Programas que involucran Archivos

Ejercicio 1

Escribir una función que pida un número entero entre 1 y 10 y guarde en un fichero con el nombre tabla-n.txt la tabla de multiplicar de ese número, done n es el número introducido.

```python
numero = raw_input("Introduce un numero del 1 al 10: ")

# w (write)
# r (read)
f = open("tabla-" + numero + ".txt", 'w')

for i in range(1, 11):
    # .read() --> Escribir información en el fichero.
    f.write(str(i) + " x " + numero + " = " + str(i * int(numero)) + "\n")
f.close()
```

Ejercicio 2

Escribir una función que pida un número entero entre 1 y 10, lea el fichero tabla-n.txt con la tabla de multiplicar de ese número, done n es el número introducido, y la muestre por pantalla. Si el fichero no existe debe mostrar un mensaje por pantalla informando de ello.

Mode	Description
`'r'`	Open a file for reading. (default)
`'w'`	Open a file for writing. Creates a new file if it does not exist or truncates the file if it exists.
`'x'`	Open a file for exclusive creation. If the file already exists, the operation fails.
`'a'`	Open for appending at the end of the file without truncating it. Creates a new file if it does not exist.
`'t'`	Open in text mode. (default)
`'b'`	Open in binary mode.
`'+'`	Open a file for updating (reading and writing)

```python
numero = raw_input("Introduce un numero del 1 al 10: ")

# w (write)
# r (read)
try:
    f = open("tabla-" + numero + ".txt", 'r')
except:
    print("El fichero no existe")
else:
    # .read() --> Lee la información del fichero.
    print(f.read())
```

Ejercicio 3

Escribir una función que pida dos números n y m entre 1 y 10, lea el fichero tabla-n.txt con la tabla de multiplicar de ese número, y muestre por pantalla la línea m del fichero. Si el fichero no existe debe mostrar un mensaje por pantalla informando de ello.

```python
'''
Ejercicio 3
Escribir una función que pida dos números n y m entre 1 y 10,
lea el fichero tabla-n.txt con la tabla de multiplicar de ese número,
y muestre por pantalla la línea m del fichero. Si el fichero no existe
debe mostrar un mensaje por pantalla informando de ello.
'''

numero_n = input("Introduce un numero del 1 al 10: ")
numero_m = int(input("Introduce un numero del 1 al 10: "))

# w (write)
# r (read)
try:
    f = open("tabla-" + numero_n + ".txt", 'r')
except:
    print("El fichero no existe")
else:
    # .readlines() --> Lee la información del fichero y lo guarda en un arreglo, separandolo por saltos de linea.
    lineas = f.readlines()
    print(lineas[numero_m-1])
```

Ejercicio 4

Escribir un programa que acceda a un fichero de internet mediante su url y muestre por pantalla el número de palabras que contiene.

```python
from urllib import request
import ssl

def palabras(url):
    context = ssl._create_unverified_context()
    try:
        file = request.urlopen(url, context=context)
    except:
        return("LA URL no existe")
    else:
        content = file.read()
        return len(content.split())


print(palabras("https://www.gutenberg.org/files/2000/2000-0.txt"))
print(palabras("https://no-existe.txt"))
```

Ejercicio 5

Escribir un programa que abra el fichero con información sobre el PIB per cápita de los países de la Unión Europea (url:https://ec.europa.eu/eurostat/estat-navtree-portlet-prod/BulkDownloadListing?file=data/sdg_08_10.tsv.gz&unzip=true), pregunte por las iniciales de un país y muestre el PIB per cápita de ese país de todos los años disponibles.

```python
from urllib import request
import ssl

def palabras(url, user_pais):
    context = ssl._create_unverified_context()
    try:
        file = request.urlopen(url, context=context)
    except:
        return("LA URL no existe")
    else:

        contenido = file.read().decode('utf-8').split("\n")
        contenido.pop(0)

        paises = { }
        for linea in contenido:
            geolocalizacion = linea.split("\t")
            pais = geolocalizacion[0].split(",")
            paises.setdefault(pais[2], geolocalizacion[1:])

        año = 2000
        if user_pais in paises:
            texto = ""
            for y in paises[user_pais]:
                texto += str(año) + " - " + y + "\n"
                año += 1
            return texto
        else:
            return ("Pais seleccionado, no existe.")
```

```python
condicional = True
while condicional:
    user_pais = input("Escribe las iniciales de un pais europeo: ").upper()
    if user_pais == "SALIR":
        condicional = False
    else:
        print(palabras("https://ec.europa.eu/eurostat/estat-navtree-portlet-prod/BulkDownloadListing?file=data/sdg_08_10.tsv.gz&unzip=true", us
```

Ejercicio 6

Escribir un programa para gestionar un listín telefónico con los nombres y los teléfonos de los clientes de una empresa. El programa incorporar funciones crear el fichero con el listín si no existe, para consultar el teléfono de un cliente, añadir el teléfono de un nuevo cliente y eliminar el teléfono de un cliente. El listín debe estar guardado en el fichero de texto listin.txt donde el nombre del cliente y su teléfono deben aparecer separados en una línea distinta.

```python
file_name = "listin.txt"

def agregar_numero():
    nombre = input("Escribir su nombre: ")
    telefono = int(input("Escribir su telefono: "))

    f = open(file_name, "a")
    f.write(nombre + "," + str(telefono) + "\n")
    f.close()

def eliminar_numero():
    user_nombre = input("Escribir el nombre que deseas eliminar: ")
    f = open(file_name, "r")
    lineas = f.readlines()
    f.close()

    existe = False
    for x in range(0, len(lineas)):
        nombre  = lineas[x].split(",")
        if nombre[0] == user_nombre:
            lineas.pop(x)
            break
        else:
            existe = True

    if existe:
        print("El contacto no existe.")

    f = open(file_name, "w")
    for linea in lineas:
        f.write(linea)
```

```python
def listar_numero():
    f = open(file_name, "r")
    lineas = f.readlines()
    f.close()
    texto = ""
    for linea in lineas:
        texto += linea.replace(",", " - ")
    print(texto)


condicional = True
while condicional:
    opcion = int(input("1) Agregar contacto \n2) Eliminar contacto \n3) Listar contactos. \n4) Salir \n\nEscribe la opcion indicada: "))
    if opcion == 1:
        agregar_numero()
    elif opcion == 2:
        eliminar_numero()
    elif opcion == 3:
        listar_numero()
    elif opcion == 4:
        condicional = False
    else:
        print("Opcion Invalida")
```

Ejercicio 7

El fichero cotizacion.csv contiene las cotizaciones de las empresas del IBEX35 con las siguientes columnas: Nombre (nombre de la empresa), Final (precio de la acción al cierre de bolsa), Máximo (precio máximo de la acción durante la jornada), Mínimo (precio mínimo de la acción durante la jornada), Volumen (Volumen al cierre de bolsa), Efectivo (capitalización al cierre en miles de euros).

Construir una función reciba el fichero de cotizaciones y devuelva un diccionario con los datos del fichero por columnas.

Construir una función que reciba el diccionario devuelto por la función anterior y cree un fichero en formato csv con el mínimo, el máximo y la media de dada columna.

```python
def preprocesado(ruta):
    try:
        f = open(ruta, 'r')
    except:
        print("El fichero no existe")
        return

    lineas = f.readlines()
    f.close()

    claves = lineas[0].replace("\n", "").split(";")

    valores = lineas[1:]

    cotizacion = {}
    for valor in valores:
        valor_linea = valor.replace("\n", "").split(";")

        dato_cotizacion = {}
        for x in range(1, 6):
            dato_cotizacion.setdefault(claves[x], valor_linea[x])

        cotizacion.setdefault(valor_linea[0], dato_cotizacion)

    return(cotizacion)
```

```python
def resumen(ruta, cotizaciones):
    f = open(ruta, 'w')
    f.write("Nombre;Final;Máximo;Mínimo\n")

    for cotizacion in cotizaciones:
        texto = cotizacion + ";" + cotizaciones[cotizacion]["Final"] + ";" + cotizaciones[cotizacion]["Máximo"] + ";" + cotizaciones[cotizacion]["Mínimo"]
        f.write(texto)

    f.close()

cotizaciones = preprocesado('cotizacion.csv')
resumen('resumen_cotizaciones.csv', cotizaciones)
```

Ejercicio 8

El fichero calificaciones.csv contiene las calificaciones de un curso. Durante el curso se realizaron dos exámenes parciales de teoría y un examen de prácticas. Los alumnos que tuvieron menos de 4 en alguno de estos exámenes pudieron repetirlo en la al final del curso (convocatoria ordinaria). Escribir un programa que contenga las siguientes funciones:

Una función que reciba el fichero de calificaciones y devuelva una lista de diccionarios, donde cada diccionario contiene la información de los exámenes y la asistencia de un alumno. La lista tiene que estar ordenada por apellidos.

Una función que reciba una lista de diccionarios como la que devuelve la función anterior y añada a cada diccionario un nuevo par con la nota final del curso. El peso de cada parcial de teoría en la nota final es de un 30% mientras que el peso del examen de prácticas es de un 40%.

Una función que reciba una lista de diccionarios como la que devuelve la función anterior y devuelva dos listas, una con los alumnos aprobados y otra con los alumnos suspensos. Para aprobar el curso, la asistencia tiene que ser mayor o igual que el 75%, la nota de los exámenes parciales y de prácticas mayor o igual que 4 y la nota final mayor o igual que 5.

MircoServicios

Manejadores —>

Kubernetes (k8s)
Docker Swarm

Docker

Envia Mensajes

Feed

Backend

FrontEnd

FullStack

Reactions

Comentarios

DB Connector

Single Responsabilty

Dev2

Pre-Prod

Desarrollo —— QA —— Production

Local

Análisis de código estatico
Compilacion
Análisis de código dinámico (Unit Test)
Testing Automático
Testing Manual

DNS - Domain Name Service

Dirección IP
(Numero de Casa)