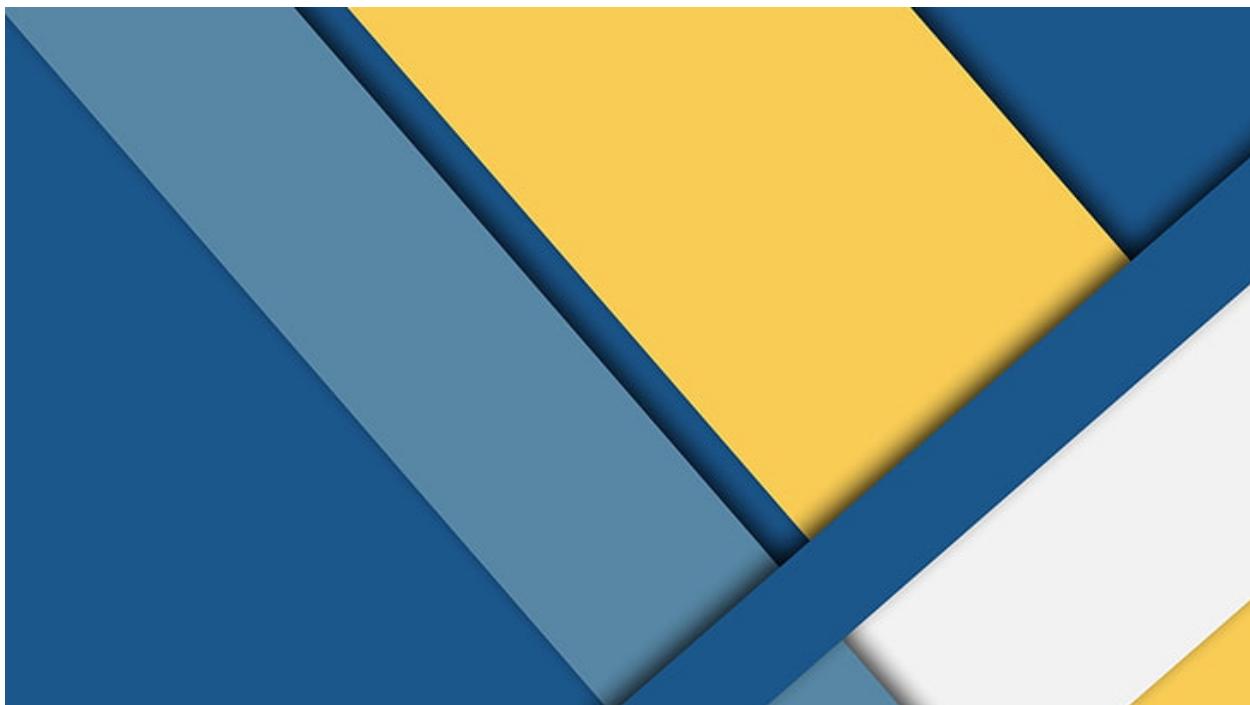


LENGUAJES FORMALES Y DE PROGRAMACIÓN B-

MANUAL TÉCNICO

PRÁCTICA 1



NOMBRE: ANA LUCIA FLETES ORDÓÑEZ
CARNÉ: 202010003

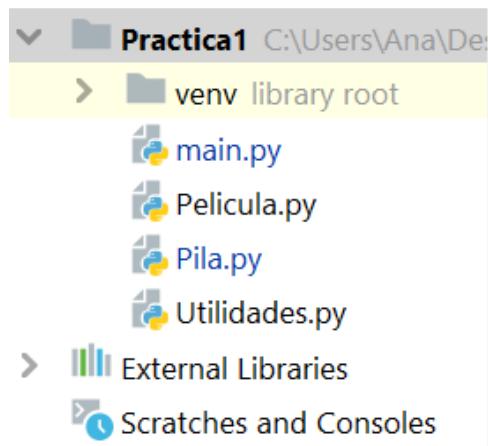
INTRODUCCIÓN

El siguiente manual técnico explica a detalle el funcionamiento del programa realizado en Python para la Práctica #1 del curso de Lenguajes Formales y de Programación. Este consiste en una aplicación de administración de películas, en donde se permite la lectura de archivos de texto con información respecto a las películas, como lo son el nombre, actores, año y género. Asimismo, debe hacer posible la visualización de los distintos datos de las películas almacenadas; el filtrado de las películas de acuerdo a los actores, año y género; y la elaboración de un gráfico (usando Graphviz) en donde se relacionen las películas con los actores.

ÍNDICE

ESTRUCTURA DEL PROYECTO.....	4
CLASE PELICULA.....	5
ATRIBUTOS.....	5
MÉTODOS.....	5
CLASE PILA.....	6
ATRIBUTOS.....	6
MÉTODOS.....	6
CLASE UTILIDADES.....	8
ATRIBUTOS.....	8
MÉTODOS.....	8
CLASE MAIN.....	9
PANTALLA INICIAL.....	9
MENÚ PRINCIPAL.....	9
CARGAR ARCHIVOS DE ENTRADA.....	10
GESTIONAR PELÍCULAS.....	10
FILTRADO.....	12
GRÁFICA.....	14
SALIR.....	16

ESTRUCTURA DEL PROYECTO



El proyecto consta de cuatro clases, las cuales se detallarán a profundidad en las siguientes páginas:

- main
- Pelicula
- Pila
- Utilidades

CLASE PELICULA

Tiene un constructor con parámetros, para que al momento de crear una instancia (objeto) de la misma, se pueda inicializar con los valores ingresados. La clase Pelicula tiene el objetivo de guardar la información de las películas que sean cargadas al sistema, y permitir la obtención o recuperación de estos datos.

ATRIBUTOS

Los atributos de esta clase son: nombre, actores, anio y genero, donde actores es una lista de elementos.

MÉTODOS

Los métodos pertenecientes a la clase Pelicula tienen el objetivo de retornar los atributos de la misma, siendo un método por atributo: obtenerNombre, obtenerActores, obtenerAnio y obtenerGenero.

```
1      class Pelicula:
2          def __init__(self, nombre, actores, anio, genero):
3              self.nombre = nombre
4              self.actores = actores
5              self.anio = anio
6              self.genero = genero
7
8          def obtenerNombre(self):
9              return self.nombre
10
11         def obtenerActores(self):
12             return self.actores
13
14         def obtenerAnio(self):
15             return self.anio
16
17         def obtenerGenero(self):
18             return self.genero
```

CLASE PILA

Tiene un constructor sin parámetros, ya que al momento de crear una instancia de esta clase no se inicializará el valor del atributo.

El fin de la clase Pila es almacenar objetos de tipo Pelicula dentro de una lista y administrar los mismos a través de diferentes métodos.

ATRIBUTOS

Esta clase cuenta con un solo atributo, el cual al inicio es una lista vacía con identificador 'peliculas'.

MÉTODOS

- **push(self, pelicula):** agrega un objeto tipo Pelicula en la lista 'peliculas'.
- **retornarPeliculas(self):** retorna la lista de películas.
- **eliminarPelicula(self, posicion):** elimina una película de la lista 'peliculas' en una posición dada.
- **devolverPelicula(self, posicion):** regresa el objeto Pelicula almacenado en la posición dada en la lista 'peliculas'.
- **devolverActoresPeliculas(self):** devuelve una lista con los nombres de los actores de todas las películas agregadas a la lista 'peliculas', sin repetir ninguno de estos.
- **tamanio(self):** retorna el tamaño de la lista 'peliculas' (es decir, la cantidad de elementos que este contiene).

```
1  class Pila:  
2      AnnFlets  
3      def __init__(self):  
4          self.peliculas = []  
5      AnnFlets  
6      def push(self, pelicula):  
7          self.peliculas.append(pelicula)
```

```
8     def retornarPelículas(self):
9         return self.películas
10
11    @ AnnFlets
12    def eliminarPelícula(self, posición):
13        self.películas.pop(posición)
14
15    @ AnnFlets
16    def devolverPelícula(self, posición):
17        return self.películas[posición]
18
19    def devolverActoresPelículas(self):
20        actores = []
21        for película in self.películas:
22            for actor in película.obtenerActores():
23                if len(actores) != 0:
24                    contador = 0
25                    actorVerificar = actor
26                    for actorPeli in actores:
27                        if actorVerificar == actorPeli:
28                            contador = contador + 1
29                    if contador == 0:
30                        actores.append(actor)
31                    else:
32                        actores.append(actor)
33
34    @ AnnFlets
35    def tamaño(self):
36        return len(self.películas)
```

CLASE UTILIDADES

Esta clase contiene métodos que realizarán alguna funcionalidad extra durante la ejecución del programa.

ATRIBUTOS

No tiene atributos.

MÉTODOS

Los métodos de esta clase tienen como finalidad el comprobar si un dato es un valor numérico o no.

- **comprobarEntero(self, valor):** devuelve True (verdadero) si el valor recibido es un número entero y False (falso) en el caso contrario.
- **comprobarDecimal(self, valor):** retorna True (verdadero) si el valor recibido es un número con decimales y False (falso) en el caso contrario.

```
1  class Utilidades:
2
3      def __init__(self):
4          pass
5
6      def comprobarEntero(self, valor):
7          try:
8              dato = int(valor)
9              return True
10         except:
11             return False
12
13     def comprobarDecimal(self, valor):
14         try:
15             dato = float(valor)
16             return True
17         except:
18             return False
```

CLASE MAIN

Esta es la clase principal y será la que se ejecutará al correr el programa.

PANTALLA INICIAL

Imprime en consola la información requerida en el enunciado de la práctica (curso, sección, carné, nombre del estudiante), y al pulsar enter continúa con las demás sentencias, en donde se crea un objeto tipo Pila y otro tipo Utilidades, y una variable bandera para su uso posterior.

```
10     print("""
11     Curso: Lenguajes Formales y de Programación
12     Sección: B-
13     Nombre: Ana Lucia Fletes Ordóñez
14     Carné: 202010003
15     """)
16     input("Pulse enter para continuar...")
17
18     miPila = Pila()
19     utilidad = Utilidades()
20     menuPrincipal = True
```

MENÚ PRINCIPAL

El menú principal se encuentra dentro de un ciclo while, que se repetirá hasta que el valor de la bandera 'menuPrincipal' cambie a falso. Este contiene una estructura try-except que capturará los distintos errores en ejecución. Se establece una variable bandera denominada 'menuSecundario' que tendrá el mismo fin que 'menuPrincipal', solamente que con menús internos, y una variable 'opcionMenuPrincipal' que tomará el valor ingresado por el usuario, la cual será verificada y permitirá el acceso a una funcionalidad específica.

```
22     while(menuPrincipal):
23         try:
24             menuSecundario = True
25             print("\n== MENÚ ==")
26             "\n1 - Cargar archivos de entrada"
27             "\n2 - Gestionar películas"
28             "\n3 - Filtrado"
29             "\n4 - Gráfica"
30             "\n5 - Salir"
31             opcionMenuPrincipal = int(input("Ingrese la opción a ejecutar:"))
```

CARGAR ARCHIVOS DE ENTRADA

Esta opción solicita al usuario la ruta del archivo de entrada, para luego leer este línea por línea a través de un bucle for. Cada línea separará sus elementos por “;” (a través del método split(parámetro)) y se almacenarán en una lista denominada ‘película’. Después, se establecen variables para guardar cada uno de los datos de la película (nombre, actores, año y género; teniendo actores un ciclo for para añadir a una lista a cada actor como un elemento de la misma). Después se tiene otro for para comprobar que la película no esté repetida en la lista de películas del objeto tipo Pila; si se repite, se eliminará el registro que estaba guardado y se agregará el nuevo a la Pila, y si no se repite, solamente se insertará el nuevo registro (que es un objeto tipo Película) en la Pila.

```
32 if opcionMenuPrincipal == 1:  
33     print("\n--- Cargar archivo de entrada ---")  
34     ruta = str(input("Ingrese la ruta del archivo de entrada: "))  
35     try:  
36         archivo = open(ruta, "r")  
37         for linea in archivo:  
38             pelicula = linea.split(";")  
39             nombrePelicula = pelicula[0]  
40             actoresPelicula = []  
41             for actor in pelicula[1].split(","):  
42                 actoresPelicula.append(actor.strip())  
43             anioPelicula = pelicula[2]  
44             generoPelicula = pelicula[3].replace("\n", "")  
45             contador = 0  
46             for peliculaGuardada in miPila.retornarPeliculas():  
47                 peliPila = peliculaGuardada.obtenerNombre().strip()  
48                 pelileida = nombrePelicula.strip()  
49                 if peliPila.lower() == pelileida.lower():  
50                     miPila.eliminarPelicula(contador)  
51                     break  
52             contador = contador + 1  
53             elementoPelicula = Pelicula(nombrePelicula.strip(), actoresPelicula, anioPelicula.strip(), generoPelicula.strip())  
54             miPila.push(elementoPelicula)  
55             archivo.close()  
56             print("Archivo cargado con éxito!")  
57     except:  
58         print("[Error]: Archivo no encontrado")
```

GESTIONAR PELÍCULAS

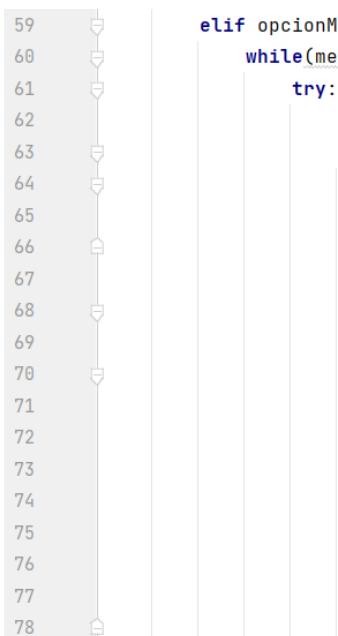
En gestionar películas se despliega un submenú al usuario, en donde este debe ingresar el valor de la función a realizar.

- **Caso ‘1’:** se mostrarán en consola la información de todas las películas guardadas hasta el momento, recorriendo la lista que las contiene por medio de un for y utilizando métodos accesores para obtener los datos específicos de las películas

(nombre, actores, anio y genero; para actores se utiliza otro for debido a que es una lista y de esa forma se puede recorrer y obtener a cada actor por separado).

- **Caso '2':** se mostrarán en consola los nombres de las películas almacenadas por el momento y se solicitará al usuario que ingrese el número de la película, para ver a los actores de la misma. Lo que sucede después es que a una variable llamada actoresPeliculaElegida se va a asignar la lista de actores que tiene el objeto Pelicula seleccionado. Posteriormente se recorre esta variable que contiene a los actores con un ciclo for, para de esa forma poder imprimir a los actores uno por uno.
- **Caso '3':** con esta opción, la variable booleana 'menuSecundario' se convertirá en falsa y no se cumplirá la condición del while para el submenú, por lo que regresará al while principal, donde se encuentra el menú principal.

Si se ingresa un valor que no coincida con ninguna de las opciones existentes o un tipo de dato diferente a un entero, se notificará al usuario con un mensaje en consola y se volverá a imprimir el submenú de Gestionar Películas.



```
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
      elif opcionMenuPrincipal == 2:
          while(menuSecundario):
              try:
                  print("\n--- Gestionar peliculas ---")
                  if miPila.tamanio() != 0:
                      print("1 - Mostrar peliculas"
                            "\n2 - Mostrar actores"
                            "\n3 - Regresar al menú principal")
                  opcionMenuSecundario = int(input("Ingrese la opción a ejecutar:"))
                  if opcionMenuSecundario == 1:
                      contador = 1
                      for pelicula in miPila.retornarPeliculas():
                          print("\nPELÍCULA #", contador)
                          print("Nombre:", pelicula.obtenerNombre())
                          print("Actores:")
                          for actor in pelicula.obtenerActores():
                              print("\t-", actor)
                          print("Año:", pelicula.obtenerAnio())
                          print("Género:", pelicula.obtenerGenero())
                          contador = contador + 1
```

```

79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
    elif opcionMenuSecundario == 2:
        contador = 1
        print("\n--- PELÍCULAS ---")
        for pelicula in miPila.retornarPeliculas():
            print("#", contador, "-", pelicula.obtenerNombre())
            contador = contador + 1
        try:
            numeroPelicula = int(
                input("Ingrese el número correspondiente a la película a elegir:")) - 1
            actoresPeliculaElegida = miPila.devolverPelicula(numeroPelicula).obtenerActores()
            print("\n***", miPila.devolverPelicula(numeroPelicula).obtenerNombre())
            print("\tActores:")
            for actor in actoresPeliculaElegida:
                print("\t-", actor)
        except:
            print("[Error]: Opción inválida")
        elif opcionMenuSecundario == 3:
            print("--- Regresando al menú principal")
            menuSecundario = False
        else:
            print("No ingresó una opción existente")
        else:
            print("No hay películas cargadas")
            menuSecundario = False
    except:
        print("[Error]: Opción inválida")

```

FILTRADO

Filtrado posee un menú secundario, el cual se ejecutará repetidamente hasta que la condición ‘menuSecundario’ sea falsa. El usuario deberá ingresar la opción a realizar.

- **Caso ‘1’:** se presentará el filtrado por actor, en donde primero se imprimirá el nombre de los actores existentes en el sistema y luego se pedirá al usuario que ingrese el nombre del actor con el cual se filtrarán las películas. Se comprueba que el usuario haya ingresado un texto y no un valor numérico en el nombre del actor; después se recorren las películas guardadas a través de un for, y posteriormente se recorren los actores de cada película, en busca de aquellas que tengan entre sus actores el nombre ingresado por el usuario. Si coinciden el valor ingresado por el usuario y el nombre del actor almacenado en una película, entonces se irá imprimiendo el nombre de la misma (de la película). Y en caso de que ninguna película tenga como actor el valor ingresado por el usuario, se imprimirá “No hay películas”.
- **Caso ‘2’:** corresponde al filtro por año, en el cual se pide al usuario que ingrese en consola el año de la película con el que se filtrarán las películas. Se recorren las películas guardadas en el sistema, y se comparan los años de las películas

almacenadas con el valor del año ingresado por el usuario. Si coinciden, se imprimirá el nombre de la película y su género; y si no hay ninguna coincidencia, se imprimirá "No hay películas".

- **Caso '3':** destinada al filtro por género, donde el usuario debe ingresar el género con el cual se filtrarán las películas. Al igual que con el filtro de año, se verifica que no sea un valor numérico, se recorren las películas guardadas y se comparan el valor de género de la misma y el ingresado por el usuario. Si hay coincidencias, se imprimirá el nombre de la película; y si no hay ninguna, se imprimirá "No hay películas".
- **Caso '4':** con esta opción, la variable booleana 'menuSecundario' se convertirá en falsa y no se cumplirá la condición del while para el submenú, por lo que regresará al while principal, donde se encuentra el menú principal.

Si se ingresa un valor que no coincide con ninguna de las opciones existentes o un tipo de dato diferente a un entero, se notificará al usuario con un mensaje en consola y se volverá a imprimir el submenú de Filtrado.

```
103 elif opcionMenuPrincipal == 3:
104     while (menuSecundario):
105         try:
106             print("\n--- Filtrado ---")
107             if miPila.tamano() != 0:
108                 print("1 - Filtrado por actor"
109                     "\n2 - Filtrado por año"
110                     "\n3 - Filtrado por género"
111                     "\n4 - Salir")
112             opcionMenuSecundario = int(input("Ingrese la opción a ejecutar:"))
113             if opcionMenuSecundario == 1:
114                 print("\n--- Filtrado por actor ---")
115                 for actorMostrar in miPila.devolverActoresPeliculas():
116                     print("* " + actorMostrar)
117                     actorFiltro = input("Ingrese el nombre del actor a buscar:")
118                     if utilidad.comprobarEntero(actorFiltro) or utilidad.comprobarDecimal(actorFiltro):
119                         print("No debe ingresar valores numéricos")
120                     else:
121                         print("\nActor:", actorFiltro)
122                         print("Resultado:")
123                         contador = 0
124                         for pelicula in miPila.retornarPeliculas():
125                             for actor in pelicula.obtenerActores():
126                                 if actorFiltro.lower() == actor.lower():
127                                     print("\t*", pelicula.obtenerNombre())
128                                     contador = contador + 1
129                         if contador == 0:
130                             print("No hay películas")
```

```

131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167

    elif opcionMenuSecundario == 2:
        print("\n--- Filtrado por año ---")
        anioFiltro = int(input("Ingrese el año a buscar:"))
        print("\nAño:", anioFiltro)
        print("Resultado:")
        contador = 0
        for pelicula in miPila.retornarPeliculas():
            if anioFiltro == int(pelicula.obtenerAnio()):
                print("\t*", pelicula.obtenerNombre(), "-", pelicula.obtenerGenero())
                contador = contador + 1
        if contador == 0:
            print("No hay películas")

    elif opcionMenuSecundario == 3:
        print("\n--- Filtrado por género ---")
        generoFiltro = input("Ingrese el género a buscar:")
        if utilidad.comprobarEntero(generoFiltro) or utilidad.comprobarDecimal(generoFiltro):
            print("No debe ingresar valores numéricos")
        else:
            print("\nGénero:", generoFiltro)
            print("Resultado:")
            contador = 0
            for pelicula in miPila.retornarPeliculas():
                if generoFiltro.lower() == pelicula.obtenerGenero().lower():
                    print("\t*", pelicula.obtenerNombre())
                    contador = contador + 1
            if contador == 0:
                print("No hay películas")
    elif opcionMenuSecundario == 4:
        print("--- Regresando al menú principal")
        menuSecundario = False
    else:
        print("No ingresó una opción existente")
else:
    print("No hay películas cargadas")
    menuSecundario = False
except:
    print("[Error]: Opción inválida")

```

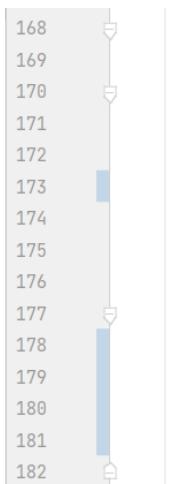
GRÁFICA

Esta opción crea un archivo con extensión 'dot', en donde se irán escribiendo las instrucciones para la creación de la gráfica. La primera instrucción es 'digraph{', debido a que se elaborará un gráfico dirigido, y este contendrá todos los nodos y relaciones entre estos; y luego se escribe 'rankdir = LR', para decir que irá de izquierda a derecha. Para las películas se tendrá el nombre de 'nodoPeliculaX', donde X representa un número que identificará a una película como única; y para los actores es lo mismo, solo que con el nombre de 'nodoActorX'.

Continuando con el código, se declara una variable 'actoresGrafica', que almacenará una lista con todos los actores (sin repetirse) y una cadena 'nodosActores', que guardará todas las instrucciones relacionadas a la creación de los nodos de los actores. Con un for se recorre la lista, y en cada iteración se crea el nodo para un actor, guardando el nombre del actor que se está recorriendo en ese momento y asignando propiedades como la forma que tendrá el nodo y el color de fondo.

Con las películas se tiene algo similar, solo que se tiene una cadena llamada 'edgePeliActor', donde se almacenarán las relaciones (aristas) existentes entre un nodoPelícula y nodoActor. Se tiene la variable 'nodosPelículas' que guardará todas las instrucciones relacionadas a la creación de los nodos de las películas, pero en este caso se hace uso de una tabla para contener los datos de nombre, año y género. Luego se recorren los actores de la película, y se crea una relación entre la película que se está iterando y el actor.

Posteriormente se escribe en el archivo .dot las instrucciones relacionadas a la creación de los nodos, tanto para películas como para actores, también aquellas correspondientes a las relaciones entre nodos, y un '}' para cerrar el 'digraph{' del inicio. Y finalmente, para convertir el .dot a pdf, se utilizó 'os.system("dot.exe -Tpdf grafica.dot -o graficoPelículas.pdf")'.



```
168 elif opcionMenuPrincipal == 4:
169     print("\n--- Gráfica ---")
170     if miPila.tamanio() != 0:
171         file = open("grafica.dot", "w")
172         file.write("digraph {\n")
173         file.write("rankdir = LR\n")
174         actoresGrafica = miPila.devolverActoresPeliculas()
175         contador = 0
176         nodosActores = ""
177         for actorGraph in actoresGrafica:
178             nodosActores += str("nodoActor" + str(contador) + 
179                               "[label=\"" + str(actorGraph) + 
180                               "\", fillcolor=\"#A0B3F2\", " +
181                               "shape=box style=filled width=3]\n")
182         contador = contador + 1
```

```

183 peliculasGrafica = miPila.retornarPelículas()
184 contador = 0
185 nodosPelículas = ""
186 edgePeliActor = ""
187 for pelícuGraph in peliculasGrafica:
188     origen = "origen" + str(contador)
189     nodosPelículas += str("nodoPelícula" + str(contador) + "[label=<" +
190         "<TABLE BORDER='0' CELLBORDER='1' CELLPADDING='0'>" +
191             "<TR>" +
192                 "<TD COLSPAN='2' bgcolor='#A0E9F2' PORT='1'" +
193                     + origen + ">" + str(pelícuGraph.obtenerNombre()) + "</TD>" +
194                 "</TR>" +
195                 "<TR>" +
196                     "<TD>" + str(pelícuGraph.obtenerAnio()) + "</TD>" +
197                     "<TD>" + str(pelícuGraph.obtenerGenero()) + "</TD>" +
198                 "</TR>" +
199             "</TABLE>> shape=none]\n")
200     for actorPeliGraph in pelícuGraph.obtenerActores():
201         edgePeliActor += str("nodoPelícula" + str(peliculasGrafica.index(pelícuGraph)) + ":" +
202             "origen" + str(peliculasGrafica.index(pelícuGraph)) + " -> " +
203             "nodoActor" + str(actoresGrafica.index(actorPeliGraph)) + "\n")
204     contador = contador + 1

205     file.write(nodosPelículas)
206     file.write(nodosActores)
207     file.write(edgePeliActor)
208     file.write("}")
209     file.close()
210     os.system("dot.exe -Tpdf grafica.dot -o graficoPelículas.pdf")
211     print("¡Gráfico generado con éxito!")
212 else:
213     print("No hay películas cargadas")

```

SALIR

Esta opción transforma la bandera ‘menuPrincipal’ en falsa, por lo que la condición del ciclo while que contiene el menú ya no se cumple y se termina la ejecución del mismo y del programa.

```

214 elif opcionMenuPrincipal == 5:
215     print("\n--- Salir ---"
216         "\nSaliendo del programa...")
217     menuPrincipal = False

```