

2K DESARROLLO DE APLICACIONES INFORMÁTICAS 30/11/2021
UT2: "Programación estructurada"

Hora comienzo: 18.50 Hora final: 20.30

Nombre y apellidos:

Calificación:

1 (4 p)	2 (6 p)

INDICACIONES GENERALES para el examen:

- La **duración** del examen es de **1:35 horas**. **Una vez terminado** el tiempo **no se podrá seguir programando** y se tendrá que **pasar al apartado de ENTREGA**.
- **Se permite** el uso de **apuntes** en papel.
- **No se permiten** dispositivos de **comunicación**.
- **Cada ejercicio** se deberá realizar **en ficheros .py diferenciados**.
- Las **funciones** deberán estar **comentadas** mediante **docstring**. (penalización = - 2 puntos)
- Las **funciones** deben tener el **nombre exacto** y **aceptar la cantidad de argumentos y retornos** propuesta. (penalización = -1 punto)
- Las **claves de los diccionarios** deben tener el **nombre exacto** y sus **valores ser del tipo indicado**. (penalización = -1 punto)

ENTREGA del examen:

- La **entrega** del examen se realizará a través de **GitHub y Classroom**.
- Se dispondrá de **5 minutos** al **comienzo** del examen para **crear un repositorio (GitHub Desktop) y proyecto (Pycharm)** donde se subirá y desarrollará el resultado de los ejercicios.
- El **nombre del repositorio** será:

Apellido_Nombre_Examen_DAPI_UT2_2M

- **Al finalizar** el examen, se dispondrá de **5 minutos** para **publicar el repositorio en GitHub** y **añadir la cuenta del profesor como colaborador** y para **realizar la entrega en classroom** al **terminar** el examen.
- Se podrá penalizar la nota **si no se cumplen los requisitos de entrega** (penalización = -1 punto)

EJERCICIO 1 (4 puntos)

Escribe una función para una compañía de gas, que convierta una lista de consumos de gas de diferentes clientes durante tres meses a otra lista de precios a cobrar.

- El **nombre** de la función será:

gas2price()

- Como **parámetros de entrada** a la función tendremos una **lista con un número indeterminado de tuplas con 3 elementos** que serán los valores de los consumos para tres meses de cada uno de los clientes, de la siguiente forma:

```
[ (Mes1_Cliente1, Mes2_Cliente1, Mes3_Cliente1),  
  (Mes1_Cliente2, Mes2_Cliente2, Mes3_Cliente2),  
  ... ,  
  (Mes1_ClienteN, Mes2_ClienteN, Mes3_ClienteN), ]
```

Los **valores** de cada uno de los **meses** será un **float**.

- Cómo **retorno** de la función deberemos obtener una **lista de precios a pagar por cada cliente**, de la siguiente forma:

```
[Precio_cliente1, Precio_cliente2, ..., Precio_clienteN]
```

Los **valores** de **precio** serán de tipo **float**.

El **cálculo** del precio se deberá **realizar llamando a otra función** dedicada a tal efecto que calculará el precio mediante la siguiente fórmula:

$$Precio = (Mes1 + Mes2 + Mes3) * 0,0615$$

- El **nombre** de la función será

price()

- Como **parámetros de entrada** a esta función tendremos las **tres variables de tipo float Mes1, Mes2 y Mes3**.
- Como **retorno** de la función tendremos la **variable de tipo float precio**.

EJERCICIO 2 (6 punto)

Escribe un programa que contenga un diccionario con la lista de prefijos telefónicos de países y que tenga una función que permita chequear e identificar si un número de teléfono está bien escrito y a qué país corresponde su prefijo. Deberá separar el número del prefijo y comprobar si el número es correcto y a que país corresponde el prefijo.

Para ello, deberás **crear el diccionario** de prefijos de la siguiente tabla:

+30	Grecia
+33	Francia
+34	España
+351	Portugal
+380	Ucrania
+39	Italia
+41	Suiza
+44	Reino Unido
+49	Alemania
+7	Rusia

- Donde **las claves** del diccionario **serán cadenas** que almacenen los **prefijos** y el **valor será otra cadena** que almacene el **país** asociado.

También deberás **crear una función** general a la que **le pasaremos el número de teléfono completo y separará el número en las dos partes.**

- El **nombre** de la función será:

phone_call()

- Como **parámetro de entrada** a la función tendremos una **cadena con el número de teléfono completo**, de la siguiente forma:

+XYZ-ABCDEFGHI dónde **+XYZ = prefijo** y **ABCDEFGHI = número**

- Cómo **retorno** de la función obtendremos **una cadena** donde tendremos **dos opciones**.
 - La primera será que el número es correcto y el país se encuentra en nuestro diccionario. Devolveremos:

"Llamando a" + <ABCDEFGHI> + (<País>)

- La segunda será o que el número no es correcto, o que no encontramos el país en nuestro diccionario. Devolveremos:

"Teléfono incorrecto o código de país no encontrado"

DENTRO DE LA FUNCIÓN, para comprobar el teléfono y el prefijo, el programa deberá llamar a dos funciones separadas:

1. La primera función deberá **comprobar que el número de teléfono es correcto en longitud (9 dígitos) y primer dígito (igual a 6)**.

Incorrecto			Correcto		
Número	Longitud	1er dígito	Número	Longitud	1er dígito
695 486 1234	10 dígitos	6	695 486 123	9 dígitos	6
754 126 987	9 dígitos	7	654 126 987	9 dígitos	6

- El **nombre** de la función será:

check_phone_number()

- Como **parámetro de entrada** a la función tendremos un **str** que contendrá el **número de teléfono**.
- Cómo **retorno** de la función tendrá que proporcionar un **booleano** con **valor False** si el **número es incorrecto** o **valor True** si el **número es correcto**.

2. La segunda función deberá **consultar el país en el diccionario**.

- El **nombre** de la función será:

check_phone_country()

- Como **parámetro de entrada** a la función tendremos un **str** que contendrá el **prefijo telefónico** en el formato de símbolo + seguido del número.

Ejemplo: "+34"

- Cómo **retorno** de la función tendrá que proporcionar un **str** con el **nombre del país asociado**.