Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ciencias y Sistemas
Organización de Lenguajes y Compiladores I
Primer semestre 2008

Segundo Proyecto JChess

El proyecto consiste en el desarrollo de un juego de ajedrez con ciertas modificaciones en las reglas de juego que se describen mas adelante. JChess es un juego de estrategia donde a través de comandos se ejecutan acciones sobre un tablero, dichas acciones tiene como objetivo comer a todas las piezas del jugador contrario. JChess funciona en red, es decir que es necesario contar con dos computadores para poder jugar, uno para cada jugador.

1. Objetivos

1.1 General

- Que el estudiante aplique los conocimientos adquiridos sobre el proceso de compilación e interpretación en el desarrollo de un sistema computacional.

1.2 Específicos

- Que el estudiante desarrolle una aplicación donde implemente un intérprete.
- Que el estudiante aplique los conocimientos adquiridos en traducciones dirigidas por la sintaxis.
- Que el estudiante afiance sus conocimientos en analizadores sintácticos ascendentes y descendentes.
- Que el estudiante desarrolle su habilidad en la creación de gramáticas aptas para un analizador sintáctico ascendente.
- Que el estudiante amplié sus conocimientos en tecnologías Java para software distribuido.
- Que el estudiante se familiarice con las herramientas existentes para la implementación de analizadores léxicos y sintácticos.
- Que el estudiante se familiarice con el proceso de integración de módulos en el desarrollo de un producto de software.

2 Descripción del Juego

JChess es un juego de guerra que proviene del ajedrez, que requiere de mucha destreza mental y conocimientos básicos de programación. Cada jugador posee 8 piezas con diferentes capacidades de movimientos, que se mueven en un tablero cuadrado de 10 x 10 casillas, alternativamente claras y oscuras (frecuentemente blancas y negras). Las piezas de cada jugador al principio de la partida son un rey, una dama o reina, dos alfiles, dos caballos y dos torres. Las piezas de los dos bandos se distinguen por sus colores, siendo tradicionalmente blancas y negras, aunque frecuentemente se utilizan colores claros y oscuros o incluso dos colores cualesquiera distintos que no necesariamente tienen que ver con los del tablero. A las piezas más claras, se las suele establecer como *las blancas*, siendo las más oscuras entonces, *las negras*. Para nombrar a los jugadores, se suele utilizar también los nombres *blanco* y *negro*, de acuerdo con las piezas que conducen.

El tablero debe colocarse de una forma al azar al empezar la partida, a diferencia del ajedrez tradicional en el que tiene un orden específico, acá las piezas pueden colocarse en las posiciones que el software decida, únicamente respetando una pieza por casilla. Existe una excepción con el alfil, que en el ajedrez tradicional debe haber uno en una casilla negra y otro en una casilla blanca; acá el programa podría colocarlo en la posición que quiera, pudiendo tener los dos ya sea en las blancas o en las negras.

Se juega por turnos. Cada jugador intenta obtener ciertas ventajas en su posición, que frecuentemente consisten en la captura de las piezas del contrario (*ganar material*). Sin embargo, la victoria **JChess** no se obtiene mediante la captura del rey sino que de todas las piezas del contrario. De igual forma, en este juego la partida puede queda empatada (lo que se conoce como **hacer tablas**).

Por tanto, capturar las piezas del contrario es el objetivo. La pérdida por abandono también es permitida. La razón es que es frecuente encontrarse en posiciones en las que el mate es inminente o las pérdidas de material son tan importantes que la partida está inexorablemente perdida frente a un jugador suficientemente experto. Por ello, el abandono se considera una muestra de respeto al contrario y, en cambio, forzar el alargamiento innecesario de la partida, hasta que se terminan las piezas, una muestra de mala educación.

3 Descripción Técnica

El juego será en una única computadora que tendrá una shell (consola) en la que se introducen los comandos.

Los turnos serán uno por jugador cada vez. Es decir, el jugador A espera hasta que el jugador B tire para poder realizar su turno, los turnos deben ser excluyentes. Cada vez que un jugador realice un tiro los cambios en el tablero deben visualizarse, así como el reloj y el puntaje.

Cada vez que un jugador haga uso de su turno e ingrese los comandos en la Shell, se analizará la cadena y luego se interpretará dicha cadena enviada por el jugador en turno y se realizaran los cambios sobre las interfaces de los jugadores.

El juego debe tener las opciones de Abrir y Guardar y llevar una bitácora de los comandos de manera que si se desea Abrir nuevamente el juego, el juego debe cargar esa bitácora con todos los comandos.

4 Resumen de las Reglas del Juego

Cuando el juego comienza, un jugador controla 8 piezas blancas y otro jugador controla 8 piezas negras. El color de cada jugador puede elegirse por común acuerdo, por azar. Las piezas se mostrarán al azar por el programa y ambos jugadores deben estar de acuerdo de jugar en ese tablero, de lo contrario se mostrará otro tablero.

Cada jugador cuenta con una consola de interpretación de comandos en la que programarán los movimientos de sus piezas y sus estrategias. Los jugadores mueven por turnos, en cada turno un jugador puede mover hasta 3 piezas al mismo tiempo. Las piezas pueden moverse a una casilla vacía o a una casilla ocupada por una pieza del contrario, que entonces será capturada, si el jugador mueve a una casilla en la que se encuentra una pieza propia, esa pieza que está moviéndose intercambia lugar con la otra. El jugador que juega con las piezas blancas es siempre el que mueve primero. Ello le concede una pequeña pero sustancial ventaja. Cada tipo de pieza se mueve de una forma diferente. La torre se mueve cualquier cantidad de casillas libres, pero sólo horizontal o verticalmente, mientras que el alfil se mueve cualquier cantidad de casillas libres, pero sólo diagonalmente. De esta forma, un alfil siempre se moverá por casillas de un solo color. Ya se mencionaba al principio que dependiendo el tablero podría haber alfiles que se muevan por el mismo color siempre y cuando los usuarios hayan aceptado jugar en ese tablero. La dama o reina combina los movimientos de la torre y el alfil, pudiéndose mover horizontal, vertical y diagonalmente tantas casillas libres como desee. El rey puede moverse solamente una casilla horizontal, vertical o diagonalmente, sin embargo posee la cualidad de ser la única pieza en este juego que si es capturado y en el siguiente turno es capturada la pieza que lo capturó, el rey regresa al juego en la casilla donde fue capturado. Finalmente, el caballo es la única pieza que puede saltar, es decir, que puede ir de la casilla de inicio a la de destino sin que se lo impida una pieza interpuesta; el caballo se mueve en forma de 'L': las casillas de origen y destino distan dos casillas horizontales más una vertical (o viceversa).

Con la única excepción del movimiento del caballo, las piezas de un bando no pueden saltar una por encima de la otra y una pieza propia puede reemplazar a otra de las nuestras en una casilla si en su trayectoria interfiere una con la otra. Las piezas contrarias tampoco pueden ser saltadas, pero sí podemos ocupar una casilla que previamente estaba ocupada por una pieza del contrario, precisamente al capturarla. Entonces, nuestra pieza ocupará esa casilla y la pieza del contrario se retirará del juego.

Un juego de ajedrez no tiene porqué terminar siempre al quedar un jugador sin piezas, cada jugador puede rendirse (*abandonar*) si la situación está claramente perdida. El juego también puede terminar en tablas (empate), en varias situaciones, como son las siguientes:

- Cuando los jugadores acuerdan tablas, bien por apreciar que la partida no puede ganarse, o bien porque no desean continuar la lucha

- Por la incapacidad de comer todas las piezas. Esto normalmente es debido a la falta de piezas apropiadas para dar mate.
- Por la incapacidad de terminar las piezas del contrario, si en 50 jugadas consecutivas no se ha capturado ninguna pieza. Esta es una norma introducida para evitar que las partidas se alarguen indefinidamente en situaciones donde un bando puede ganar pero no sabe cómo o simplemente cree que puede ganar y no es cierto.
- Por llegar a una situación en la que la misma jugada se repite una y otra vez (*repetición de jugadas*)

A efectos de contabilizar los resultados en un torneo, al jugador que vence en una partida, se le otorga 1 punto y al que pierde 0 puntos. Si son tablas, cada jugador recibe 1/2 punto.

5 Comandos del intérprete para mover las piezas

La destreza en la programación de cada jugador es la clave de la victoria, para mover cada pieza el jugador cuenta con estructuras de control y objetos que tienen sus métodos propios descritos a continuación:

5.1 Objetos disponibles

Cada uno de los objetos cuenta con los siguientes atributos y métodos:

5.1.1 Atributos

- **PosX.** Indica la posición en las filas del tablero en que se encuentra la pieza, es tipo int.
- **PosY**. Indica la posición en las columnas del tablero en que se encuentra la pieza, es tipo int.
- **Name.** Indica el nombre del objeto, por default el juego asigna “Torre1”, “Torre2”… etc. Sin embargo, cada jugador puede cambiar el nombre del objeto a su preferencia es tipo string.
- **Picture.** Permite al usuario cargar una imagen distinta de la pieza la cuál podrá ser vista únicamente por el usuario que la ha cambiado es un string que contiene el path de la imagen cargada.

5.1.2 Métodos

- **Move(int x,int y);**

  Indica a la pieza que debe moverse a la posición x,y que

se le ha indicado.

- **MoveTo(string o);**

  Indica que debe moverse hacia una pieza propia para intercambiarla. Para que sea válida la instrucción, el movimiento debe ser un movimiento válido, el parámetro o es el nombre de la pieza a la que se dirigirá. Ejemplo: Torre1.MoveTo(“Rey1”);

- **MoveToward(string o, int x);**

  Indica que debe moverse hacia una pieza propia pero solo cierto número de casillas. Para que sea válida la instrucción, el movimiento debe ser un movimiento válido, el parámetro o es el nombre de la pieza a la que se dirigirá y el parámetro x es la cantidad de espacios. Ejemplo: Alfil1.MoveToward(“Torre2”,2);

- **MoveAwayFrom(string o, int x);**

  Indica que debe moverse en dirección contraria a una pieza propia pero solo cierto número de casillas. Para que sea válida la instrucción, el movimiento debe ser un movimiento válido, el parámetro o es el nombre de la pieza a la que se dirigirá y el parámetro x es la cantidad de espacios. Ejemplo: Alfil1.MoveAwayFrom(“Torre2”,2);

6 Estructuras de control

- **Do nothing;**

  Con esta instrucción el jugador elige no hacer nada en su turno.

- **Do in order**

  …

  Do_end;

  Las acciones que ingresemos adentro de la estructura se realizarán una por una, recordando siempre que el número máximo de piezas que se pueden mover por turno es 3. Cuando hay un intercambio de piezas, dos piezas son movidas, por lo tanto solo se puede mover una más.

- **Do togheter**

…

Do_end;

Las acciones que ingresemos adentro de la estructura, si son de más de una pieza, se realizarán en paralelo, recordando siempre que el número máximo de piezas que se pueden mover por turno es 3.

- **Condicionales**

  If condición **if confición**

  Else **end if**

  End if

  La condición recibirá los operadores lógicos >,<, >=, ==, <=, !=, &&, y ||. Además de poder incluir los operadores aritméticos +, -, * para condiciones como **if Torre1.PosX+Torre1.Pos == 8 …**

- **Ciclo Loop Times**

  Loop CANTIDAD times

  …

  End_loop

  Esta estructura encicla las acciones la cantidad de veces que se ingresa. Si cantidad se escribe la palabra reservada “Infinity”, se realizará infinitas veces hasta que coma una pieza o realice 50 jugadas sin comer una pieza.

- **Ciclo While**

  While condición

  …

  Wend

  Esta estructura encicla las acciones mientras la condición sea verdadera, no es válido instrucciones como “While True … wend”.

- **Print**

  Print(String s);

Le envía al otro jugador un mensaje que le aparecerá en su consola de comandos, el string puede contener cualquier caracter. Ejemplo: Print(“Te voy a gan@r”);

- **Stats();**

  Muestra una gráfica de los juegos ganados por cada usuario (únicamente dos usuarios: blancas y negras).

Cada jugador cuenta con 1 minuto máximo para escribir sus jugadas. De lo contrario la Shell por si misma enviará un “do nothing” haciendo que el jugador pierda el turno.

Se deben mostrar dos relojes, uno que indique el tiempo del turno del jugador (indistintamente del jugador), que se reinicia cada turno y otro del tiempo total del juego. Además de eso, el Score de juegos entre ambos jugadores.

JChess cuenta con la opción de jugar contra un oponente. Es decir, para poder jugar se necesitan de dos jugadores. Por ejemplo, se tiene dos jugadores: jugador A y jugador B. El jugador A en un computador tendrá en su pantalla el tablero de juego, el mismo tablero para el jugador B. En la consola de movimientos el jugador A puede realizar su ataque contra el jugador B y este ataque tendrá efecto tanto en el tablero del jugador A como el del jugador B. Así mismo, al turno del jugador B podrá realizar su ataque y tendrá efecto sobre ambos tableros. Todos los movimientos tendrán que ajustarse a las reglas del juego definidas anteriormente.

Manejo de Errores:

Errores Léxicos:

Caracteres no incluidos en el alfabeto del lenguaje o palabras reservadas no definidas entre los componentes léxicos.

Ejemplo:

Alfil1.Mover(1,1);

@lfil1.Move(1.1);

Errores Sintácticos:

Componentes léxicos aceptados por el lexer pero que no son aceptados por la gramatica del lenguaje.

Alfil1.move(1,8) // Falta punto y coma

Torre1.move(1.3); // Se esperaba , pero se encontro “ . “

Errores Semánticos:

Errores que léxica y gramaticalmente están correctos pero en la semántica tienen ciertas validaciones para que sea coherente.

Ejemplo:

Alfil1.Move(15,20); // El tablero es de 10 x 10

Archivo de Salida:

El proyecto deberá mostrar por cada instrucción un archivo de salida con la información de las jugadas que se realizaron.

Ejemplo:

Instrucción: Alfil1.move(1,5);

Salida: “Jugador 1 ha movido el Alfil1 a la posición 1,5”

El CD deberá contener:

- Archivo Lex
- Archivo Cup
- Archivo de salida del proyecto
- Archivo jar ejecutable compilado
- Todas las clases generadas por Java lex y cup (scanner, parser, sym, etc)
- Un archivo de entrada que hayan utilizado ustedes y que sepan que funciona
- Archivos de salida generados con sus archivos de prueba

Notas Importantes del Proyecto

- El proyecto se entrega el Lunes 28/Abril/2008
- El proyecto es individual.
- Los archivos serán parseados con Jlex & Cup en GNU/Linux.
- La interfaz gráfica debe ser creada en Java.
- La calificación será personal y durará como máximo 30 minutos, en un horario que se entregará en fechas cercanas a la entrega.
- Durante la calificación no podrán estar terceras personas alrededor o de lo contrario no se calificará el proyecto.
- No se aceptarán modificaciones durante la calificación del proyecto.
- No se aceptará la ejecución de procesos en consola a excepción del archivo ejecutable de la aplicación.

- COPIAS de proyectos tendrán una nota de CERO puntos y el correspondiente reporte ante la Escuela de Sistemas quienes se encargarán de esto.
- Al momento de ejecutar el código, la aplicación deberá ser capaz de realizar la llamada al compilador y ejecutar el código. No se permitirá hacer llamadas al compilador de forma manual para compilar el código.
- No se aceptarán proyectos los cuales únicamente manejan errores y mucho menos proyectos que solo manejen archivos ideales (sin errores).
- Para tener derecho a la calificación del proyecto, el proyecto debe ser completamente funcional en el sistema operativo GNU/Linux.
- **Usted deberá indicar claramente el porcentaje de funcionalidad que realiza su aplicación al momento de la entrega. Deberá marcar su fólder que contiene su CD con las siguientes etiquetas:**

  75%-100%

  50%-75%

  25%-50%

  1%-25%
- El CD deberá ser entregado en un fólder junto con la hoja de calificación. En caso contrario el proyecto no será recibido.

  - Color Negro para la sección “A”

  - Color Azul para la sección “B”

  - Color Rojo para la seccion “C”
- Si no se presenta alguno de los anteriores requisitos NO se calificará el proyecto.