Sesión4 de Estructura de Datos: **Curso 2011/2012**

El objetivo de esta sesión es aprender cómo programar las pilas (STACK) y colas (QUEUE). La fecha límite de entrega de esta sesión es el día 01-04-2011. Documentar el código.

Ejercicio

Crear una aplicación en modalidad texto usando las estructuras de pilas (STACK) y colas (QUEUE) como listas enlazadas. Usar el esqueleto de programa python (*ED_P2_2011_entrega.py*) disponible en el campus virtual. Implementar la clase **Nodo** y las dos clases **STACK** y **QUEUE**. Cada método debe ser programado, y está prohibido usar las funciones de python *pop y push.*

Los pasos a seguir son los siguientes:

Paso 1. Bajar desde el Campus Virtual el fichero *ED_P2_2011_entrega.py* que contiene las definiciones de las clases y los nombres de métodos necesarios para esta entrega. Podéis modificar este fichero según vuestro criterio.

Paso 2. Cargar el fichero `peliculas100.dat` que contiene una lista de películas cada una definida según el formato de la Sesión 1. Usar la función *loadMovieList* que ya se ha codificado en la entrega anterior. La función devuelve una lista de objetos MOVIE.

Paso 3. Limpiar la clase MOVIE de las sobrecargas previamente implementadas y no necesarias, y codificar el método *GetTitle.*

Paso 4. Definir la clase *Nodo* necesaria para implementar las estructuras de datos.

STACK

Un STACK o pila es una estructura de datos de tipo LIFO (Last in First out), que significa que la última película entrada en la estructura de datos es la primera que saldrá.

Paso 5. Definir la clase STACK con sus datos y métodos, usando los nombres de definidos en el fichero *ED_P2_2011_entrega.py*. CUIDADO!!! Gestionar bien los casos excepcionales (STACK vacío, etc.)!

Paso 6. Comprobar que el código funcione correctamente usando las siguiente llamadas (el output de cada línea de código está indicado en azul en este documento).

```
>>>newStack = Stack()
>>>newStack.printStack()
----------
Empty Stack!
==========

>>>newNode = Node(movieList[0])
>>>newStack.push(newNode)
>>>newStack.printStack()
----------
Toy Story
```

```
==========

>>>newStack.push(Node(movieList[1]))
>>>newStack.printStack()
----------
Toy Story
Jumanji
==========

>>>newStack.push(Node(movieList[2]))
>>>newStack.printStack()
----------
Toy Story
Jumanji
Grumpier Old Men
==========

>>>m1 = newStack.pop()
>>>newStack.printStack()
----------
Toy Story
Jumanji
==========

>>>m2 = newStack.pop()
>>>newStack.printStack()
----------
Toy Story
==========

>>>m3 = newStack.pop()
>>>newStack.printStack()
----------
Empty Stack!
==========

>>>m4 = newStack.pop()
>>>newStack.printStack()
Empty Stack! Pop is not allowed!
----------
Empty Stack!
==========

GUESS WHAT HAPPENS...
>>>print m1.GetTitle()
>>>print m2.GetTitle()
>>>print m3.GetTitle()
>>>print m4.GetTitle()
```

QUEUE

Un QUEUE o cola es una estructura de datos de tipo FIFO (First in First out), que significa que la primera película entrada en la estructura de datos es la primera que saldrá.

Paso 7. Definir la clase QUEUE con sus datos y métodos, usando los nombres definidos en el fichero *ED_P2_2011_entrega.py*. CUIDADO!!! Gestionar bien los casos excepcionales!

Paso 8. Comprobar que el código funcione correctamente usando las siguiente llamadas (el output de cada línea de código está indicado en azul en este documento).

```
>>>newQ = Queue()
>>>newQ.printQ()
----------
Empty Queue!
==========

>>>newNode = Node(movieList[0])
>>>newQ.encueue(newNode)
>>>newQ.printQ()
----------
Toy Story
==========

>>>newQ. encueue(Node(movieList[1]))
>>>newQ.printQ()
----------
Jumanji
Toy Story
==========

>>>newQ. encueue(Node(movieList[2]))
>>>newQ.printQ()
----------
Grumpier Old Men
Jumanji
Toy Story
==========

>>>m1 = newQ. decueue ()
>>>newQ.printQ()
----------
Grumpier Old Men
Jumanji
==========

>>>m2 = newQ. decueue ()
>>>newQ.printQ()
----------
Grumpier Old Men
```

==========

>>>m3 = newQ. decueue ()
>>>newQ.printQ()

Empty Queue!
==========

>>>m4 = newQ.decueue()
>>>newQ.printQ()
Empty Queue! Decueue is not allowed!

Empty Queue!
==========

GUESS WHAT HAPPENS...
>>>print m1.GetTitle()
>>>print m2.GetTitle()
>>>print m3.GetTitle()
>>>print m4.GetTitle()

NOTA: La entrega de la sesión ha de ser únicamente por el Campus Virtual dentro del límite de tiempo predefinido. Se entregará un único fichero ".py" que contenga el código fuente documentado.