

Informática - Práctica de laboratorio 04

Programación IV: Interacción con el usuario, listas y for [each]

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).

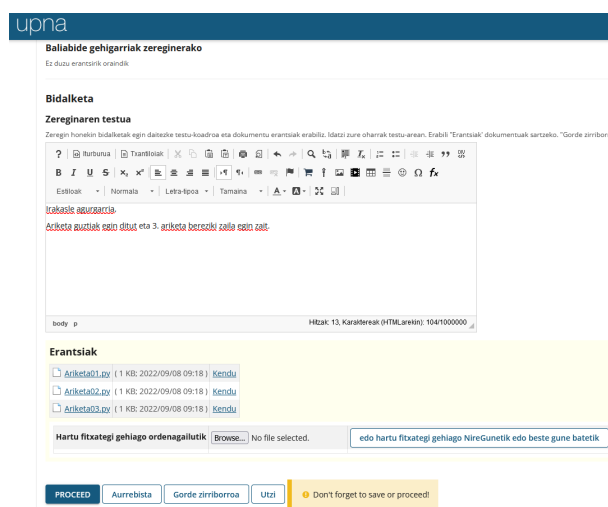


Figura 2: Primer paso para entregar las tareas.

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. Lista de Ejercicios

Ejercicio 1. Escribe una función que reciba un número representando el día de la semana (1 para lunes, 2 para martes, etc.) y luego muestre el nombre del día de la semana correspondiente. Si el valor ingresado no está en el rango de 1 a 7, el mensaje a mostrar debe ser **Valor desconocido**. Para este ejercicio, debes usar el tipo de datos `list`.

Código 1: Programa principal del ejercicio.

```
1 # definir la funcion
2 ...
3
4 # ejecucion de la funcion
5 dia_semana(1)
6 dia_semana(4)
7 dia_semana(-5)
```



Ejercicio 2. Escribe una función que reciba el número del mes (1 para enero, 2 para febrero, etc.) y el año. Luego, debe mostrar la cantidad de días que tiene ese mes. Ten en cuenta que febrero tiene 29 días en años bisiestos, para esto puedes usar la función `es_bisiesto` del ejercicio 02. Si el valor ingresado no está en el rango de 1 a 12, el mensaje a mostrar debe ser **Valor desconocido**. Para este ejercicio, debes usar el tipo de datos `list`.

Código 2: Programa principal del ejercicio.

```
1 # definir la funcion
2 ...
3
4 # ejecucion de la funcion
5 dias_mes(1,2020)
6 dias_mes(4,2021)
7 dias_mes(-5,-2023)
```



Ejercicio 3. Escribe una función que calcule la longitud de una lista.



Ejercicio 4. Escribe una función que verifique si todos los elementos de una lista son números. La función debe devolver un valor booleano.



Ejercicio 5. Escribe una función llamada `reverso_cadena`. La función tomará una cadena como entrada y devolverá la cadena en orden inverso.

□

Ejercicio 6. Escribe una función que permita al usuario crear una contraseña. El usuario deberá ingresar una contraseña y la función verificará si cumple con ciertos criterios de seguridad. Si la contraseña cumple con los requisitos, la función la devolverá. De lo contrario, ofrecerá la oportunidad de generar una nueva contraseña. Los criterios de seguridad incluyen tener al menos 8 caracteres, una letra mayúscula, una letra minúscula, un número y al menos uno de los siguientes caracteres especiales: -, @, ., #, o -.

□

Ejercicio 7. Crea una función que cuente los dígitos de un número. La función se llamará `contar_digitos`. Tomará un número entero como entrada y devolverá la cantidad de dígitos del número.

□

Ejercicio 8. Crea una función que solicite al usuario ingresar `n` números (enteros o reales). El único argumento de entrada será `n`, que indicará cuántos números debe ingresar el usuario. Si alguno de los elementos ingresados no es un número, la función deberá mostrar el mensaje `valor incorrecto`. y terminar.

□

Ejercicio 9. Crea una función que sume todos los elementos de una lista de números. La función se llamará `suma` y devolverá la suma calculada.

□

Ejercicio 10. Crea una función que calcule el promedio de todos los elementos de una lista de números. La función se llamará `promedio` y devolverá el número promedio calculado.

□

Ejercicio 11. Crea una función que identifique el valor máximo de una lista de números. La función se llamará `maximo` y devolverá el número máximo calculado.

□

Ejercicio 12. Crea una función que identifique el valor mínimo de una lista de números. La función se llamará `minimo` y devolverá el número mínimo calculado.

□

Ejercicio 13. Crea una función que ordene los dígitos de una lista de números según su valor. La función se llamará `ordenatu` y tomará una lista de números enteros como entrada. Devolverá una lista con los números ordenados de menor a mayor según sus dígitos.

□

Ejercicio 14. Crea una función que calcule la mediana de una lista de números. La función se llamará `mediana` y devolverá el valor de la mediana. Puedes utilizar la función 13 para ordenar la lista antes de calcular la mediana. Consulta <https://eu.wikipedia.org/wiki/Mediana> para más detalles.

□

Ejercicio 15. Utiliza las funciones de 8, 9, 10, 11, 12, y 14 para crear un programa que calcule parámetros básicos de estadística descriptiva para una lista de números. El programa pedirá al usuario la cantidad de elementos para la lista y luego solicitará que ingrese los números. Finalmente, mostrará la suma, el promedio, el máximo, el mínimo y la mediana de los números. Puedes utilizar cualquier función adicional que hayas creado para este propósito.

□