**FIUBA - 75.07**

**Algoritmos y programación III**

*Trabajo práctico 2: AlgoCraft*

1er cuatrimestre, 2015

(Trabajo Grupal)

Alumnos:

|  |  |  |
| --- | --- | --- |
| **Nombre** | **Padrón** | **Mail** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

***Fecha de entrega final***: Miércoles 24/06/2015 - Jueves 25/06/2015

***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 interface 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

AlgoCraft es un juego ***similar*** al famoso StarCraft[[1]](#footnote-1) pero no en tiempo real sino por turnos. El mismo es un juego de guerra de estrategia y se basa en la construcción y administración de un imperio que domine todo el mapa, hasta extinguir a todos los oponentes.

## La raza

* **Terran**: Son humanos exiliados de la Tierra y hábiles en adaptarse a cualquier situación.

Gestión de recursos

Durante el juego se deberá recolectar:

1. **Minerales**: habrá nodos de minerales distribuidos por todo el mapa.
2. **Gas vespeno**: habrá volcanes que expulsan dicho gas cerca de los nodos de minerales.

Para poder colectar el gas vespeno se deberá construir una “refinería de gas” sobre el volcán dónde se origine el gas vespeno que desea extraer. Lo mismo ocurre para los minerales. Ambos colectan +10 en cada turno.

Construcciones

Los terran cuentan con:

* **Centro de mineral**: Se encarga de reunir el mineral sobre un solo cristal.
* **Barraca**: Edificio que entrenan los soldados Marine.
* **Depósito de suministros**: Determinan la cantidad máxima de unidades que se pueden construir, si se desea tener un ejército mayor se necesita construir una cantidad de Depósitos de Suministros suficientes. ( suma +5 capacidad de población)
* **Refinería:** Se construyen sobre los géisers de gas vespeno para permitir su recolección.
* **Fábrica (requiere barraca):** se construyen unidades mecánicas terrestres.
* **Puerto estelar(requiere fábrica):** aquí se construyen las unidades mecánicas aéreas.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nombre** | **Costo** | **Tiempo construcción** | **Vida** | **Unidad que habilita a construir** |
| Centro de mineral | 50 M | 4 | 500 | - |
| Barraca | 150 M | 12 | 1000 | * Marine |
| Depósito suministro | 100 M | 6 | 500 | - |
| Refinería | 100 M | 6 | 750 | - |
| Fábrica | 200 M / 100 G | 12 | 1250 | * Golliat |
| Puerto Estelar | 150 M / 100 G | 10 | 1300 | * Espectro |

Unidades[[2]](#footnote-2)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Nombre** | **Transporte** | **Costo** | **Tiempo construcción** | **Daño** | **Suministro[[3]](#footnote-3)** | **Rango Ataque** | **Vida** |
| Marine | 1 | 50 M | 3 | 6 **A**ire / 6 **T**ierra | 1 | 4 | 40 |
| Golliat | 2 | 100 M / 50 G | 6 | 10 A / 12 T | 2 | 5 A / 6 T | 125 |
| Espectro | 0 | 150 M / 100 G | 8 | 20 A / 8 T | 2 | 5 | 120 |

**Unidades mágicas.** Las unidades mágicas además de tener su vida, cargan “energía” que le permiten disparar sus magias. Para el caso de los terran, tenemos:

**Turnos**

El juego es por turnos, cada jugador elige las acciones que desea hacer en su turno y luego le pasa el turno a su contrincante.

**Jugadores**

La aplicación se juega con 2 jugadores en simultáneo, uno vs el otro. Al iniciar el juego el sistema debe consultar a los usuarios su nombre, su color y con qué raza jugará la partida.

*Validaciones*:

* Nombre de usuario debe contener por lo menos 4 caracteres.
* Jugador 2 no puede tener el mismo nombre que Jugador 1, ni el mismo color (sí la misma raza)

**Comienzo de partida**

Cada jugador empieza en una **base[[4]](#footnote-4)** y con 200 de mineral. Solo puede ver la parte del mapa en dónde se encuentra ubicado. El resto del mapa lo vé negro. A medida que sus unidades, “recorren” el mapa se irá descubriendo lo que hay en el mapa y quedará visible siempre. Los jugadores empiezan en lados opuestos del mapa.

**Mapa**

Los mapas deben tener áreas de tierra y áreas espaciales. Sólo las unidades voladoras pueden circular libremente por cualquier tipo de superficie. Los tamaños de los mapas se definen por la cantidad de bases que presenten y la distribución de las mismas deben ser equidistante al punto de origen de cada jugador. No puede haber un mapa que tenga 6 bases en total y haya 5 de una lado muy próximas a dónde comenzó un jugador..

**Población**

La capacidad de población máxima es de 200. Cada raza tiene un edificio propio que actúa en forma de “casa”. Para el caso de los terran dicho edificio es el “depósito de suministros”. Cada vez que se construye un depósito de suministros, aumenta la capacidad de población en +5. De manera contraria, si se destruye un edificio depósito de suministro, la capacidad de la población baja 5.

Los protoss poseen una construcción análoga llamada “pilón”. Cada pilón otorga también +5 en la capacidad de población.

La columna “suministro” en la descripción de unidades, indica cuántos cupos de casa ocupa la unidad construida.

**Opcionales para subir nota**

Para que cada opcional se considere hecho se tienen que cumplir todos los puntos que indican cada uno a los efectos de que las razas queden balanceadas.

1. *Agregar Raza Zerg (Mayor puntaje adicional)*

Todas las unidades y “construcciones” zerg, son seres vivos biológicos, con lo cual, cuando las mismas son dañadas, se recuperan lentamente con el paso del tiempo hasta sanar al 100 %. A diferencia de las otras razas los zerg no tienen constructores, sino que sus larvas evolucionan zángano y este ser es el que muta en forma de “construcción”.

* ***Criadero:*** *Estructura en donde nacen las Larvas. Máximo 3 larvas en simultáneo. Genera una larva nueva por turno sino tiene las 3.*
* ***Extractor:*** *Se construye en un géiser de Gas Vespeno para refinarlo y extraerlo.*
* ***Reserva de reproducción:*** *Construcción que permite evolucionar Larvas a Zerglings Es el prerrequisito para poder construir la guarida de hidralisco Zerg.*
* ***Guarida de hidralisco:*** *Construcción que permite evolucionar Larvas a Hidraliscos*
* ***Espiral:*** *Permite crear a los Mutaliscos y la reina zerg.*

***Unidades zerg***

Los zerg no crean unidades en sus “edificios”. A diferencia de los terran y los protoss, todas las unidades zerg se crean a partir de las larvas que tenga el criadero. Los “edificios” zerg habilitan a las larvas a transformarse en las nuevas unidades que ese edificio habilite.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Nombre** | **Transporte** | **Visión** | **Costo** | **Tiempo construcción** | **Daño** | **Suministro** | **Rango Ataque** | **Vida** |
| Zerling | 1 | 5 | 25 M | 2 | 5 T | 0.5 | 1 | 35 |
| Hidralisco | 2 | 6 | 75 M / 25 G | 4 | 10 A / 10 T | 1 | 4 | 80 |
| Mutalisco | 0 | 7 | 100 M / 100 G | 6 | 9 A / 9 T | 2 | 3 | 120 |
| Reina | 0 | 10 | 100 M / 100 G | 7 | 0 | 2 | 0 | 120 |
| Amo Supremo | Capacidad 8 | 9 | 100 M | 4 | 0 | 0 | 0 | 200 |

Reina: Cada turno carga +10 de energía. Tiene una capacidad máxima de 200 energía. Al crearla empieza con 50 de energía. Posee 2 magias:

**Red**: Lanza una red de baba que inmoviliza a las unidades enemigas durante 3 turnos. (Cuesta 75 energía)

**Infestar**: Destruye automáticamente a la unidad enemiga seleccionada y encima crea 2 zerling propios producto de esa infestación. (Cuesta 150 de energía)

Amo supremo: Este ser tiene 2 particularidades, sirve de transporte y también de “casa”, cada vez que se crea un amo supremo aumenta la capacidad de población en +5.

*2) Construcciones*

1. Los edificios Terran pueden despegarse del suelo y volar para ser trasladados.
2. Los zerg solo pueden construir sobre su moho, el cual se expande progresivamente desde los criaderos
3. Los protoss solo construyen sobre el radio de alcance del pilón

*3) Defensas*

1. Zergs, agregan la lengua defensiva terrestre, y las esporas aéreas
2. Protoss tienen el cañon de fotones (ataque tierra y aire)
3. Terran incorporan el bunker a ser llenados por 4 marines y la torreta aérea

*4) Unidades aéreas*

1. Terran poseen el battlecruiser, con arma yamato gun
2. Protoss tienen el carrier con hasta 8 voladores
3. Zerg: Los mutaliscos pueden evolucionar a guardian con alto alcance y se agregan nuevos seres voladores: los kamikazes

*5) Evoluciones*

* *Edificio Terran Bahía d*[*e ingeniería*](http://es.wikipedia.org/w/index.php?title=Bahía_de_ingeniería&action=edit&redlink=1)**:** Mejora el armamento y las armaduras de las unidades orgánicas o de infantería.
  + Armamento nivel 1
  + Armamento nivel 2
  + Blindaje nivel 1
  + Blindaje nivel 2
* *Zerg: Cámara de evolución: Construcción en donde se puede mejorar los niveles de desarrollo (mejorar caparazones y mejorar ataque con garras ) Nivel 1 y 2 de cada uno*
* *Edificio Protoss Fragua****:*** *A pesar de que su tecnología es increíblemente avanzada, los Protoss se están esforzando constantemente en mejorar y refinar sus armas y blindajes mediante la Fragua.*
  + *Escudo Nivel 1*
  + *Escudo Nivel 2*
  + *Armamento Nivel 1*
  + *Armamento Nivel 2*
  + *Protección vida Nivel 1*
  + *Protección vida Nivel 2*

# **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 funcionando que contemplen para cada raza:

* Pruebas unitarias al mapa, que pueda generar escenarios para que empiecen los jugadores
* Pruebas de recolección de minerales y gas con los edificios correspondientes.
* Pruebas de creación de unidades:
  + Haber construido primero el edificio que me habilite a construirla
  + Contar con recursos (mineral y/o gas) necesarios
  + Contar con límite de población para crearla

2da Entrega: **Modelo del Juego Completo** con todas las pruebas unitarias y de integración que abarquen todos los casos del enunciado, simulando batallas, interacciones, fin de partida, magias, etc.

Algunas pruebas de integración finales de ejemplo:

1) Se construye una nave de ciencia, se pasan turnos para que acumule la suficiente energía.

* A continuación se sitúa una unidad protoss, se le aplica el misil EMP. Se verifica que:
  + La nave de ciencia tenga menos energía
  + El protoss no tenga más escudo
* Se sitúa un alto templario y otra nave de ciencia (propia o no). se lanza el EMP. Se verifica que:
  + La nave de ciencia tenga menos energía
  + El alto templario y la otra nave de ciencia no tengan más energía
* Se sitúan una unidad protoss y un alto templario. Se lanza el EMP fuera del rango de alcance del mismo. Se verifica que:
  + La nave de ciencia tenga menos energía
  + El alto templario y la unidad protoss no han sufrido modificaciones.

2) Se contruye un alto templario, se pasan turnos para que acumule la suficiente energía.

* se sitúan unidades enemigas, se lanza la tormenta psiónica. Se comprueba el daño.
* Se sitúa una unidad propia y se le aplica la alucinación. Se corrobora:
  + Solo poseen escudo y no vida
  + Se les hace pelear contra un enemigo y el enemigo no recibe daño.
  + Al recibir un EMP, mueren instantáneamente.

3) Unidades de tierra no pueden atacar a unidades de aire y visceversa en todas las combinaciones, cuando corresponda.

4) Se verifica unidad por unidad, que ataque en el rango correspondiente.

5) Naves de transporte soportan su capacidad. Se debe probar que se sube unidades, desde un punto A , se pasa por espacio y se bajan del otro lado tierra B.

6) Perder unidades en batalla disminuye la población, construirlas suma.

7) No se pueden construir más unidades que el límite de población, ni que el límite máximo de población (200). Verificar que construir “más casas” no sigue subiendo el límite de 200.

8) Simular, jugar y ganar/perder una partida

9) Verificar recuperación del escudo protoss con el paso de los turnos.

10) Unidades terrestres no pueden pasar por espacios aéreos.

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 03 / 06 / 2015 - Jueves 04 / 06 / 2015*

2da Entrega: *Miércoles 10 / 06 / 2015 - Jueves 11 / 06 / 2015*

3er Entrega: *Miércoles 17 / 06 / 2015 - Jueves 18 / 06 / 2015*

4ta y última Entrega: *Miércoles 24 / 06 / 2015 - Jueves 25 / 06 / 2015*

Informe

Supuestos

## No se puede entrenar a más de una unidad por edificio por vez.

* Sólo un edificio recolector puede ocupar el espacio terrestre de una casilla con recurso.
* Cada casilla puede albergar como máximo una unidad por cada espacio (aéreo y terrestre).
* Un jugador se considera destruido (y por lo tanto fuera de juego) cuando todas sus unidades están destruidas.
* Las unidades están limitadas a un ataque por turno.
* Los edificios que dependen de otros (Fábrica-Barraca, Puerto Estelar-Fábrica) tienen su dependencia asociada a sólo un edificio. Es decir, no importa cuantas Barracas haya, si se destruye la asociada a una Fábrica, ésta deja de funcionar.
* Hay un ganador cuando todos los jugadores menos uno fueron destruidos.

## Modelo de dominio

El modelo parte de una clase Juego, que se encarga de distribuir e inicializar los distintos componentes (jugadores (a traves de la clase ListaDeJugadores), mapas), siendo además el encargado de reflejar el paso de turnos y el final de partida.

La clase Mapa es una colección de Casillas asociadas a una Coordenada. Su principal tarea es la generación de estas Casillas, asegurando la existencia de bases, recursos, etc.

La clase Jugador es, como su nombre indica, principalmente una administradora de los datos, recursos y unidades de cada jugador. Mantiene campos con distintos datos y una lista de soldados a posicionar.

Los recursos MinaDeMinerales y VolcanDeGasVespeno se encuentran modelados de una manera simple, siendo cada uno una subclase de Recurso.

Las Unidades (UnidadEdificio y UnidadMovil), están implementadas de forma también simple, guardando cada una los datos correspondientes, con la particularidad de contar con un campo PlanoAccion que permite definir los comportamientos sin preguntar si una Unidad es voladora o terrestre.

Por último, las UnidadSoldado cuentan con un campo EstadoDeAtaque, que impide que una misma unidad ataque más de una vez por turno.

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

Un punto conflictivo en el trabajo fue el poder avisarle a la vista de la casilla las modificaciones realizadas en el modelo.. Para resolverlo, se implementó el patrón de diseño Observer, en el cual, se hizo que la clase Casilla extienda de Observable, y en los métodos en los que se realizaban cambios en los atributos de la casilla que notifique a sus observadores de los cambios, mientras que se programó la clase VistaCasilla para que implemente la interfaz Observer, y por lo tanto, implementando el método update, actualizando los datos de la vista.

Otro punto conflictivo fue el posicionamiento de los distintos recolectores sobre casillas con recursos. Pasa solucionar el problema, se decidió que los recursos fueran los que agreguen los edificios a las casillas, implementando los métodos abstractos agregarCentroMineral y agregarRefineria a la clase Recurso, y en las clases heredantes, implementando estos métodos para que lancen un error en caso de que la construcción no corresponda con el recurso, y posicionando el recolector en caso contrario.

Excepciones

## 

*Las excepciones creadas fueron NoHaySoldadosParaPosicionarError, que se lanza cuando el jugador no tiene más soldados en la cola de unidades creadas. Otra de ellas es NombresInvalidos error, que se lanza cuando que intenta nombrar un jugador con el mismo nombre que otro jugador, o cuando no se ingresa nombre. Otra excepción en PoblacionLimiteAlcanzadaError, que se lanza cuando se valida la cantidad de suministros del jugador, si la suma de estos más los suministros aportados por la unidad son mayores que los suministros posibles. RecursosInsuficientesError es una excepción que se lanza que se lanza cuando se valida la cantidad de recursos del jugador, si estos son menores que el costo de las unidades. RecolectorInvalidoError es una excepción que se lanza cuando un recurso agrega en su posición una unidad de recurso que no e la que soporta. PerteneceAOtroJugadorError es una excepción que se lanza cuando que desa crear, mover o atacar con una unida de un jugador diferente al jugador de la unidad….*

# 

# 

# **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?

1. <http://es.wikipedia.org/wiki/StarCraft> [↑](#footnote-ref-1)
2. Las unidades con transporte = 0 indican unidades voladoras [↑](#footnote-ref-2)
3. Indica la cantidad de cupos de población que ocupa la unidad [↑](#footnote-ref-3)
4. Se llama base a un lugar en el mapa donde encontraremos muchos cristales juntos y un volcán de gas. [↑](#footnote-ref-4)