

**UADER Facultad de Ciencia y Tecnología – CdelU**

***Carrera:* Licenciatura en Sistemas de Información**

***Asignatura:* Ingeniería de Software 2**

***Autor:* Escar Camilo**

***Profesor:* Dr. Pedro E. Colla**

**Ayudante: Hernán Sanchez**

***Año:* 3er Año**

***Ciclo lectivo:* 2023**

**Trabajo práctico Nº 1 – Conceptos básicos**

**Introducción a la Ingeniería de Software**

**Gestión de la configuración**

**Instale los siguientes paquetes de software en la versión apropiada para el sistema operativo que utilice.**

* **Git. ✔**
* **Python 3 (instalar desde python.org) ✔**
* **Pip3 (instalar desde python.org) ✔**

Obtenga una cuenta en www.github.com y a la que llamará UADER\_IS2\_{su\_apellido}, a continuación, genere una estructura de carpetas formada por:

* src
* doc
* bin
* script

Obtenga el programa primos.py (en Source Python.gz) y siga las siguientes consignas:

* Colóquelo en el directorio src local en su máquina.
* Ejecútelo con “python3 primos.py” y verifique que corre bien.
* Sincronícelo con el repositorio github.
  + git add .
  + git commit -n carga\_inicial
  + git push origin
  + verifique la correcta actualización.
* Simule el borrado “accidental” en su máquina y a continuación recupere el archivo desde el repositorio Github.
* Coloque comentarios al programa, al finalizar pruebe que el mismo siga ejecutando correctamente. Al hacerlo sincronice con el repositorio GitHub.

\*A la hora de ejecutar el comando “§ git push origin” me genera un error a la hora, corregir mañana // **CORREGIDO**

Actualice el propósito del repositorio en GitHub por medio de un archivo README.md que coloque en la raíz del repositorio. Actualice y verifique. Utilice la notación simple para que resalten los títulos, secciones, referencias. Incorpore al menos tres niveles de títulos, dos listas ordenadas, una lista numerada, una figura y una referencia a una página Web fuera del repositorio. ✔

**Utilice el comando pip para instalar el paquete Matplotlib e intente ejecute el archivo code/charts/line.py ✔**

Obtenga el programa fuente factorial.py y ejecute con python3 factorial 10 confirme que funciona correctamente. Guarde en repositorio GitHub en una carpeta específica dentro del árbol “src” denominada “factorial”.

* Realice una modificación al programa para que si se omite el número como argumento lo solicite. Pruebe. Sincronice en GitHub. ✔
* Modifique el argumento (y el ingreso manual) para aceptar números en el rango desde-hasta (ej. 4-8) y que calcule los factoriales entre ambos extremos. Pruebe. Sincronice en GitHub.✔
* Modifique el argumento (y el ingreso manual) para que acepte rangos sin límite inferior “-hasta” calculando entre 1 y el número indicado (ejemplo “-10”), lo mismo para “desde-“ calculando entre el número indicado y 60. Tenga la precaución de transformar las cadenas de caracteres de la especificación de argumentos en valores enteros antes de intentar operaciones matemáticas. Pruebe. Sincronice en GitHub.
* Agregue comentarios al código generado. Pruebe. Sincronice con GitHub

Genere un proyecto copia del anterior denominado “factorial\_OOP” donde tomando como base el programa “factorial.py” genere un programa “factorial\_OOP.py” donde se construya la lógica de cálculo de factorial mediante una clase Factorial con un constructor y un método “run(min,max)” que calcule como resultado el factorial entre los números min y max. Pruebe. Sincronice en GitHub. **❌ \*\*ver run(min,max)**

Desarrolle un programa en python para calcular el número de Collatz (conjetura 2n+1) para los números entre 1 y 10000, realice un gráfico donde en el eje de órdenadas muestre el número n de comienzo de la secuencia y en la absisas el número de iteraciones que tardó en converger a una secuenca repetitiva. Coloque en una carpeta en la jerarquía “src”. Pruebe. Sincronice en GitHub. ✔

A la hora de guardar un trabajo lo hacia guardando un archivo nuevo, no actualizando el repositorio