Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Lenguajes Formales y de Programacion
Sección A-
Inga. Vivian Damaris Campos Gonzales
Aux. Mario Josué Solis Solórzano
Primer semestre 2023

MANUAL TECNICO

APLICACIÓN DE LECTURA DE ARCHIVOS DE TEXTO

Nombre:	Eduardo Misael Lopez Avila	Registro académico:	202100147
Fecha:	24/02/2022	Sección:	A-

1. Descripción

Se requiere el desarrollo de una aplicación que permita la lectura de un archivo de texto plano que contiene diferentes listas de valores separados por punto y coma. Estos valores contienen información como nombre de una película, actores que participan en ella, año de estreno y género de esta. Lo que se solicita realizar con la información contenida en cada línea de texto es el ordenamiento de información para una mejor forma de lectura y comprensión, por ello, se solicita a los estudiantes de Lenguajes Formales y de Programación, la implementación de un software de tipo consola, que permita presentar de una forma más simple la información relacionada con los datos brindados.

2. Aplicación de lectura de archivos de texto.

2.1. Librerías en uso.

Se utilizan las siguientes librerías:

Msvcrt: Esta librería sirve para solicitar el uso del teclado para continuar con la instrucción.

Csv: Esta librería sirve para la búsqueda de archivo con una estructura de datos definida para que pueda ser leído por el sistema.

Os: Esta librería sirve para verificar dentro de la aplicación mediante comandos del sistema si el archivo a cargar existe en la dirección correcta.

Graphviz: Esta librería nos sirve para generar un grafo con una estructura definida a partir de la información cargada al sistema durante la ejecución.

2.2. Vista general de la aplicación

Se inicializa la aplicación.

Cada línea de texto se imprime mediante un print(), además de un getch para solicitar al usuario que presione un botón para continuar.

```
print("LENGUAJES FORMALES Y DE PROGRAMACION")
print("SECCION: A-")
```

```
print("LENGUAJES FORMALES Y DE PROGRAMACION")
print("SECCION: A-")
print("CARNE:202100147")
print("NOMBRE: EDUARDO MISAEL LOPEZ AVILA")
print("Presione cualquier tecla para continuar...")
msvcrt.getch()
```

```
LENGUAJES FORMALES Y DE PROGRAMACION
SECCION: A-
CARNE:202100147
NOMBRE: EDUARDO MISAEL LOPEZ AVILA
Presione cualquier tecla para continuar...
```

```
--------MENU PRINCIPAL--------
1.CARGAR ARCHIVO DE ENTRADA
2.GESTIONAR PELICULAS
3.FILTRADO
4.CREAR GRAFICA
5.SALIR
------------------------------
Seleccione una opcion:
```

2.3. Cargar archivo de entrada

Se debe ingresar una carga de datos de un archivo en formato .LFP, el cual contiene los datos con una determinada estructura para ordenar. Para esto, se debe ingresar la opción 1 “Cargar archivo de entrada” el imprimirá en consola una instrucción para ingresar la dirección.Si el archivo contiene datos correctos y la dirección es correcta, los datos se guardan.

Si el archivo contiene datos incorrectos o la dirección es incorrecta, los datos no se guardan:

Si el archivo contiene datos repetidos, estos no se agregan:

```
print("INGRESE LA DIRECCION DEl ARCHIVO")
directorioA= input()
if os.path.exists(directorioA):
    csvfile = open(directorioA, 'r',encoding='utf-8')
    reader = csv.reader(csvfile, delimiter=';')

    for row in reader:
        nombrePelicula = row[0].strip()
        nombreActor = row[1].strip()
        anio=row[2].strip()
        genero=row[3].strip()
        actores=nombreActor.split(',')
        limpio=[]
        for i in actores:
            limpio.append(i.strip())
        for n in people:
            if nombrePelicula==n.nombrePelicula and limpio==n.actores and anio==n.anio and genero==n.genero:
                print("La película ",nombrePelicula," ya se encuentra en el registro.")
                break
        else:
            newInfo = informacion(nombrePelicula, limpio, anio, genero)
            people.append(newInfo)
    print("-------------------------")
    print("-----ARCHIVO CARGADO-----")
    print("-------------------------")
```

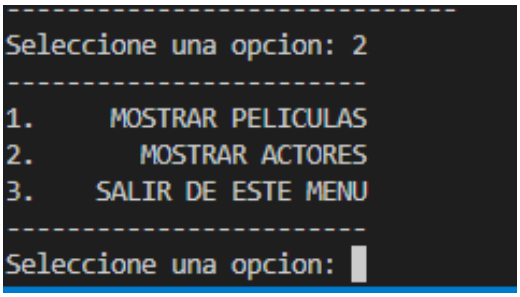
Al inicio se define las listas people y peliculas, estas almacenaran la informacion que se carga en el codigo

Dentro de directiorioA se almacena la direccion, seguido se verifica mediante un if si existe en el directorio, si existe, se utiliza funciones de csv para abrir el archivo encontrado y leerlo con ciertos atributos.
Si los datos estan en la estructura correcta, se proceden a ingresar dentro de las listas, se define un for que recorre cada linea dentro del archivo, tomando el dato delimitado por ; según la estructura de datos, se agregan a su respectiva parte de su lista, ademas se limpian “,” y espacios vacios al inicio y al final, dejando asi el

ingreso de datos limpio y ordenado dentro de las listas. Tambien se verifica si una pelicula ya existe dentro de las listas, si esta existe, no se agrega.

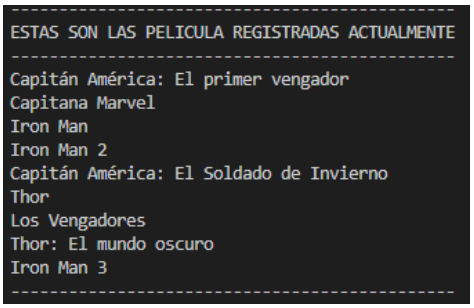
2.4. Gestionar películas

Se muestra un menú con distintas opciones para la gestión de peliculas.



2.4.1. Mostrar películas

Se muestra la lista de las películas que se han guardado en el sistema

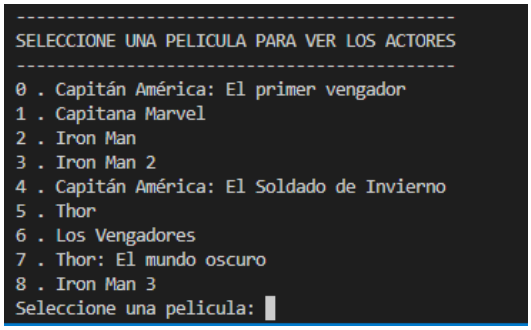


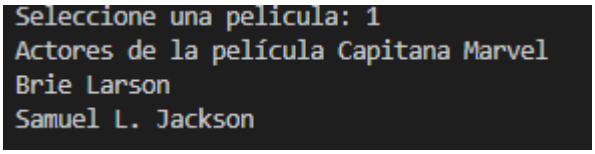

```
print("-----------------------------------------------")
print("ESTAS SON LAS PELICULA REGISTRADAS ACTUALMENTE")
print("-----------------------------------------------")
for person in people:
    person.print_nombrePelicula()
print("-----------------------------------------------")
print("Presione cualquier tecla para continuar...")
msvcrt.getch()
```

Dentro de este apartado, únicamente se inicializa un for para imprimir la lista completa de películas según su nombre, recorriendo cada item dentro de la lista.

2.4.2. Mostrar actores

Se muestra la lista de actores según la película en la que participan






```
print("-------------------------------------------")
print("SELECCIONE UNA PELICULA PARA VER LOS ACTORES")
print("-------------------------------------------")
# Mostrar la lista de películas registradas
contador=0
for person in people:
    peliculas.append(person)
    print(contador,".",person.nombrePelicula)
    contador +=1
peliculaSeleccionada = int(input("Seleccione una pelicula: "))# Preguntar al usuario de qué película quiere ver los actores
if int(peliculaSeleccionada)>=contador:
    print("SELECCIONE UNA OPCION VALIDA")
    peliculas.clear()
    contador=0
    break
else:
    actoresMostrar = peliculas[peliculaSeleccionada].actores# Obtener la lista de actores de la película seleccionada
    print("Actores de la película", peliculas[peliculaSeleccionada].nombrePelicula)# Imprimir la lista de actores
    for actor in actoresMostrar:
        print(actor)
    print("-------------------------------------------")
    peliculas.clear()
    contador=0
    print("Presione cualquier tecla para continuar...")
msvcrt.getch()
```

En este apartado, se recorre (con el primer for)la lista de películas y se imprime en la consola para que el usuario vea una película y la pueda seleccionar enumerándolas con un contador, se selecciona la película según el numero a partir de un input(), si se selecciono un numero correcto, seguirá la ejecución, sino, se detendrá.

Después de seleccionar el numero, en actoresMostrar se define mediante peliculaSeleccionada (que es el input) la película elegida y se comienza a iterar con un for los actores que se deban mostrar de esa película, imprimiéndolos en la consola.

2.4.3. Salir de este menú

Opción para salir del menú y regresar al anterior

```
------------------------
Seleccione una opcion: 3
REGRESANDO A MENÚ PRINCIPAL
--------MENU PRINCIPAL--------
1.CARGAR ARCHIVO DE ENTRADA
2.GESTIONAR PELICULAS
3.FILTRADO
4.CREAR GRAFICA
5.SALIR
----------------------------
Seleccione una opcion:
```

```
case "3":
    print("REGRESANDO A MENÚ PRINCIPAL")
    break
```

Mediante un break, se termine el while que mantiene el menú y se regresa al menú principal

2.5. Filtrado

Menú con opciones según el filtrado que requiera el usuario

```
Seleccione una opcion: 3
OPCION PARA FILTRAR
--------FILTRADO--------
1. FILTRADO POR ACTOR
2. FILTRADO POR AÑO
3. FILTRADO POR GÉNERO
4. REGRESAR A MENÚ PRINCIPAL
------------------------
Seleccione una opción:
```

2.5.1. Filtrado por Actores

Se muestra la lista de los actores y al seleccionarlos se muestra la lista de las películas en las que el actor participa.

```
Seleccione una opción: 1
FILTRADO POR ACTOR
--------------------------
ACTORES REGISTRADOS:
Natalie Portman
Robert Downey Jr.
Adrianne Palicki
Brie Larson
```

```
------------------------
Seleccione un actor: Chris Evans
--------------------------
RESULTADO DEL FILTRADO:
-------- Pelicula: 1 --------
Capitán América: El primer vengador
Accion
2011
--------------------------
-------- Pelicula: 2 --------
Capitán América: El Soldado de Invierno
Accion
2014
--------------------------
-------- Pelicula: 3 --------
Los Vengadores
Accion
2012
--------------------------
```

```python
if filtroOpcion == "1":
    print("FILTRADO POR ACTOR")
    print("------------------------")
    # Mostrar la lista de actores registrados
    actores_registrados = set()
    for person in people:
        actores_registrados.update(person.actores)
    print("ACTORES REGISTRADOS:")
    for actor in actores_registrados:
        print(actor)
    print("------------------------")
    # Preguntar al usuario qué actor quiere filtrar
    actor_filtro = input("Seleccione un actor: ")
    # Buscar las películas en las que participa el actor
    peliculas_filtradas = []
    for person in people:
        if actor_filtro in person.actores:
            peliculas_filtradas.append(person)
    # Imprimir el resultado del filtrado
    print("--------------------------")
    print("RESULTADO DEL FILTRADO:")
    contadorActores=1
    if len(peliculas_filtradas) == 0:
        print("No se encontraron películas con el actor seleccionado.")
    else:
        for pelicula in peliculas_filtradas:
            print("--------","Pelicula:",contadorActores,"--------")
            pelicula.print_nombrePelicula(), pelicula.print_genero(), pelicula.print_anio()
            print("--------------------------")
            contadorActores +=1
    print("Presione cualquier tecla para continuar...")
    msvcrt.getch()
```

Se define una lista de objetos iterables en actores_registrados = set(), dentro del cual se agregan con un for, la lista de actores de la lista principal (people), de esta manera, el siguiente for, se lee los ítems dentro de esta lista de objetos y se muestran los actores.

Después, se pide seleccionar un actor mediante un input(), se define una nueva lista que almacenara los datos del actor escogido mediante un for, y se imprimirá en consola posteriormente mediante un for, validando que el usuario haya escogido un actor correctamente.

2.5.2. Filtrado por Año

Se muestra la lista de los años de las películas en el registro y al seleccionar un año se muestra la lista de las películas que se estrenaron en ese año.

```
Seleccione una opción: 2
FILTRADO POR AÑO
-------------------------
AÑOS REGISTRADOS:
2011
2019
2008
```

```
-------------------------
Ingrese el año a filtrar: 2013
-------------------------
RESULTADO DEL FILTRADO:
-------- Pelicula: 1 --------
Thor: El mundo oscuro
Accion
-------------------------
-------- Pelicula: 2 --------
Iron Man 3
Accion
-------------------------
Presione cualquier tecla para continuar...
```

```python
print("FILTRADO POR AÑO")
print("-------------------------")
# Mostrar la lista de actores registrados
anios_registrados = set()
for person in people:
    anios_registrados.update([person.anio])
print("AÑOS REGISTRADOS:")
for anioos in anios_registrados:
    print(anioos)
print("-------------------------")
# Preguntar al usuario el año a filtrar
anio_filtro = input("Ingrese el año a filtrar: ")
# Buscar las películas del año seleccionado
peliculas_filtradas = []
for person in people:
    if person.anio == anio_filtro:
        peliculas_filtradas.append(person)
# Imprimir el resultado del filtrado
print("-------------------------")
print("RESULTADO DEL FILTRADO:")
contadorAños=1
if len(peliculas_filtradas) == 0:
    print("No se encontraron películas para el año seleccionado.")
else:
    for pelicula in peliculas_filtradas:
        print("--------","Pelicula:",contadorAños,"--------")
        pelicula.print_nombrePelicula(), pelicula.print_genero()
        print("-------------------------")
        contadorAños+=1
print("Presione cualquier tecla para continuar...")
msvcrt.getch()
```

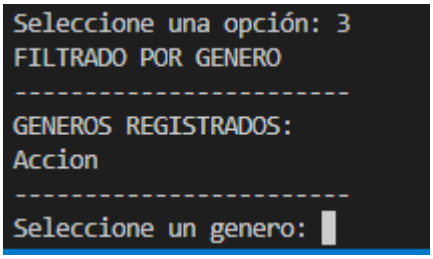
Se define la lista de objetos iterable años_registrados=set(), se le inserta mediante un for los valores de año por películas existentes en la lista people, luego los imprime al usuario mediante un for.

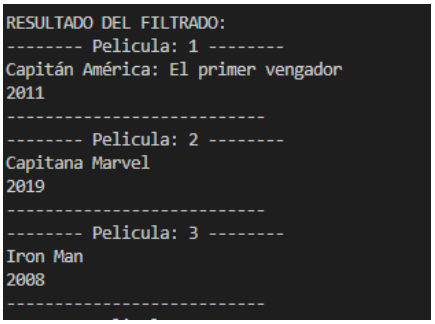
Después se define un input() que indicara que año queremos revisar, se ingresa el año y se almacena en la variable. Luego, se define una lista películas_filtradas = [] en donde ira almacenado los datos de las películas que fueron estrenadas en el año escogido, esto mediante un for que itera cada ítem dentro de la lista people en el subíndice anio.

Si el año se ingresa mal o no hay películas para ese año, no se mostrara ninguna película.

2.5.3. Filtrado por Genero

Se muestra una lista de los géneros de las películas en el registro y al seleccionar un genero se muestra la lista de películas de ese mismo genero






```
elif filtroOpcion == "3":
    print("FILTRADO POR GENERO")
    print("------------------------")
    # Mostrar la lista de genero registrados
    generos_registrados = set()
    for person in people:
        generos_registrados.update([person.genero])
    print("GENEROS REGISTRADOS:")
    for genre in generos_registrados:
        print(genre)
    print("------------------------")
    # Preguntar al usuario qué genero quiere filtrar
    genero_filtro = input("Seleccione un genero: ")
    # Buscar las películas en las que participa el genero
    genero_filtradas = []
    for person in people:
        if genero_filtro in person.genero:
            genero_filtradas.append(person)
    # Imprimir el resultado del filtrado
    print("--------------------------")
    print("RESULTADO DEL FILTRADO:")
    contadorGeneros=1
    if len(genero_filtradas) == 0:
        print("No se encontraron películas con el genero seleccionado.")
    else:
        for pelicula in genero_filtradas:
            print("--------","Pelicula:",contadorGeneros,"--------")
            pelicula.print_nombrePelicula(), pelicula.print_anio()
            print("--------------------------")
            contadorGeneros +=1
    print("Presione cualquier tecla para continuar...")
    msvcrt.getch()
```

Se define la lista de objetos iterables géneros_registrados = set(), se almacenan los datos de los géneros del subindice people mediante un for, y se muestran al usuario mediante un for que imprime los géneros que están registrados.

Después, el usuario escribe mediante un input() las películas del genero que quiere ver, se almacena y se guarda en la variable genero_filtro.

Finalmente, se define una lista genero_filtradas, en la cual se almacenan las películas según el genero escogido mediante la variable genero_filtro mediante un for, y con otro for se imprime en consola las películas de ese genero.

S

2.6. Crear grafica

Se crea una grafica que une mediante flechas la relación que tiene cada película con sus respectivos actores.

```
------------------------------
Seleccione una opcion: 4
--------------------------
GRAFICA GENERADA CON EXITO
--------------------------
```

```
def graphRelations(people):
    grafo = graphviz.Digraph('ejemplo', filename="D:\\USAC\\2023\\Primer Semestre\\LFP\\LAB-LFP\\Practica 1\\[LFP]Practica1_202100147\\grafo")
    grafo.attr(rankdir="LR", ranksep="3", nodestep="1")

    # Creamos los nodos para las películas y los actores
    for person in people:
        grafo.node(str(person.nombrePelicula).replace(':',''), f'''<
            <TABLE BORDER="0" CELLBORDER="1" CELLSPACING="0" CELLPADDING="4">
                <TR>
                    <TD COLSPAN="2" bgcolor="#507bef"><FONT COLOR="black">{person.nombrePelicula}</FONT></TD>
                </TR>
                <TR>
                    <TD><FONT COLOR="black">{person.anio}</FONT></TD>
                    <TD><FONT COLOR="black">{person.genero}</FONT></TD>
                </TR>
            </TABLE>>''', shape="none")
        for actor in person.actores:
            grafo.node(str(actor), shape="box", style="filled", color="#36d6b2", fontcolor="black")

            # Agregamos una conexión desde el actor hacia la película
            grafo.edge(str(person.nombrePelicula).replace(':','')+":e",str(actor))

    grafo.view()
print("--------------------------")
print("GRAFICA GENERADA CON EXITO")
print("--------------------------")
graphRelations(people)
```

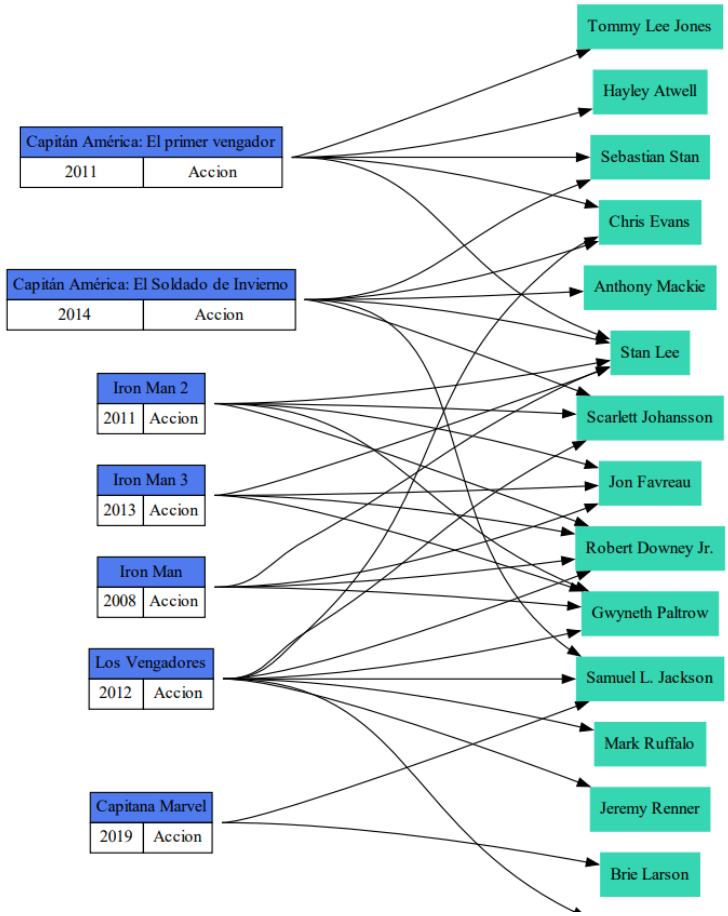
Se define la función graphRelations dentro del switch-case option5, esta función nos sirve para generar una grafica tipo grafo, que une mediante flechas la relación entre las películas y los actores.

Primero. Se define el grafo (grafo) y se indica que es de graphviz.Digraph, la librería proporciona sus funciones de generador, se define el ejemplo y el nombre del archivo (se coloca la dirección y nombre del archivo) y también que tipo de archivo. Para generar la estructura del grafo, se define grafo.node() que almacena los atributos que tendrá la gráfica, se define str(person.nombrePelicula).replace() que seria la etiqueta del nodo, junto a replace () que nos sirve para validar los : dentro del nombrePelicula, evitando errores comunes de texto o enlazamiento. Se define la estructura de la tabla que contiene el nombre de la película, el año y el genero de la misma.

Luego, se define mediante un for, un nodo por cada actor registrado en la lista people, nuevamente se define la característica de cada nodo y se almacena.

Finalmente, se define la unión de cada nodo con Edge (grafo.edge), que une con una flecha las películas a los actores dependiendo si un actor participa o no en esa película, esta se une con cada etiqueta del identificador de películas y actores.

EJEMPLO DE LA GRAFICA GENERADA EN PDF



2.7. Salir

Opción para terminar la ejecución de la aplicación.

```
--------MENU PRINCIPAL--------
1.CARGAR ARCHIVO DE ENTRADA
2.GESTIONAR PELICULAS
3.FILTRADO
4.CREAR GRAFICA
5.SALIR
------------------------------
Seleccione una opcion: 5
--------------------------
SALIENDO, TE TENGA BUEN DIA
--------------------------
```

```
case "5":
    print("--------------------------")
    print("SALIENDO, TE TENGA BUEN DIA")
    print("--------------------------")
    break
```

Se define la utilidad del case 5 del menú principal, esta únicamente termina la ejecución del while principal, terminando así la ejecución de la aplicación en Python