Universidad Pública de Navarra
Curso 2024/2025

Informática - Práctica de laboratorio 01

Programación: Fundamentos de programación

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`. `Spyder` 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*.

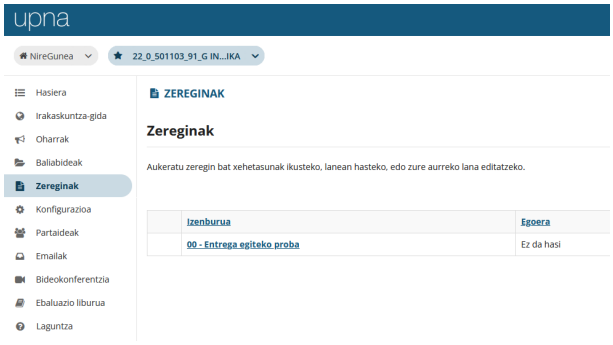

Figura 1: Primer paso para entregar las tareas.

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).

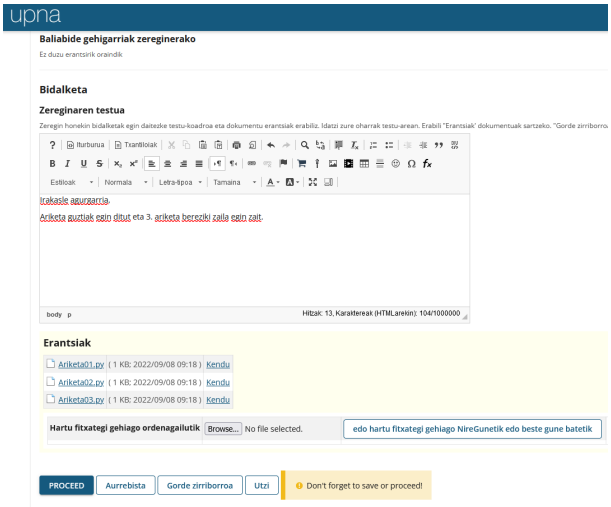

Figura 2: Primer paso para entregar las tareas.

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 y ejecuta un *script* que muestre el mensaje `¡Hola Mundo!` en pantalla.

□

El ejercicio 1 es muy sencillo porque solo se necesita usar la función `print`. El resultado puede verse en el 1 Script. En este caso, la función se utiliza para mostrar directamente un texto pasado como argumento.

Código 1: Resultado del ejercicio 1 en la práctica Lab01.

```
1 # Python script
2 # Autor: Unai Perez-Goya
3 # Descripcion: Ejercicio 01
4
5 print('Hola mundo!')
```

Ejercicio 2. Escribe un *script* que muestre tu nombre en pantalla.

□

Ejercicio 3. Escribe un *script* que muestre `¡Hola Mundo!` en pantalla. Esta vez deberás utilizar comillas dobles (“”) al ejecutar el programa.

□

Ejercicio 4. Escribe un *script* que muestre `Este es un string "str"` en pantalla. Puedes utilizar tanto comillas dobles como simples para realizarlo.

□

Ejercicio 5. Escribe un *script* que muestre `Este es un string 'str'` en pantalla. Puedes utilizar tanto comillas dobles como simples para realizarlo.

□

Ejercicio 6. Escribe un *script* que calcule la operación $7 + 3$ y muestre el resultado en pantalla.

□

Ejercicio 7. Escribe un *script* que calcule la operación $4 \cdot (5 - 2)$ y muestre el resultado en pantalla.

⊡

Ejercicio 8. Escribe y ejecuta un *script* que muestre el mensaje `¡Hola Mundo!` en pantalla usando una variable.

⊡ La función `print` también puede usarse para mostrar el valor de una variable en pantalla. Por ejemplo, el código `print(mensaje)` mostrará el valor que esté almacenado en la variable `mensaje`.

Una solución posible para el ejercicio 8 es usar la función `print` para mostrar el valor de una variable. Podemos asignarle el valor `'¡Hola Mundo!'` a esa variable, como se puede ver en el Script 2. Ten en cuenta que si la variable `mensaje` no ha sido definida (sin valor asignado), el código `print(mensaje)` generará un error.

Código 2: Resultado del ejercicio 8 en la práctica Lab01.

```
1 # Python script
2 # Autor: Unai Perez-Goya
3 # Descripcion: Ejercicio 02
4 texto='Hola mundo!'
5 print(texto)
```

☞ *En los Scripts 8 y 2, se pueden ver algunas líneas de código que comienzan con el símbolo hash (#). Este símbolo se utiliza para agregar comentarios en el código, y el texto que aparece después de él es ignorado por el intérprete de Python, es decir, no se ejecuta. Los comentarios en este ejercicio se utilizan para introducir el script, pero se pueden colocar en cualquier parte del código.*

Ejercicio 9. Escribe y ejecuta un *script* que almacene el número `5` en la variable `num`. Luego, el valor de la variable debe mostrarse en la línea de comandos.

⊡

Este ejercicio es muy similar al ejercicio 8 y se puede resolver utilizando los métodos que se ven en el Script 2. La función `print` puede mostrar tanto texto como números en pantalla, eligiendo el formato adecuado según el tipo de información que se le pase.

Ejercicio 10. Usando una variable llamada `num` de tipo entero, escribe y ejecuta un programa que muestre el mensaje `'El valor de la variable num es X'`. Donde `X` será el valor almacenado en la variable `num`. *Ejemplo.* Si el valor de `num` es `5`, el programa debe mostrar `'El valor de la variable num es 5'` en la pantalla.

⊡

Ejercicio 11. Declara dos variables que contengan números, calcula su suma y muestra el resultado en pantalla. Si los valores asignados son `5` y `2`, el mensaje que debe aparecer en pantalla es `5 + 2 = 7`. Cada acción debe realizarse en una línea separada.

⊡

Ejercicio 12. Declara dos variables que contengan números, calcula su resta y muestra el resultado en pantalla. Si los valores asignados son `5` y `2`, el mensaje que debe aparecer en pantalla es `5 - 2 = 3`. Cada acción debe realizarse en una línea separada.

⊡

Ejercicio 13. Declara tres variables que contengan números, calcula su multiplicación y muestra el resultado en pantalla. Si los valores asignados son `5`, `2` y `3`, el mensaje que debe aparecer en pantalla es `5 * 2 * 3 = 30`. Cada acción debe realizarse en una línea separada.

⊡

Ejercicio 14. Declara dos variables que contengan números, calcula su división y muestra el cociente en pantalla. Si los valores asignados son `5` y `2`, el mensaje que debe aparecer en pantalla es `5 / 2 = 2.5`. Cada acción debe realizarse en una línea separada.

⊡

Ejercicio 15. Declara dos variables que contengan números, calcula su división y muestra el cociente y el resto en pantalla. Si los valores asignados son `5` y `2`, el mensaje que debe aparecer en pantalla es `5 / 2 = cociente 2 y resto 1`. Cada acción debe realizarse en una línea separada.

⊡

Ejercicio 16. Declara una variable que contenga un número y cambia su valor usando el operador de asignación (`=`). El programa debe mostrar todos los valores de la variable en pantalla. Cada acción debe realizarse en una línea separada, y solo se puede asignar una variable. Los mensajes que deben aparecer al ejecutar el programa son: `El valor de mi_variable es 5. El valor de mi_variable es 8.`

⊡

Ejercicio 17. Declara una variable que contenga tu edad como un número entero y muestra esa información en pantalla. Además de la edad, el programa debe mostrar el tipo de dato de la variable. Cada acción debe realizarse en una línea separada.

⊡

Ejercicio 18. Declara una variable que contenga tu altura como un número real en metros. El programa debe mostrar la altura y el tipo de dato de la variable antes de finalizar. Cada acción debe realizarse en una línea separada. ¿Cómo se representa un número real? ¿Es `5,4` un número real?

⊡

Ejercicio 19. Declara una variable que contenga un nombre y muestra su valor en pantalla. Además, el programa debe mostrar el tipo de dato de la variable. Realiza cada acción en una línea diferente.

⊡

Ejercicio 20. Declara una variable que contenga una cadena de caracteres y muestra su valor en pantalla. Además, el programa debe mostrar el tipo de dato y la longitud de la cadena. Realiza cada acción en una línea diferente. Recuerda que puedes usar la función `len()` para calcular la longitud.

⊡

Ejercicio 21. Declara dos variables que contengan cadenas de caracteres y concatena sus valores. Antes de finalizar, el programa deberá mostrar la cadena concatenada. Realiza cada acción en una línea diferente. Solo se permite usar un `print`, que debe mostrar la variable resultante.

⊡

Ejercicio 22. Escribe un programa que muestre tres mensajes en tres líneas diferentes. Al ejecutar el programa deberá aparecer lo siguiente:

```
'Esta es la primera linea'
'Esta es la segunda linea'
'Esta es la tercera linea'
```

Para este ejercicio, solo se puede utilizar un `print`.

⊡

3.1. Ejercicios de Expresiones Lógicas

Ejercicio 23. Declara dos variables que contengan números enteros y determina si los números son iguales o no. El resultado debe mostrarse como `True` o `False`. Realiza cada acción en una línea diferente.

⊡

Ejercicio 24. Declara dos variables que contengan números reales y determina si los números son iguales o no. El resultado debe mostrarse como `True` o `False`. Realiza cada acción en una línea diferente.

⊡

Ejercicio 25. Declara dos variables que contengan cadenas de caracteres y determina si las cadenas son iguales o no. El resultado debe mostrarse como `True` o `False`. Realiza cada acción en una línea diferente.

⊡

3.2. Operaciones entre booleanos

Ejercicio 26. Escribe un programa que declare dos valores booleanos y muestre el resultado del operador `and`. El resultado se mostrará como `True` o `False`. Cada acción debe aparecer en una línea diferente.

⊡

Ejercicio 27. Escribe un programa que declare dos valores booleanos y muestre el resultado del operador `or`. El resultado se mostrará como `True` o `False`. Cada acción debe aparecer en una línea diferente.

⊡

Ejercicio 28. Escribe un programa que declare dos valores booleanos y muestre el resultado del operador `not`. El resultado se mostrará como `True` o `False`. Cada acción debe aparecer en una línea diferente.

⊡

Ejercicio 29. Utilizando los ejercicios anteriores, crea todas las combinaciones posibles de dos variables booleanas. Completa la siguiente tabla con los resultados usando los operadores `and`, `or` y `not`.

1ª Variable	2ª Variable	Operador	Resultado
`True`	`True`	`and`	
`True`	`False`	`and`	
`False`	`True`	`and`	
`False`	`False`	`and`	
`True`	`True`	`or`	
`True`	`False`	`or`	
`False`	`True`	`or`	
`False`	`False`	`or`	
-	`True`	`not`	
-	`False`	`not`	

⊡

Ejercicio 30. Escribe un programa que declare dos valores booleanos y muestre si son iguales o no. El resultado se mostrará como `True` o `False`. Cada acción debe aparecer en una línea diferente. ¿Has utilizado el operador de comparación? ¿Tiene sentido comparar booleanos?

⊡

Ejercicio 31. Escribe un programa que almacene el resultado de la comparación $5 > 3$ en una variable. Antes de terminar, imprime el valor de la variable y el tipo de dato de la variable en pantalla.

⊡

Ejercicio 32. Escribe un programa que declare dos variables con números enteros. Luego, en una tercera variable, guarda si las dos primeras variables son iguales o no. Antes de terminar, imprime el valor de la tercera variable y el tipo de dato de la variable en pantalla.

⊡

Ejercicio 33. Escribe un programa que indique si el valor de la temperatura almacenada en una variable es menor o igual a 32 grados. La expresión se mostrará como `True` o `False`.

⊡

Ejercicio 34. Escribe un programa que verifique si la edad almacenada en una variable está entre 18 y 65 años (ambos inclusive). La expresión se mostrará como `True` o `False`.

⊡

Ejercicio 35. Escribe un programa que declare una variable booleana `es_estudiante` con valor `True` y que verifique si la variable `edad` es menor de 25. Imprime el resultado en pantalla antes de finalizar.

⊡

Ejercicio 36. Declara dos variables: una llamada `puntaje` que contenga un número y otra llamada `tiene_certificado` que contenga un valor booleano. Comprueba si `puntaje` es mayor o igual a 90 y si `tiene_certificado` es verdadero. Muestra si se cumple la condición para obtener un premio. Realiza cada acción en una línea diferente.

⊡

Ejercicio 37. Escribe un programa que convierta una cadena de caracteres que representa un número entero en una variable de tipo entero. Antes de finalizar, muestra el nuevo valor y el tipo de dato. ¿Qué ocurre si la cadena de caracteres representa un número real?

⊡

Ejercicio 38. Escribe un programa que convierta una cadena de caracteres que representa un número real en una variable de tipo entero. Antes de finalizar, muestra el nuevo valor y el tipo de dato. ¿Qué ocurre si la cadena de caracteres representa un número entero?

⊡

Ejercicio 39. Convierte una cadena de caracteres que representa un número real en un número entero y guarda el resultado en una variable de tipo `float`. ¿Qué ha ocurrido? ¿Qué tipo de dato tenemos al final?

⊡

Ejercicio 40. Escribe un programa que convierta una variable de tipo numérico (`int` o `float`) en una cadena de caracteres. ¿Es posible concatenar el resultado con otra cadena de caracteres?

⊡

Ejercicio 41. Escribe un programa que asigne dos números, uno real y otro entero. Antes de finalizar, muestra el valor y el tipo de dato de ambos números.

Ejemplo: Si los valores iniciales son `2` y `3.3`, los mensajes que deben mostrarse en pantalla son `'El primer número, 2, es de tipo int.'` y `'El segundo número, 3.3, es de tipo <class 'float'>.'`.

⊡

Ejercicio 42. Escribe un programa que asigne un número real a la variable `mi_variable`. Luego, duplica el valor de la variable y conviértelo en un número entero. Antes de finalizar, convierte el valor de la variable en una cadena de caracteres y muéstralo una tercera vez en pantalla.

Ejemplo: Si el valor inicial es `2.8`, los mensajes que deben mostrarse en pantalla son: `'El valor es 2.8 y es de tipo <class 'float'>.'`, `'El valor es 5 y es de tipo <class 'int'>.'`, y `'El valor es 5 y es de tipo <class 'str'>.'`.

⊡

Ejercicio 43. Escribe un programa que asigne un número entero. Luego, muestra en una sola línea los primeros 10 múltiplos de ese número.

Ejemplo: Si el valor inicial es `4`, el mensaje que debe mostrarse en pantalla es `4 8 12 16 20 24 28 32 36 40` (todos los números en la misma línea).

⊡

Ejercicio 44. Escribe y ejecuta un programa que asigne dos números, uno real y otro entero. Luego, utilizando un mensaje, muestra ambos números y sus tipos de dato, así como el resultado de la operación entre ambos números y el tipo de dato del resultado. ¿Qué tipo de dato se muestra en el resultado? ¿Qué ha ocurrido?

Ejemplo: Si los valores iniciales son 3.3 y 5, los mensajes que deben mostrarse en pantalla son: `'El valor es 3.3 y es de tipo <class 'float'>.'`, `'El valor es 5 y es de tipo <class 'int'>.'` y `'La multiplicación de los dos números es 16.5 y es de tipo <class 'float'>.'`.

⊡

Ejercicio 45. Escribe y ejecuta un programa que asigne un número a cada una de tres variables (cada una con un número diferente) y que muestre el siguiente mensaje:

Ejemplo: Si los valores de las tres variables son `2`, `3` y `4`, el mensaje mostrado en pantalla será `2 y 3 y 4`.

Nota: La palabra 'y' solo puede aparecer una vez en el código escrito.

⊡

Ejercicio 46. Escribe un programa que asigne dos números enteros a dos variables. El programa debe intercambiar los valores de las dos variables, es decir, si las variables `var1` y `var2` tienen los valores `5` y `2` respectivamente, al final deben tener los valores `var1=2` y `var2=5`. En este ejercicio, los valores de las variables (`2` y `5`) solo pueden aparecer en las dos primeras líneas.

⊡

Ejercicio 47. Escribe un programa que asigne un dígito a una variable de tipo cadena (`str`). Multiplica la variable asignada por `4` y muestra el resultado en pantalla. ¿Cuál es el resultado de la operación? ¿Qué ha ocurrido?

⊡

Ejercicio 48. Escribe un programa que, dado un valor para el radio de una circunferencia almacenado en una variable, muestre en pantalla el área y el perímetro de la circunferencia. ¿Cómo se representan las constantes?

Ejemplo: Si el valor del radio es `6.0`, el perímetro de la circunferencia será `37.699` y el área de la misma circunferencia será `113.100`.

⊡