**EJERCICIOS UD INTRO (Python)**

**Relación de Ejercicios I**

**1. Define las variables entero, real y cadena (15, 7.56, "Hola Profesor")**

--‐ Realiza las operaciones de suma, resta, multiplicación, división y módulo entre la variable **entero** y la variable **real**. ¿Se realizan todas las operaciones correctamente? ¿De qué tipo es la respuesta obtenida?

--‐ Realiza la suma entre la cadena y el número entero. ¿Qué está pasando?

--‐ Realiza la suma entre la cadena y el siguiente literal: “Aprendiendo Python”.

¿Qué diferencia observas con el anterior resultado?

--‐ Realiza la multiplicación de la variable cadena con la variable entero. ¿Qué observas?. ¿Y si lo multiplicas por la variable real?.

**2. Realiza las siguientes actividades**

¿Qué pasa si concatenas strings con números?

>>> 'PYTHON' + 3

>>> 'PYTHON' + 3.1415

Luego prueba con lo siguiente:

>>> 'PYTHON' + str(3)

>>> 'PYTHON' + str(3.1415)

¿Qué ocurre al ejecutar estas instrucciones?

>>> "3" + 3

>>>int("3")+3

>>>"3"+str(3)

¿Qué pasa cuando usas índices negativos? Ejemplo: ‘PYTHON’[-1]. Prueba omitiendo parte del rango, por ejemplo, ‘PYTHON’[3:] o ‘PYTHON’[:3]

Crear un programa que lea por teclado una cadena, y muestre la siguiente información:

* 1. Imprima los dos primeros caracteres.
  2. Imprima los tres últimos caracteres.
  3. Imprima dicha cadena cada dos caracteres. Ej.: recta debería imprimir rca
  4. Dicha cadena en sentido inverso. Ej.: hola mundo! debe imprimir !odnum aloh

**3. Crear un programa que lea por teclado una cadena y un carácter, e inserte el carácter entre cada letra de la cadena. Ej: separar y , debería devolver s,e,p,a,r,a,r**

**4. Crear un programa que lea por teclado una cadena y un carácter, y reemplace todos los espacios por el carácter. Ej: mi archivo de texto.txt y \_ debería devolver mi\_archivo\_de\_texto.txt**

**5. Crear un programa que lea por teclado una cadena y un carácter, y reemplace todos los dígitos en la cadena por el carácter. Ej: su clave es: 1540 y X debería devolver su clave es: XXXX**

**6. Crear un programa que lea por teclado una cadena y un carácter, e inserte el carácter cada 3 dígitos en la cadena. Ej. 2552552550 y . debería devolver 255.255.255.0**

**7. Crea un programa python que lea una cadena de caracteres y muestre la siguiente información:**

- La primera letra de cada palabra. Por ejemplo, si recibe Universal Serial Bus debe devolver USB.

- Dicha cadena con la primera letra de cada palabra en mayúsculas. Por ejemplo, si recibe república argentina debe devolver República Argentina.

**8. Escribe una función es\_par que tome un número x como entrada y devuelva True si x es par y False si no lo es.**

**9. Escribe una función factorial que tome como entrada un entero x y devuelva el factorial de ese número.**

**10. Escribe una función es\_primo que tome un número x como entrada y devuelva el booleano True si x es primo y False si no lo es.**

**11. Escribe una función anti\_vocal que tome como entrada un string texto y devuelva el texto sin las vocales.**

**12. Escribe una función purificar que tome una lista de números, quite todos los impares de la lista, y devuelva el resultado.**

**13. Escribe una función producto que tome como entrada una lista de enteros y devuelva el producto de todos los elementos de la lista.**

**14. Escribe una función quitar\_repetidos que tome una lista de números y quite los elementos de la lista que son iguales.**

**15. Definir una función que calcule la longitud de una lista o una cadena dada. (Es cierto que python tiene la función len() incorporada, pero escribirla por nosotros mismos resulta un muy buen ejercicio.  
  
16. Escribir una función que tome un carácter y devuelva True si es una vocal, de lo contrario devuelve False.  
  
17. Definir un histograma procedimiento() que tome una lista de números enteros e imprima un histograma en la pantalla. Ejemplo: procedimiento([4, 9, 7]) debería imprimir lo siguiente:**  
  
\*\*\*\*  
\*\*\*\*\*\*\*\*\*  
\*\*\*\*\*\*\*

**18. Escribir una función filtrar\_palabras() que tome una lista de palabras y un entero n, y devuelva las palabras que tengan más de n caracteres.**  
**19. Completa 3 diccionarios llamados Lloyd, Alice y Tyler. A cada uno asígnale las palabras clave "nombre", "tareas", "pruebas" y "exámenes". Has que la palabra clave "nombre" sea el nombre del estudiante y las demás referencias clave sean una lista vacía**

Ahora completa tu diccionario Lloyd con las notas correspondientes.

Tareas: 90, 97, 75, 92  
Pruebas: 88, 40, 94  
Notas de exámenes: 75, 90

Completa el resto de diccionarios

Ahora pon los diccionarios Lloyd, Alice y Tyler en una lista llamada estudiantes.

Imprime todos los datos de tu diccionario estudiantes como en el siguiente ejemplo.

Lloyd # nombre

[90, 97, 75, 92] # tareas

[88, 40, 94] # pruebas

[75, 90] # exámenes

Usa un bucle for para no repetir el mismo código para cada estudiante. Aunque no hay problema con usar cuatro sentencias print.

Primero, escribe la función promedio que calcula el promedio de una lista compuesta solamente por números

Escribe la función calcularPromedio la cual toma como entrada el diccionario de un estudiante y calcula su promedio ponderado. Usa como ayuda tu función promedio. Tareas corresponde al 10%, pruebas al 30% y exámenes al 60%.

Ahora escribe una función obtenerCalificacionenLetras que toma la nota como entrada y devuelve como string la calificación en letras que debe obtiene el estudiante.

Este es el corte de la calificación. (¡Asegúrate de redondear las notas de los estudiantes antes de asignar las calificaciones! Revisa la Sugerencia si necesitas ayuda con esto.)

Notas de 90 o superiores: return "A"  
If 80 <= nota < 90: return "B"  
If 70 <= nota < 80: return "C"  
If 60 <= nota < 70: return "D"  
If nota < 60: return "F"

Por último, pasa a tu función el resultado de calcularPromedio de Lloyd e imprime la calificación resultante en letras.

Escribe una función llamada calcularPromedioClase que toma la lista de estudiantes como entrada y calcula el promedio de todo el grupo.

Para esto lo que tienes que hacer es usar calcularPromedio para cada estudiante y luego calcular el promedio de esos promedios.

Por último, ejecuta tu función calcularPromedioClase para tus estudiantes para imprimir la calificación numérica Y en letras

**Relación de Ejercicios II**

**1. Realiza un módulo que haga operaciones aritméticas entre dos números (suma, resta, producto, división). A continuación, en otro fichero, importa el módulo y prueba alguna de las operaciones anteriores.**

**2. Imprime el directorio actual de trabajo.**

**3. Pide por teclado un directorio de tu disco duro y elimínalo.**

**4. Copia un directorio de tu disco duro a otra ubicación**

**5. Liste recursivamente el contenido de su directorio de usuario (C:\Users\alumno). Muestre cada ruta en una línea, y etiquete cada una como DIRECTORIO o como FICHERO.**

**6. Liste todos los archivos ejecutables (extensión '.exe') del directorio C:\Windows\system32.**

**7. Realice un listado de ficheros del directorio C:\windows\system32. Almacene en un fichero llamado "listado\_windows.txt" la ruta y tamaño de cada fichero con el formato siguiente: ruta;tamaño con una linea para cada fichero encontrado.**

**8. Use fichero "listado\_windows.txt" para determinar el número de ficheros que tienen igual extensión.**

**9. ¿Cuál es el fichero de mayor tamaño de los registrados en "listado\_windows.txt"?**

**10. Suponga una baraja de 10 cartas. Cada una tiene un número que va del 1 al 10.  
Baraje y reparta 5 para usted y 5 para el ordenador. Muestre su jugada y la del ordenador. Gana el que obtenga mayor puntuación. Imprima el ganador. En caso de empate, juegue de nuevo.   
NOTA: No puede repartir una carta que ya haya sido repartida.**

**11. Utilizando rjust() y format (es hacer el ejercicio de dos maneras distintas), realiza la siguiente tabla, resultado de la operación de un número al cuadrado y al cubo.**

**1 1 1**

**2 4 8**

**3 9 27**

**4 16 64**

**5 25 125**

**6 36 216**

**7 49 343**

**8 64 512**

**9 81 729**

**10 100 1000**

**12. Escribir un programa, llamado head que reciba un archivo y un número N e imprima las primeras N líneas del archivo.**

**13. Escribir un programa, llamado cut.py, que dado un archivo de texto, un delimitador, y una lista de campos, imprima solamente esos campos, separados por ese delimitador.**

**14. Escribir un programa, llamado wc.py que reciba un archivo, lo procese e imprima por pantalla cuántas líneas, cuantas palabras y cuántos caracteres contiene el archivo.**

**Relación de Ejercicios III**

**1. Cree una clase llamada Cuenta. Como atributo tendrá un número float llamado saldo, con una cantidad inicial de 0 euros.  
Tendrá dos métodos:**

* + ***ingresar*, con un parámetro que indica la cantidad a sumar al saldo.**
  + ***retirar*, con un parámetro que será un número float de euros a restar del saldo.**

**Cree un programa para ingresar 125.23, 503.4 y 50 euros y luego retire, 333.34 euros.  
Muestre tras cada operación, el saldo de la cuenta.**

**2. Cree una clase llamada Fichero. Esa clase tendrá un atributo llamado ruta y otro llamado texto. Implemente dos métodos:**

* + ***leer\_fichero*, que leerá el fichero dado por el atributo *ruta* y guardará su contenido en el atributo *texto*.**
  + ***mostrar\_fichero*, que imprimirá por pantalla el texto del fichero leído.**

**Use su nueva clase para leer el fichero que contiene al ejercicio 1.**

**3. Cree una clase llamada *Persona*. Contendrá el *nombre*, *DNI*, *dirección*, *teléfono* y *e-mail* (este último opcional) de un individuo.   
Por método tendrá *mostrar* que imprimirá por pantalla los datos de la persona.   
Utilice esa clase para crear una lista de personas y mostrar cada una de ellas. Los datos de cada persona están en un fichero** [***personas.txt***](http://www.iaa.es/python/ejercicios/personas.txt)**. El formato de este fichero es una línea con los datos de cada persona. Están separados por ';'. Contiene información por este orden:**

**nombre;DNI;dirección;teléfono;e-mail**

**4. Modifique la clase *Cuenta* del ejercicio 1. Haga que el atributo *saldo* quede oculto. Cuando alguien desee leer ese atributo, debe escribir un número pin. Ese valor se introducirá como parámetro de la función *leer\_saldo*, que agregará como nuevo método de la clase.  
El PIN para poder realizar la operación será 3210.   
Ahora que la variable saldo está oculta, ¿podría acceder a ella para mostrar su contenido sin conocer el PIN?. En caso afirmativo, explique cómo.**

**5. Cree una nueva clase que herede de *Cuenta* del ejercicio 1. Se llamará *Cuenta\_ahorro*. Contendrá un nuevo método oculto *\_\_avisar* que informará en el momento en el que se realice una operación (*retirar*) que dé lugar a un saldo negativo. El mensaje a mostrar será *'NUMEROS ROJOS'*.  
Modifique el método *retirar* para que no deje sacar dinero si el saldo es negativo. En ese caso mostrará el mensaje por pantalla *'NO TIENE CREDITO'*.   
Cree un programa que genere una cuenta en la que se ingresan 300 euros y se sacan, 290, 20 y 30 euros. En cada operación, muestre el saldo actual.**

**6. Cree una clase llamada *Buscador*. Sirve para buscar y reemplazar palabras dentro de un fichero. Esa clase heredará de la clase *Fichero* del ejercicio 2.  
Tendrá tres nuevos métodos:**

* + ***buscar*, que tendrá como parámetro una cadena de caracteres que buscar en el fichero. Devolverá el número de veces que se ha encontrado el texto buscado.**
  + ***reemplazar*, que tendrá como parametros dos cadenas de caracteres: la cadena a buscar y la cadena por la que reemplazar la cadena buscada.**
  + ***escribir*, que guardará el fichero en la ruta que se pase como parámetro a este nuevo método.**

**Ejecute un codigo Python que cree un objeto *Buscador*. Leerá el fichero *ejercicio1.py* y devolverá el número de veces que aparezca la palabra *self*. Posteriomente, reemplazará la palabra *self* por *this*, y guardará el texto resultante en el fichero *ejercicio1\_mod.py*. Compruebe que los cambios se han llevado a cabo en este último fichero.**

**7. Cree una clase llamada *Cliente*. Contendrá los datos de una *Persona* (ejercicio 3) y de una *cuenta de ahorro* (ejercicio 7) como atributos (herede *Cliente* de *Persona* y *Cuenta\_ahorro*). Tendrá un nuevo atributo llamado *credit* que, por defecto, será de 1000 euros.   
Tendrá un método nuevo que será *mostrar\_datos*, que imprimirá por pantalla los datos de la persona y su saldo y credito actuales.   
Cree una persona con estos datos:**

**Nombre: Mariano Fuentes**

**DNI: 45365968L**

**Direccion: C/ de las Alpujarras 2, bajo**

**Telefono: 958776566**

**Saldo: 2500**

**Credito: 1500**

**email: mfuentes@iaa.es**

**y muestre estos datos.**

**8. Cree una clase *Entidad*. Contendrá por atributos el *número de entidad*, su *director* y *clientes*. Los dos primeros datos serán proporcionados al crear la entidad. Los siguientes se leerán del fichero *personas.txt*.   
La clase *Entidad* tendrá dos métodos:**

* + ***num\_clientes*, que devolverá el número de clientes asociados a la entidad.**
  + ***balance*, que devolverá la suma de los saldos de los clientes de la entidad.**