**FIUBA - 75.07**

**Algoritmos y programación III**

*Trabajo práctico 2: FonTruco*

2do cuatrimestre, 2015

(trabajo grupal)

Alumnos:

|  |  |  |
| --- | --- | --- |
| **Nombre** | **Padrón** | **PuertoEstelarTerranMail** |
| Bourbon Navarro, Rodrigo |  | rodrigobourbon@gmail.com |
| Gomez Peter, Federico Manuel | 96091 |  |
|  |  |  |
|  |  |  |

***Fecha de entrega final***: Miércoles 2/12/2015 - Jueves 3/12/2015

***Tutor***:

***Nota Final***:

# 

# 

Introducción

# Objetivo del trabajo

Aplicar los conceptos enseñados en la materia a la resolución de un problema, trabajando en forma grupal y utilizando un lenguaje de tipado estático (Java)

# Consigna general

Desarrollar la aplicación completa, incluyendo el modelo de clases e interfaz gráfica. La aplicación deberá ser acompañada por prueba unitarias e integrales y documentación de diseño. En la siguiente sección se describe la aplicación a desarrollar.

Descripción de la aplicación a desarrollar

Se deberá desarrollar una aplicación que implemente el juego de cartas popular Truco (https://es.wikipedia.org/wiki/Truco\_(juego\_de\_naipes)) .

Las características del mismo serán:

* Se podrá seleccionar la variante con/sin Flor.
* El juego será entre jugadores humanos.
* Se podrá jugar en las variantes 2 jugadores, 4 jugadores y pica-pica.
* En la variante de 2 jugadores, se podrá jugar contra la computadora, con una inteligencia mínima y determinística (esto se comentará en clase).

Se desarrollará la interfaz visual para la interacción entre los jugadores.

**Entregables**

* Código fuente de la aplicación completa, incluyendo también: código de la pruebas, archivos de recursos
* Script para compilación y ejecución (ant)
* Informe, acorde a lo especificado en este documento

## Formas de entrega

## Habrá 4 entregas formales. Las mismas tendrán una calificación de APROBADO o NO APROBADO en el momento de la entrega.

Aquél grupo que acumule 3 no aprobados, quedará automáticamente desaprobado con la consiguiente pérdida de regularidad en la materia. En cada entrega se debe traer el informe actualizado.

1er Entrega: *Mínimamente* (se aconseja avanzar lo más posible) pruebas de integración y unitarias funcionando que contemplan:

* Modelado de las cartas y los valores entre ellas
* Modelado de ‘manos’ y ‘mesa’
* Modelado de cálculo envido y flor para una ‘mesa’ dada

2da Entrega: **Modelo del Juego Completo** con todas las pruebas unitarias y de integración que contemplen todos los casos del enunciado, simulando partidas completas en todas sus variantes.

3er Entrega: Interfaz gráfica Parcial. A determinar por el ayudante.

4ta y última Entrega: Trabajo Práctico completo funcionando, con interfaz gráfica final, sonidos e informe completo.

## Fechas de entrega programadas

1er Entrega: *Miércoles 11 / 11 / 2015 - Jueves 12 / 11 / 2015*

2da Entrega: *Miércoles 18 / 11 / 2015 - Jueves 19 / 11 / 2015*

3er Entrega: *Miércoles 25 / 11 / 2015 - Jueves 26 / 11 / 2015*

4ta y última Entrega: *Miércoles 2 / 12 / 2015 - Jueves 3 / 12 / 2015*

Informe

Supuestos

Para el desarrollo del trabajo se supuso que:

* El pica pica es una forma de juego que se da siempre en un juego de 3 personas por equipo.

## Modelo de dominio

Las principales clases que se desarrollaron en el modelo son *Mesa*, *Carta*, *Equipo*, *Jugador*, *Mano*, *CalculadorTanto*, *ValoresTruco*, *Mazo*, *Vuelta*, *Puntaje* con sus posibles estados, *Buenas* o *Malas*, los distintos palos según la interfaz *Palo*, y todos los estados posibles que contempla el juego como implementaciones de *EstadoJuego* (*NadaCantado*, *TantoJugado*, *TrucoCantadoTantoNoJugado*, *TrucoCantadoTantoJugado*, *TrucoQuerido*, etc.) que se mostrarán en la sección siguiente en el diagrama de clases.

La mesa es quien maneja por completo el flujo del juego, todas las acciones son referidas a la mesa, cada una de las cuales es delegada en mayor parte en las distintas instancias de *Vuelta*, que son *PrimeraVuelta*, *SegundaVuelta* y *TerceraVuelta*. Estas últimas son las encargadas de determinar el ganador de cada ronda y el final de la misma, sumándole los puntos correspondientes a cada equipo.

Los distintos estados del juego son los encargados de determinar qué se puede o no cantar, y cuántos puntos deben ser asignados a un equipo en caso de que el estado sea querido o no querido.

El mazo es conocido por la mesa y se le da la responsabilidad de repartir las cartas a todos los jugadores. Cada jugador recibe una instancia de *Mano* con tres cartas.

Aparte del modelo se encuentra la inteligencia artificial, *JugadorIA*, que hereda de *Jugador*, y posee como atributo una instancia de la mesa en la cual está jugando de manera de poder comunicarle su accionar.

Diagramas de clases

## 

*[Varios diagramas de clases, mostrando la relación estática entre las clases, pueden agregar todo el texto necesario para aclarar y explicar su diseño, recuerden que la idea de todo el documento es que quede documentado y entendible como está hecho el TP]*

Diagramas de secuencia

## 

*[Varios diagramas de secuencia, mostrando la relación dinámica entre las clases planteando una gran cantidad de escenarios que contemplen las situaciones del trabajo práctico]*

Diagrama de paquetes

## 

*[incluir un diagrama de paquetes para mostrar el acoplamiento de su trabajo ]*

Diagramas de estado

## 

*[Incluir diagramas de estados, mostrando tanto los estados como las distintas transiciones de los mismos para varias entidades del trabajo práctico ]*

Detalles de implementación

El patrón de diseño *State* fue utilizado para resolver uno de los temas más sensibles del trabajo, como lo es los posibles cantos que tiene cada jugador en un momento determinado de la ronda. Se creó la clase *Vuelta* como clase abstracta de la cual heredan *PrimeraVuelta*, *SegundaVuelta* y *TerceraVuelta*, las cuales se encargan de decidir el ganador de cada vuelta y el ganador de la ronda completa, determinan le final de la ronda y la mesa delega en ellas todos los cantos de los jugadores. A su vez, las instancias de estas clases poseen una referencia al estado del juego, que es el que sabe qué se cantó hasta el momento y qué se puede cantar.

Para el puntaje de los equipos también se utilizó el patrón  *State*, ya que el equipo puede estar en las *buenas*, o en las *malas*, y según en cuál se encuentre, responderá de manera distinta cuando se quiera determinar los puntos que el falta envido debe entregar al ganador.

Excepciones

## 

*[Explicar las excepciones creadas, con qué fin fueron creadas y cómo y dónde se las atrapa explicando qué acciones se toman al respecto una vez capturadas.]*

# 

# 

# **Checklist de corrección**

Esta sección es para uso exclusivo de los docentes, por favor no modificar.

## Carpeta

**Generalidades**

* ¿Son correctos los supuestos y extensiones?
* ¿Es prolija la presentación? (hojas del mismo tamaño, numeradas y con tipografía uniforme)

**Modelo**

* ¿Está completo?¿Contempla la totalidad del problema?
* ¿Respeta encapsulamiento?
* ¿Hace un buen uso de excepciones?
* ¿Utiliza polimorfismo en las situaciones esperadas?

## Diagramas

**Diagrama de clases**

* ¿Está completo?
* ¿Está bien utilizada la notación?

**Diagramas de secuencia**

* ¿Está completo?
* ¿Es consistente con el diagrama de clases?
* ¿Está bien utilizada la notación?

**Diagrama de estados**

* ¿Está completo?
* ¿Está bien utilizada la notación?

# **Código**

**Generalidades**

* ¿Respeta estándares de codificación?
* ¿Está correctamente documentado?