**FIUBA - 75.07**

**Algoritmos y programación III**

*Trabajo práctico 2: FonTruco*

2do cuatrimestre, 2015

(trabajo grupal)

Alumnos:

|  |  |  |
| --- | --- | --- |
| **Nombre** | **Padrón** | **PuertoEstelarTerranMail** |
| Daciuk, Alexis | 97630 |  |
| Llauro, Manuel | 95736 |  |
| Risaro, Lucas | 94335 |  |
| Rozanec, Matias | 97404 |  |

***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 pruebas 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

## 

*[Documentar todos los supuestos hechos sobre el enunciado. Asegurarse de validar con los docentes]*

* Al momento de poder cantar envido, es legal repetir el llamado infinitamente (i.e. no hay límite en cuanto a cantidad de llamadas), sin importar la jerarquía del Envido (o sea, se puede llamar Envido en respuesta a un Real Envido, que da más puntos).
* Los grupos son generados automáticamente, dividiendo a los jugadores entre ‘pares’ e ‘impares’ (por ej.: *grupo1*: jugador1, jugador3; *grupo2*: jugador2, jugador4).

## Modelo de dominio

*[Explicar los elementos más relevantes del diseño. Es decir: qué entidades se han creado, qué responsabilidades tienen asignadas, cómo se relacionan, etc]*

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

## 

*[Deben* ***detallar/explicar*** *qué estrategias utilizaron para resolver los puntos más conflictivos del trabajo práctico. (Incluida la persistencia) ]*

Un punto conflictivo del trabajo fue poder saber exactamente en qué estado se encontraba la partida, i.e. si estaba en juego una llamada de envido, si el truco/retruco/vale 4 ya fueron cantados o no, etc., para poder conocer la cantidad de puntos que habría que otorgarle a los ganadores de la ronda una finalizada cada una de ellas, y poder verificar de alguna manera (no demasiado rebuscada) si una llamada que quisiera hacer algún jugador sería válida en el contexto dado.

Para solucionar esto recurrimos al patrón de diseño *state*. Así, la mesa guarda el estado en que se encuentra, pudiendo verificarse rápidamente todo lo que se debe verificar, ya que cada estado que puede almacenar la mesa tiene guardada la información necesaria (i.e. posibles llamadas desde ese estado – por ej.: estando en truco se puede cantar retruco, pero no envido) – y el puntaje que otorga ese estado – por ej.: truco otorga 2 puntos).

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.]*

* *DonTHaveThatCardException:*
* *FirstTeamWonException:*
* *InvalidGameCallException:*
* *InvalidNumberOfPlayersException:*
* *NobodyWonYetException:*
* *NotCardThrownException:*
* *NotYourTurnException:*
* *PlayerDoesNotExistsException:*
* *SecondTeamWonException:*
* *TeamDoesntExistException:*

# 

# 

# **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?