TRABAJO PRÁCTICO DE PARADIGMAS DE PROGRAMACION

Agüero, Juan Ignacio – 48062

Caro, Ariana – 47943

Fermanelli, Sebastián – 47983

Natalicchio, Oriana Sofia – 48170

Comisión: 202.

Año calendario: 2021.

El estudio cinematográfico CINEMA tiene registrada la información que maneja. Para resolver la situación que plantearemos necesitamos contar con los datos del personal que desempeña tareas varias en forma permanente (información que ya está cargada) y la de las películas que son producidas en el mismo.

Del personal permanente se conocen los siguientes datos:

- **Número de documento**
- **Nombre y apellido**
- **Plus** (porcentaje sobre el remanente del presupuesto, que cada empleado cobra, una vez terminado el rodaje de una película, y es distinto para cada uno)
- **Sueldo básico** (que es el mismo para todos los empleados)
- **Especialidad** (rol que cumple en el estudio)

Además de los empleados existen otras personas que integran el staff, estos son los actores y el equipo de dirección.
De cada uno de ellos, ya sea actor o parte del equipo de dirección se conoce:

- **Número de documento**
- **Apellido y nombre**
- **Nacionalidad**

Si es actor, además tendrá pactado un **cachet** (monto que cobrará por su participación en la película), en cambio, si es del al equipo de dirección acordará un **porcentaje** sobre el presupuesto asignado a la película, a partir del cual se calculará su paga.

De cada película que produce el estudio se almacena la siguiente información:

- **Código**
- **Título**
- **Staff** (personal del estudio, actores y equipo de dirección)
- **Presupuesto asignado**
- **Presupuesto remanente**

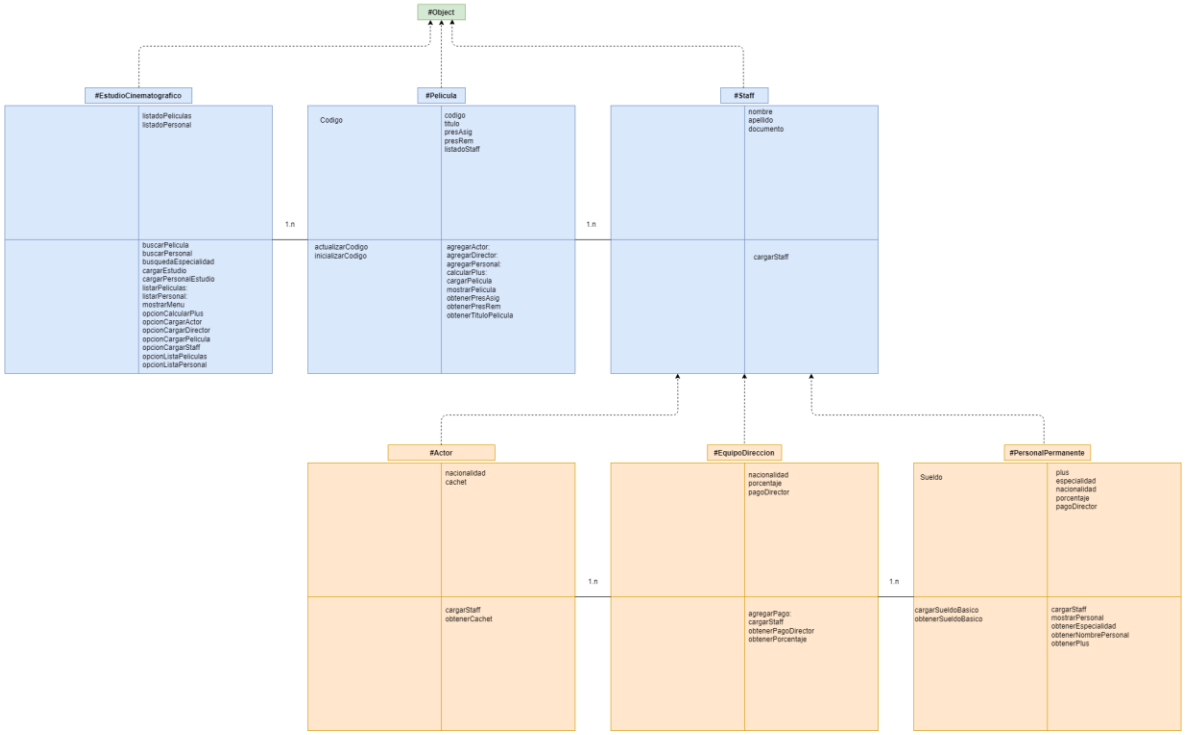
En el momento de instanciar una película solo se cargarán los datos básicos, el personal del estudio que participará del rodaje (validando que forme parte del personal permanente) y el presupuesto asignado.

A medida que se van agregando personas (actores y equipo de dirección) al staff se ira calculando el presupuesto remanente.

Se desea confeccionar el correspondiente diseño del sistema siguiendo los lineamientos de la programación orientada objetos y un programa en Smalltalk que permita a través de un menú realizar las siguientes operaciones:

1. Instanciar una película.
2. Agregar una persona al staff.
3. Calcular el plus que cobrará un empleado determinado, una vez finalizado el rodaje de una película.

DIAGRAMA DE CLASES



Programa principal

```
| a |
	a := AAEstudioCinematografico new.
	a mostrarMenu
```

#AAEstudioCinematografico

MC

MI

```
instanceVariableNames: 'listadoPeliculas listadoPersonal'
buscarPelicula
	| pelicula objeto |
	listadoPeliculas isEmpty
		ifTrue: [MessageBox notify: 'No hay peliculas cargadas']
		ifFalse:
			[[objeto isNil] whileTrue:
				[pelicula := Prompter prompt: 'Ingrese el nombre de la pelicula'.
				objeto := listadoPeliculas detect: [:i | pelicula = i obtenerTituloPelicula]
					ifNone:
						[objeto := nil.
						MessageBox notify: 'No hay ninguna pelicula con ese titulo']]].
	^objeto

buscarPersonal
	| personal objeto |
	listadoPersonal isEmpty
		ifTrue: [MessageBox notify: 'No hay ningun empleado cargado']
```

```
        ifFalse:
            [[objeto isNil] whileTrue:
                [personal := Prompter prompt: 'Ingrese el nombre del personal'.
                objeto := listadoPersonal detect: [:i | personal = i obtenerNombrePersonal]
                    ifNone:
                        [objeto := nil.
                        MessageBox notify: 'El personal no existe']]].
    ^objeto

busquedaEspecialidad
    | colec especialidad |
    especialidad := Prompter
                prompt: 'Ingrese la especialidad a buscar. S(Sonidista) | I(Iluminador) | C(Camarografo)'.
    colec := listadoPersonal select: [:i | i obtenerEspecialidad = especialidad].
    colec isEmpty
        ifTrue: [MessageBox notify: 'No hay ningun empleado con esa especialidad']
        ifFalse: [self listarPersonal: colec]

cargarEstudio
    listadoPeliculas := OrderedCollection new.
    listadoPersonal := OrderedCollection new.
    AAPelicula inicializarCodigo.
    MessageBox notify: 'Estudio Cinematografico CINEMA'.
    PersonalPermanente cargarSueldoBasico.
    self cargarPersonalEstudio

cargarPersonalEstudio
    | nuevo personal |
```

```
        MessageBox notify: 'Cargar los empleados del estudio'.
        nuevo := true.
        [nuevo] whileTrue:
                [personal := PersonalPermanente new.
                personal cargarStaff.
                listadoPersonal add: personal.
                nuevo := MessageBox confirm: '¿Desea agregar otro empleado?']

listarPeliculas: lista
        Transcript
                clear;
                cr;
                show: 'COD';
                tab;
                tab;
                show: 'TITULO';
                tab;
                tab;
                show: 'PRES ASIG';
                tab;
                tab;
                show: 'PRES REM';
                cr.
        lista do: [:i | i mostrarPelicula]

listarPersonal: lista
        Transcript
                clear;
                cr;
                show: 'NOMBRE';
```

```
            tab;
            tab;
            show: 'APELLIDO';
            tab;
            tab;
            show: 'DOC';
            tab;
            tab;
            show: 'PLUS';
            tab;
            tab;
            show: 'ESP';
            cr.
        lista do: [:i | i mostrarPersonal]

mostrarMenu
        | opcion |
        self cargarEstudio.
        opcion := nil.
        [opcion = '0'] whileFalse:
                [MessageBox
                        notify: '1- Cargar una pelicula. 2- Cargar un
actor/director. 3- Calcular plus de un empleado. 4-Ver listado de peliculas. 5- Ver
listado de empleados. 0- Salir'.
                opcion := Prompter prompt: 'Elija una opcion'.
                opcion = '1' ifTrue: [self opcionCargarPelicula].
                opcion = '2' ifTrue: [self opcionCargarStaff].
                opcion = '3' ifTrue: [self opcionCalcularPlus].
                opcion = '4' ifTrue: [self opcionListaPeliculas].
                opcion = '5' ifTrue: [self opcionListaPersonal]]
```

```
opcionCalcularPlus
	| pelicula personal pago |
	pelicula := self buscarPelicula.
	pelicula isNil
		ifFalse:
			[personal := self buscarPersonal.
			personal isNil
				ifFalse:
					[pago := pelicula calcularPlus: personal.
					MessageBox
						notify: 'En la pelicula ' , pelicula obtenerTituloPelicula , ' el empleado '
							, personal obtenerNombrePersonal , ' obtendra un plus de $'
							, pago printString]]

opcionCargarActor
	| actor pelicula |
	pelicula := self buscarPelicula.
	pelicula isNil
		ifFalse:
			[actor := Actor new.
			actor cargarStaff.
			pelicula agregarActor: actor]

opcionCargarDirector
	| director pelicula pago |
	pelicula := self buscarPelicula.
	pelicula isNil
		ifFalse:
			[director := EquipoDireccion new.
```

```
                director cargarStaff.
                pago := pelicula obtenerPresAsig * director obtenerPorcentaje
/ 100.
                director agregarPago: pago.
                pelicula agregarDirector: director]

opcionCargarPelicula
        | pelicula nuevo personal |
        pelicula := AAPelicula new.
        pelicula cargarPelicula.
        MessageBox notify: 'Ingrese el personal del estudio que trabajara en la
pelicula'.
        nuevo := true.
        [nuevo] whileTrue:
                [personal := self buscarPersonal.
                personal isNil ifFalse: [pelicula agregarPersonal: personal].
                nuevo := MessageBox confirm: '¿Desea agregar otro
personal?'].
        listadoPeliculas add: pelicula

opcionCargarStaff
        | opcion |
        opcion := nil.
        [opcion = '0'] whileFalse:
                [MessageBox notify: '1- Cargar un actor. 2- Cargar un director.
0- Salir'.
                opcion := Prompter prompt: 'Elija una opcion'.
                opcion = '1' ifTrue: [self opcionCargarActor].
                opcion = '2' ifTrue: [self opcionCargarDirector]]

opcionListaPeliculas
        self listarPeliculas: listadoPeliculas
```

opcionListaPersonal

self busquedaEspecialidad

#AAPelicula

MC

classVariableNames: 'Codigo'

actualizarCodigo

```
Codigo := Codigo + 1
```

inicializarCodigo

```
Codigo := 1
```

MI

instanceVariableNames: 'codigo titulo presAsig presRem listadoStaff'

agregarActor: actor

```
presRem := presRem - actor obtenerCachet.
listadoStaff add: actor
```

agregarDirector: director

```
presRem := presRem - director obtenerPagoDirector.
listadoStaff add: director
```

agregarPersonal: personal

```
presRem := presRem - PersonalPermanente obtenerSueldoBasico.
listadoStaff add: personal
```

calcularPlus: personal

```
| pago |
pago := presRem * personal obtenerPlus / 100.
^pago
```

cargarPelicula

```
codigo := Codigo.
AAPelicula actualizarCodigo.
```

```
	titulo := Prompter prompt: 'Ingrese el titulo de la pelicula ' , codigo printString.
	presAsig := (Prompter prompt: 'Ingrese el presupuesto de la pelicula ' , titulo) asNumber.
	presRem := presAsig.
	listadoStaff := OrderedCollection new
```

mostrarPelicula

```
	Transcript
		cr;
		show: codigo printString;
		tab;
		tab;
		show: titulo asUppercase;
		tab;
		tab;
		show: presAsig printString;
		tab;
		tab;
		show: presRem printString;
		cr
```

obtenerPresAsig

```
	^presAsig
```

obtenerPresRem

```
	^presRem
```

obtenerTituloPelicula

```
	^titulo
```

#AAStaff

MC

MI

instanceVariableNames: 'nombre apellido documento'

cargarStaff

```
nombre := Prompter prompt: 'Ingrese el nombre de la persona'.
apellido := Prompter prompt: 'Ingrese el apellido de la persona'.
documento := (Prompter prompt: 'Ingrese el documento de identidad') asNumber
```

AAStaff subclass: #Actor

MC

MI

instanceVariableNames: 'nacionalidad cachet'

cargarStaff

```
super cargarStaff.
nacionalidad := Prompter prompt: 'Ingrese la nacionalidad del actor'.
cachet := (Prompter prompt: 'Ingrese el cachet a cobrar por el actor') asNumber
```

obtenerCachet

```
^cachet
```

AAStaff subclass: #EquipoDireccion

MC

MI

instanceVariableNames: 'nacionalidad porcentaje pagoDirector'

agregarPago: pago

```
pagoDirector := pago
```

cargarStaff

super cargarStaff.

nacionalidad := Prompter prompt: 'Ingrese la nacionalidad del director'.

porcentaje := (Prompter prompt: 'Ingrese el porcentaje a cobrar por el director') asNumber asFloat

obtenerPagoDirector

^pagoDirector

obtenerPorcentaje

^porcentaje

AAStaff subclass: #PersonalPermanente

MC

classVariableNames: 'Sueldo'

cargarSueldoBasico

Sueldo := (Prompter prompt: 'Ingrese el sueldo basico de los empleados del estudio') asNumber

obtenerSueldoBasico

^Sueldo

MI

instanceVariableNames: 'plus especialidad'

cargarStaff

super cargarStaff.

plus := (Prompter prompt: 'Ingrese el porcentaje a cobrar por el empleado') asNumber asFloat.

especialidad := (Prompter

prompt: 'Ingrese la especialidad del empleado. S(Sonidista) | I(Iluminador) | C(Camarografo)')

asUppercase

```
mostrarPersonal
	Transcript
		cr;
		show: nombre asUppercase;
		tab;
		tab;
		show: apellido asUppercase;
		tab;
		tab;
		show: documento printString;
		tab;
		tab;
		show: plus printString;
		tab;
		tab;
		show: especialidad asUppercase;
		cr

obtenerEspecialidad
	^especialidad

obtenerNombrePersonal
	^nombre

obtenerPlus
	^plus
```

a. ***Indique si encuentra clases abstractas. ¿Por qué son abstractas? Definir y dar ejemplos.***

   En nuestro trabajo identificamos la clase 'Staff' como abstracta, ya que, según la definición de esta, no se crean instancias, solo se usa para organizar.

b. ***Indique si encuentra clases concretas. ¿Por qué son concretas? Definir y dar ejemplos.***

   En nuestro trabajo, identificamos varias clases concretas, como 'Actor' o 'Personal Permanente', ya que poseen métodos en donde se crean instancias.

c. ***¿Qué relaciones de herencia reconoce? Dar ejemplos. Explicar el concepto de herencia.***

   Las clases "Actor", "EquipoDireccion" y "PersonalPermanente" heredan de la clase "Staff" sus atributos.
   Es la propiedad de las subclases de una clase específica, mediante la cual heredan (incluyen) las propiedades de sus superclases.

d. ***¿Qué relaciones de ensamble reconoce? Dar ejemplos. Explicar el concepto de ensamble.***

   Las clases "Pelicula", "EstudioCinematografico" y "Staff" son ejemplos claros de ensamble, ya que, la instancia de una o más películas pueden pertenecer al estudio cinematográfico. También, uno o varios staffs pueden estar ligados a una instancia de película, sin embargo, no son de herencia, ya que, hablamos de objetos(clases) totalmente diferentes.
   "Es la relación todo-parte o es parte de, en la cual los objetos que presentan los componentes de algún conjunto se asocian a un objeto que representa el todo".

e. ***¿Existen métodos polimórficos? Dar ejemplos. Explicar el concepto de polimorfismo.***

   Las clases "Actor", "PersonalPermanente" y "EquipoDireccion" usan un método polimórfico llamado "cargarStaff", ya que, según la definición, permite enviar el mismo mensaje a objetos diferentes y que cada uno

responda en la forma apropiada según el tipo de objeto que sea, ejecutando el mismo.
"Es la capacidad de las diferentes clases para responder al mismo protocolo, esta característica habilita al programador para tratar uniformemente objetos que provienen de clases diferentes."

f. ¿Existen métodos con redefinición polimórfica? Dar ejemplos. Explicar el concepto de redefinición.

Al igual que en la consigna anterior, el mismo método usa el concepto de redefinición polimórfica, ya que, desde la clase "Staff" heredan ciertos atributos y luego, desde la misma se redefine agregando otros atributos.

g. Identifique las variables de Clase utilizadas. Dar ejemplos. Explique qué son las variables de Clases y porque/cómo se utilizan en el presente trabajo.

En la clase "PersonalPermanente" utilizamos la variable de clase "Sueldo". Las variables de clase sirven para determinar una variable que compartirán todas las instancias de la clase.
"Son variables compartidas por todas las instancias de una clase y sus subclases. Tienen el mismo valor para todas las instancias. Solo pueden ser accedidas por los métodos de clases y los métodos de instancia de la clase y sus subclases".

h. Identifique las variables de Instancia utilizadas. Dar ejemplos. Explique qué son las variables de Instancia y sus diferencias con las variables de Clase.

En la clase "Staff" se utiliza la variable de instancia "documento", esta variable indica que al ser de instancia, cada instancia creada de la clase staff tendrá un documento diferente a las demás instancias. En este caso, el fin es determinar un documento de identidad único para cada persona.
"Las variables de instancia existen durante todo el tiempo de vida de un objeto y representan el estado del objeto".

i. Identifique los métodos de Clase. Dar ejemplos. ¿Por qué son necesarios los métodos de clase?

Un método de clase utilizado es "inicializarCodigo", la misma sirve para poder manipular las variables de clase, ya que, estas solo pueden ser modificadas por métodos de clase.

j. Identifique mensajes unarios, binarios y de palabra clave. De ejemplos de cada uno de ellos.

listadoPersonal isEmpty. El mensaje representa uno unario, ya que no recibe parámetros.

*pago := presRem * personal obtenerPlus / 100.* El mensaje representa uno binario, ya que, realiza el cálculo para obtener el pago mediante el presupuesto remanente de la pelicula multiplicando por el porcentaje del plus a cobrar, dividido entre 100.

pelicula agregarPersonal: personal. El mensaje representa uno de palabra clave, en este caso a la variable receptora “pelicula” se le agrega una instancia de “personal” mediante el método “agregarPersonal:”.