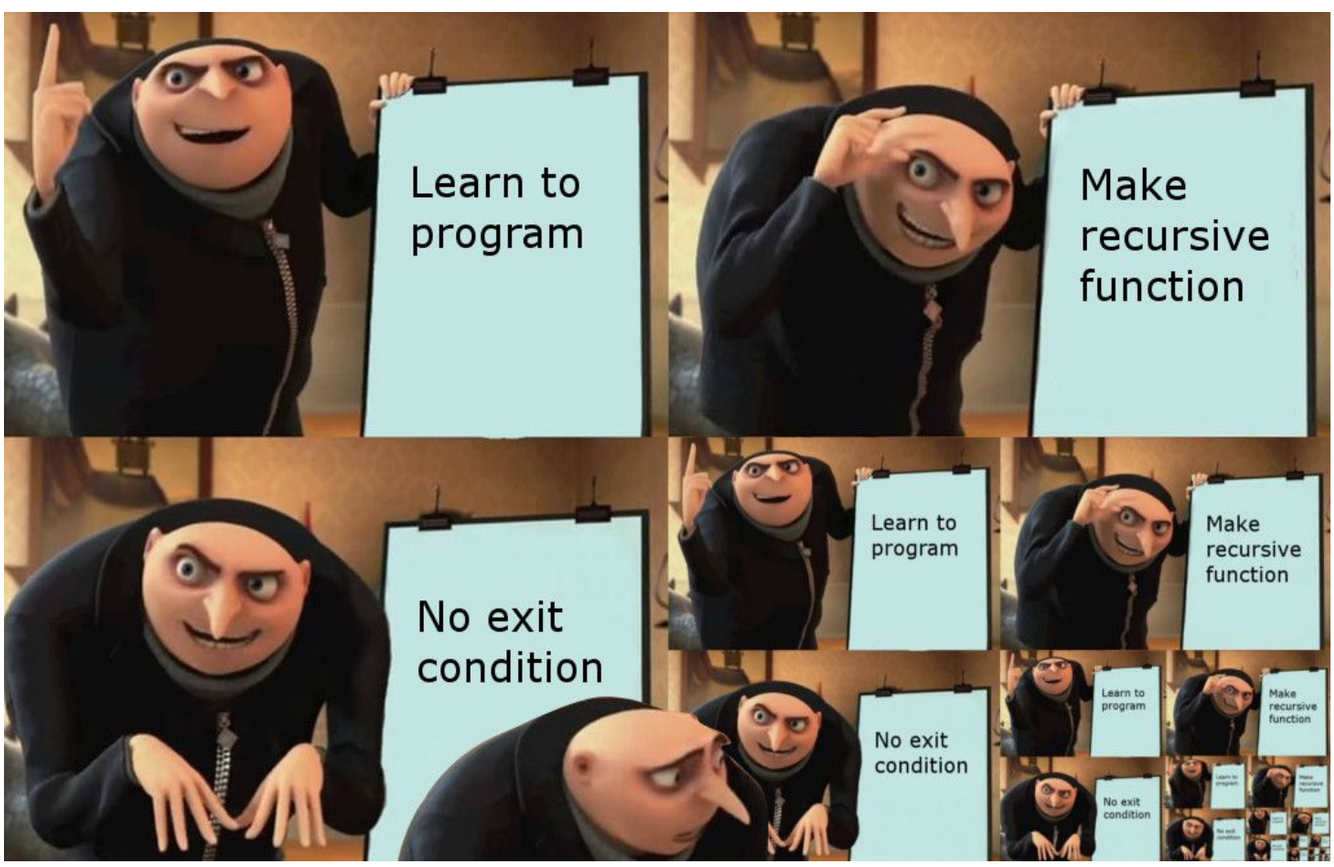
(10 puntos)

Cree un archivo “ejercicio02.py” que calcule **recursivamente** e imprima el triángulo de Pascal (Que funciona para el Biniomio de Newton) con un argumento entero de entrada:

```
~:$ python ejercicio01.py 5
        1
      1   1
    1   2   1
  1   3   3   1
1   4   6   4   1
```

```
Protip: Las listas pueden ser muy útiles acá.
[1]
[1,1]
[1,2,1]
[1,3,3,1]
[1,4,6,4,1]
...
```

Si no se calcula recursivamente, no se obtiene puntuación por este ejercicio.



Ejercicio 3

(40 puntos)

Escriba un programa (ejercicio02.py) que resuelva el problema de la Torre de Hanoi.

Dice la leyenda (guiño, guiño) que en un monasterio Indio se tienen tres postes de marfil. En el primer poste hay 64 discos de oro de distintos diámetros. Están puestos uno sobre el otro en orden según su tamaño, dejando el disco más pequeño en la cima y el más grande abajo. El objetivo de los monjes en el monasterio es mover la torre de discos de un poste a otro. Pero se deben obedecer las siguientes reglas:

- Se pueden mover los discos a cualquiera de los otros postes.
- Se puede mover un disco a la vez.
- Un disco grande no puede ubicarse sobre un disco pequeño.

Cuando los monjes terminen su labor, el final del mundo llegará.

Se puede pensar el problema en forma recursiva como:

- Si queremos mover el disco más grande (diámetro N) del poste inicial al poste final, debemos mover todos los discos de encima.
- Si queremos mover el disco de diámetro (N+1) movemos el resto de los discos de encima…

El programa debe contar e imprimir cada uno de los movimientos. por ejemplo, si N=4, los primeros movimientos son:

```
.4 3 2 1
.
.

.4 3 2
.1
.

.4 3
.1
.2
```

Ejercicio 4

(40 puntos)

Calcular cos(x) con los cuatro primeros términos de la serie de Taylor.

5.6. The Taylor series for $\sin x$, $\cos x$, and e^x are well known for real x:

$$\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \dots$$

$$\cos x = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \dots$$

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots$$

para x = pi / 6 radianes. Escriba la función factorial de manera recursiva.

Comparar con el valor de cos(x) que provee la librería math de python (preferiblemente una diferencia porcentual).

```
import math as m

# Imprime el valor de PI de la liberia
print( m.pi )

# Un ángulo en radianes: pi/6
x = m.pi / 6

# Imprime el valor de la funcion coseno de la librería
print( m.cos( x ) )

def factorial(n):
     operaciones con n.
     return algo.
```

Formato de Entrega

Carpeta comprimida en formato tar llamada `hw04-username.tar` que contenga los scripts de los tres ejercicios.