# MANUAL DE USUARIO

Nicolás Muñiz Rodríguez : nicolas.muniz@udc.es

Pablo José Pérez Pazos : pablo.perez.pazos@udc.es

## · Mecanismo de ejecución:

Para conseguir ejecutar el programa lo primero que debemos hacer es extraer los archivos en código python del archivo en formato zip adjunto (“linked_ordered_positional_list.py”, ”array_ordered_positional_list.py” “doubly_linked_base.py” y “main.py”, aunque también añadimos el fichero de películas) por ejemplo a una carpeta del escritorio. A continuación, debemos abrir el archivo “main.py” mediante un editor de código, en nuestro caso VSCode. Aquí, debemos abrir la terminal (basta con pulsar Ctrl + J o uno de los cuatro iconos cuadrados de arriba a la derecha) y, después de haber navegado con comandos cd/ls a la carpeta en la que están los archivos, introducir el comando de ejecución con la forma python main.py [archivo de texto]:

```
PS C:\Users\NICO\Desktop\Entrega FP2> python .\main.py .\peliculas.txt
```

(Los caracteres “.\” salen al autocompletar el nombre de los archivos con Tab.)

A continuación se iniciará el programa y la lectura del archivo de texto, por lo que saltará el menú de opciones, que el usuario podrá introducir desde la misma terminal.

## · Fases de desarrollo:

Primero, decidimos construir el lector del archivo de texto, lo cual no fue complicado ya que nos pudimos apoyar en los ejemplos de las prácticas previas. Posteriormente, creamos la clase abstracta “Pelicula” que hace referencia al concepto general de una película, para esta clase decidimos incorporar las variables imprescindibles (título, director/a, año de estreno y puntuación media) como variables privadas (denotadas por “_” delante del nombre) y creamos algunos métodos clásicos para la correcta funcionalidad de los procesos como *__init__()* y *__str__()*.

También, por primera vez decidimos implementar los denominados métodos mágicos como*: __eq__()*, *__gt__()*, *__lt__()* y *__ge__()*. Además, para en un futuro poder llamar atributos que inicialmente son privados, creamos los *getters* de las diferentes variables.

Al finalizar la clase “Película”, decidimos crear la clase de “SimuladorPeliculas” para así poder crear una lista posicional ordenada, sin duplicar ni título ni director en la propia lista tal y como exige el enunciado. Para esto empleamos las funciones *“crear_lista_peliculas()”,* en el que se encuentra un tradicional bucle *for*, y *“eliminar_repetidos()”*, cuya gestión e implementación nos dió un trabajo considerable. Creamos también un menú interactivo con el cual el usuario podrá mediante el tecleo de una letra (de la A a la E y la Q para salir si el usuario lo desea, admitiendo sus variantes en minúsculas) elegir si prefiere ver todas las películas del catálogo, buscar película por director o año, crear un archivo sin películas duplicadas o mostrar tres estadísticas: número de películas por autor, puntuación media por autor y puntuación media por año de estreno.
Esto proporciona orden y organización para el usuario.

Cabe destacar que nos surgieron ciertos problemas con la funcionalidad del programa, tanto como en el uso de la librería *Pandas*, en la que nos surgió un inconveniente con el añadido de los datos al *dataframe* general (el cual paliamos al encontrar cómo utilizar el parámetro *loc* del *dataframe*), como el no modificar la lista original (esta debía permanecer intacta para que no hubiese conflicto al solicitar varias opciones del menú en sucesión), o con el hecho de gestionar las películas repetidas de forma que funcionase correctamente para ambos tipos de lista que fue la implementación que más veces tuvimos que replantear y atacar de diferentes maneras.

Por último, ha resultado para nosotros la práctica en la cual menos problemas hemos tenido que afrontar, debido a nuestro estable conocimiento de la POO y el TAD de las listas posicionales. No sólo eso, sino que absolutamente todos los inconvenientes que nos surgieron están parcheados y el programa tiene una funcionalidad completa.