

Informática - Práctica de laboratorio 06

Programación VI: Errepaso y Recursividad

1. Normas de Entrega

Todas las prácticas deberán entregarse siguiendo una convención de nombres y un formato específico. En el caso de las prácticas de Python, se solicitará un script de Python para cada ejercicio, generando así varios archivos de texto por práctica.

Todas las prácticas se desarrollarán utilizando el IDE [spyder](https://www.spyder-ide.org/). [Spyder](https://www.spyder-ide.org/) es un entorno científico de código abierto y gratuito diseñado para científicos, ingenieros y analistas de datos. Ofrece una combinación única de funcionalidades avanzadas para edición, análisis, depuración y perfilado, con capacidades excelentes para exploración de datos, ejecución interactiva, inspección profunda y visualización de paquetes científicos. Puedes acceder a él aquí: <https://www.spyder-ide.org/>.

1.1. Nombre del Archivo

Cada ejercicio de programación debe estar codificado en un archivo independiente, es decir, cada ejercicio será un archivo separado. A menos que se indique lo contrario, la convención de nombres será *Ejercicio_YY.py*, donde XX será el número de la práctica y YY el número del ejercicio en el documento de la práctica. Por ejemplo, el primer ejercicio de la práctica cero llevará el nombre *Ejercicio_01.py*.

Para entregar los ejercicios, debes acceder a la sección del curso en mi aula virtual (20XX_0_501103_91_G). Allí, en la columna izquierda, aparecerá la sección *Tareas*.



Figura 1: Primer paso para entregar las tareas.

En cada entrega aparecerá una tarea y podrás entregar los ejercicios. También tendrás la opción de escribir un mensaje junto con la entrega. Para cada práctica, se solicitarán diferentes ejercicios, y deberás subir un archivo para cada ejercicio (ver imagen 2).

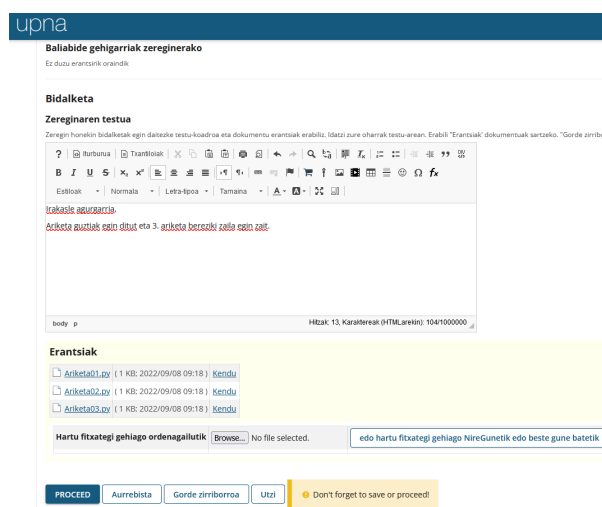


Figura 2: Primer paso para entregar las tareas.

2. Normas de Desarrollo

Con cada práctica tendrás una chuleta (LabXXPR_CheatSheet). En la chuleta encontrarás un resumen del material trabajado en clase. Por ejemplo, en Lab01PR_CheatSheet, se presentarán los elementos necesarios para realizar Lab01PR. Se recomienda imprimir la chuleta o, al menos, tenerla a la vista mientras desarrollas el programa. La CheatSheet contendrá tipos de datos, operadores, estructuras y funciones trabajadas en clase. Cualquier elemento de Python utilizado fuera de estos será evaluado con una nota de 0 en los evaluables.

Al desarrollar los programas, debes seguir las siguientes normas

1. Los nombres de las variables deben ser descriptivos
 - Todos en minúsculas
 - Los nombres compuestos por más de una palabra se separarán con '_'
2. Todos los programas deben tener una cabecera obligatoria
 - La primera línea de comentario debe ser `# python script`
 - En la segunda línea debe aparecer `# Autor: tu nombre`, donde `tu nombre` es el nombre del estudiante
 - En la tercera línea debe aparecer `# Descripción: descripción del programa`, donde `descripción del programa` es una frase o párrafo que describa el programa
3. No se deben usar funciones que no aparezcan en la CheatSheet.

3. Ariketa Zerrenda

Ejercicio 1. Define una función llamada factorial, cuya entrada será un número entero y debe devolver su factorial.

□

Ejercicio 2. Crea una función que, utilizando recursividad, repita `print('hola mundo')` `n` veces.

□

Ejercicio 3. Crea una función que calcule el factorial de un número.

□

Ejercicio 4. Calcula divisiones con decimales utilizando operadores enteros (división entera y operador módulo). El programa calculará hasta un máximo de 5 decimales.

□

Ejercicio 5. Crea una función llamada fibonacci que tome un número entero `n` como argumento y genere los primeros `n` términos de la secuencia de Fibonacci como una lista. Este ejercicio debe resolverse utilizando recursividad.

□

Ejercicio 6. Crea una función llamada `contar_digitos` que tome un número entero como entrada y cuente cuántos dígitos tiene. Este ejercicio debe resolverse utilizando recursividad.

□

Ejercicio 7. Genera un número aleatorio entre 1 y 100 y desafía al usuario a adivinarlo. Proporciona pistas como "más alto." "más bajo" hasta que adivinen el número correcto.

□

Ejercicio 8. Crea una función que determine si una cadena es un palíndromo usando recursividad.

□

Ejercicio 9. Escribe una función que sume los dígitos de un número usando recursividad.

□

Ejercicio 10. Crea una función que invierta una cadena usando recursividad.

□

Ejercicio 11. Escribe una función que cuente cuántas veces aparece una letra en una cadena usando recursividad.

□