Algoritmos

- Nombre de la clase: BST (Binary Search Tree)

Descripción: Mediante esta clase podemos crear un árbol binario a partir de la lista de películas, y así mismo otras operaciones sobre este.

Método: innerAdd
Parámetros: recibe un objeto de tipo película y una variable de tipo puntero.
Descripción: Recibe el objeto de tipo película y obtiene su duración. Luego si el árbol esta vacío, el elemento se agrega en la raíz del árbol y si no, compara la duración de la película a ingresar con la del nodo "actual" al que se esta apuntando. Si es menor se agrega a la izquierda, y si es mayor a la derecha, esto, de no haber un elemento en estas posiciones. Si ya existe, se retorna un llamado recursivo a la misma función, tomando la posición "izquierda" o "derecha" del nodo actual como referencia.
Retorno: True - se agregó la película exitosamente
False - no se logró añadir película

Método: convert
Parámetros: reciba una lista de películas
Descripción: Se ingresa un elemento, y si es del tipo de lista de película entonces se hace referencia al primer elemento. Si este tiene un valor, entonces se agrega el valor a un nodo de árbol, y se pasa al siguiente elemento. Mientras los nodos no tengan una referencia a "None" y se haya terminado de convertir la lista, el proceso de conversión se repite. Como resultado cada nodo ahora sera del tipo nodo de árbol binario.
Retorno: True - se convirtió cada nodo de forma exitosa
False - no se lograron convertir los elementos

Método: create
Parámetros: recibe una lista de películas.
Descripción: Este método recibe una lista de películas e instancia un objeto de tipo árbol. Luego de esto, hace uso del método "convert" del árbol recien creado para convertir la lista.

Retorno: Retorna el árbol que se ha creado.
Retorna "None" si el parámetro que se ingresó no era del tipo 'lista de películas'.

Método: innerSubTree
Parámetros: valor del cual desearos obtener el subárbol y una variable tipo apuntador.
Descripción: Busca el nodo que contenga el valor ingresado mediante comparación. Cuando finalmente se encuentra el valor se instancia un objeto de tipo árbol y se le asigna a su raíz la dirección del valor ingresado.
Retorno: - retorna el subárbol encontrado
- Retorno "None" si no se apunta a una variable de tipo BST Node o si no se encontró el valor ingresado.

Método: innerSearch
Parámetros: recibe el valor a buscar y un apuntador.
Descripción: Busca el elemento comparando el valor ingresado con el valor al cual se apunta. Si es menor o mayor se retorna un llamado recursivo de la función tomando como referencia el nodo a la izquierda o a la derecha respectivamente. El proceso se repite hasta que se encuentra el valor.
Retorno: True - se encontró el valor
False - el árbol esta vacío, el apuntador no hace referencia a un objeto de tipo BST Node, o si no se encontró el valor.

Método: height
Parámetros: recibe una variable de tipo puntero y una variable contador.
Descripción: este método calcula la altura del árbol. Inicialmente, al no recibir el parámetro de tipo puntero, cuenta a la raíz como primer elemento. Luego, se navega hacia abajo evaluando cada siguiente nivel del nodo actual. Si este nodo tiene un hijo, se hace llamado recursivo con referencia a este y con el contador que incrementa a 1. Si tiene ambos hijos entonces se hace un llamado recursivo a cada uno y se evalua cual de estos tiene el subárbol más largo, para tomar este como el contador a retornar.

Retorno: El contador total de todo el árbol.

- Nombre de la clase: BSTNode

Descripción: Esta clase nos permite crear los elementos(nodos) que constituyen el TDA árbol binario.

Atributos: value (valor del nodo), right (apunta al hijo derecho), left (apunta al hijo izquierdo).

- Nombre de la clase: Node

Descripción: Esta clase nos permite crear los elementos (nodos) que constituyen el TDA lista enlazada.

Atributos: value (valor del nodo), next (apunta al próximo elemento).

- Nombre de la clase: Linked List

Descripción: Esta clase nos permite crear una lista enlazada a partir de nodos. Contiene diversos métodos que nos permiten hacer operaciones sobre esta.

Atributos: first (primer nodo de la lista enlazada)

Método: push

Parámetros: recibe el valor a agregar y la posición donde se desea agregar.

Descripción: si se ha ingresado una posición, se evalúa si es la primera posición o en cualquier otra. Se recorren los elementos de la lista hasta encontrar la posición en la que deseamos agregar. Si no se ingresa una posición o es una posición mayor a la longitud, entonces, si esta vacía agrega en la primera posición y si no, se agrego al final.

Retorno: True - se agregó el valor exitosamente.

Método: pop

parámetros: recibe la posición del elemento que se desea eliminar.

Descripción: si se ha ingresado una posición válida, se válida si es en la primera posición o en cualquier otra. Si es la primera posición se le asigna a "first" la referencia del siguiente elemento de la lista. Si es en cualquier otra, se hace referencia al nodo anterior, al siguiente elemento después del elemento que borraremos.

retorno: True - se eliminó el elemento.
False - no se eliminó el elemento (la lista esta vacía, no se ingresó la posición, o la posición es mayor al tamaño de la lista.

Método: length
Parámetros: ---
Descripción: mediante una variable de tipo apuntador se recorren las posiciones de la lista enlazada y por cada elemento que se recorre, una variable de tipo contador incrementa en uno. Se termina de "contar" cuando se encuentra una referencia a "null"
Retorno: 0 - la lista esta vacía
count - el número de elementos en la lista enlazada.

Método: Search
Parámetros: recibe la posición que se busca.
Descripción: se recorren los elementos de la lista enlazada, y por cada elemento que se recorre el contador aumenta en 1. Cuando la posición ingresada sea igual al contador, se regresa el valor del nodo en esta posición.
Retorno: False - la lista esta vacía, la posición ingresada no existe.
Retorna el valor en dicha posición.

Método: __str__
Parámetros: ---
Descripción: se recorre cada elemento con una variable apuntador y se convierte cada valor en una cadena.
Cada elemento se concatena en una variable de tipo string para su posterior retorno.
Retorno: la cadena donde están incluidos todos los elementos de la lista enlazada.

- Nombre de la clase: Movie
  Descripción: Los objetos de esta clase constituiran nuestra lista de películas a partir de la cual se basa el programa.
  Atributos: Nombre de la película, Id, Duración, autor, Descripción, y Género.

Método : str
Descripción: convierte a cadena una lista a partir de las atributos de la película.

- Nombre de la clase: MovieList
  Descripción: crea una lista de películas en base a una lista enlazada.
  Atributos: una lista enlazada

Método: MoviePush
Parámetros: objeto de tipo película.
Descripción: agrega elementos a la lista de películas por medio del "agregar" del TDA lista enlazada.
Retorno: ---

Método: moviePop
Parámetros: La posición del elemento que se desea eliminar.
descripción: elimina los elementos de la lista de películas por medio del "borrar" del TDA lista enlazada.
Retorno: ---

Método: movieSearch
Parámetros: La posición del elemento que se desea buscar.
Descripción: busca elementos en la lista de películas por medio del "buscar" del TDA lista enlazada

- Nombre de la clase: Table
  Descripción: Esta clase construye una tabla ASCII a partir de la lista de películas. Muestra: Id de Película, Nombre, su Duración y la Descripción.
  Atributos: ---

Método: generateTable
Parámetros: ---
Descripción: un apuntador recorre las películas y muestra los atributos antes mencionados en la descripción de la clase Table. La duración se muestra en segundos.

- Nombre de la clase: ToSeconds
  Atributos: ---
  Descripción: Esta clase contiene un único método encargado de convertir el valor de duración de una película. del formato "HH:MM:SS" a segundos.

  Método: toSeconds
  Parámetros: recibe el valor en formato "HH:MM:SS"
  Descripción: separa la cadena que se recibe por cada ":", mediante el método Split, y se almacena cada elemento resultante en una variable. Las variables representan las horas, minutos y segundos respectivamente. Por último se retorna la suma de estos valores convertidas a enteros multiplicados por su equivalente en segundos.
  Retorna: el equivalente en segundos del parámetro ingresado.

- Nombre de la clase: NodeTree
  Descripción: Esta clase nos permite crear los elementos (nodos) que constituyen el TDA árbol jerárquico.
  Atributos: value (valor del nodo), next (apunta al siguiente nodo), children (Tipo lista enlazada)

- Nombre de la clase: TreePure
  Descripción: Mediante esta clase podemos crear un árbol jerárquico a partir de la lista de películas, y así mismo otras operaciones sobre este.
  Atributos: la raíz del árbol y sus hijos los cuales constituyen las categorías.

  Método: push
  Descripción: agrega elementos al árbol mediante el "push" de la lista enlazada.
  Parámetros: El valor que se desea agregar y una variable de tipo apuntador para recorrer las posiciones a donde se van a agregar los elementos.
  Retorno: ---

  Método: search
  Descripción: busca elementos en el árbol mediante el "search" de la lista enlazada.

Parámetros: El valor que se desea buscar y una variable de tipo apuntador para recorrer los posiciones a lo largo de la búsqueda.

Método: length
Descripción: recorre cada unos de los elementos de la lista enlazada que constituye el árbol para encontrar la longitud de este.
Retorno: una variable contador con la longitud del árbol.

- Nombre de la clase: File Manager
  Descripción: la clase File Manager contiene los métodos importantes para escribir y leer en el disco duro un archivo de formáto Json que contiene las peliculas en el inventario de peliculas.
  Atributos: ---

Método: readFileJSON
Descripción: Método que recibe el nombre del archivo JSON y devuelve un objeto LinkedList.
Parámetros: Archivo JSON
Retorno: una lista enlazada.

Método: writeInFile
Descripción: Método que recibe como parámetro el archivo donde se va a ingresar la información y la lista enlazada de donde se obtiene dicha información.
Parámetros: un archivo y una lista enlazada.
Retorno: ---

Método: convertJSONToLinkedList
Descripción: convierte el diccionario en una lista enlazada.
Parámetros: un diccionario JSON.
Retorno: una lista enlazada.

Método: convertLinkedListToJSON
Descripción: convierte una lista enlazada en un diccionario.
Parámetros: una lista enlazada
retorno: un diccionario JSON.