Proyecto 03
"Retrospectiva"

Universidad Nacional Autónoma de México
Facultad de Ciencias
Modelado y Programación

Sebastián Alamina Ramírez
Vianey Aileen Borrás Pablo
Kevin Axel Prestegui Ramos

Octubre,2019

Contents

1 Introducción

El presente proyecto es sobre un programa de ajedrez que trabajamos en nuestro primer semestre, el cual fue elaborada en Java 8. Decidimos trabajar con este proyecto debido a las múltiples mejoras que se pueden agregar para mejorar su funcionamiento, así como ser más amable para el usuario. Además de que todo esto será programado con un nuevo lenguaje, Ruby.

2 Definición del problema

Se necesita programar un ajedrez que pueda calcular las casillas a las cuales se pueden mover las piezas de un juego de ajedrez, considerando los casos cuando capturarán a una pieza enemiga, también se necesitará modelar el tablero y las piezas, por lo que necesitaremos una interfaz gráfica que permita realizar lo anterior.

3 Análisis del problema

Dado el programa anterior que se hizo en el primer semestre, se propuso mejorarlo con respecto a las posiciones válidas de las piezas, ya que anteriormente el programa marcaba posibles movimientos que lógicamente son correctos, pero que están fuera del tablero, como movimientos válidos (ej[-2,i]).

Como el programa será interactivo crearemos una interfaz gráfica, la cual contenga el tablero así como todas las piezas tanto blancas como negras del ajedrez.

4 Selección de la mejor alternativa

Para este proyecto decidimos usar Ruby, ya que Ruby es un lenguaje de programación interpretado, reflexivo y orientado a objetos, además de que su sintáxis es muy parecida a Python.
Como Ruby es un lenguaje orientado a objetos, nos permite que todos los tipos de datos sean un objeto, incluidos las clases y toda función es un método.
Además de que el código quedá de manera más limpia (clean code) y ordenado por lo que facilita la lectura del código.

Para crear la interfaz se ocupara Shoes que es un kit de herramientas GUI basado en el lenguaje de programación de Ruby, esta diseñado para hacer que las aplicaciones sean lo más fácil posibles.Shoes puede ser ejecutado en Microsoft Windows, Mac OS y Linux.

5 Pseudocódigo

```
Inicio
  Repetir
   // mostrar menú
   Imprimir " 1) Saber si cierto movimiento es válido."
   Imprimir " 2) Conocer todos los movimientos válidos."
   Imprimir " 3) Mover de lugar a la pieza ( a cualquier
              parte del tablero.)"
   Imprimir " 4) Crear otra pieza (y descartar esta)."
   Imprimir " 5) Salir."
   Imprimir "¿Qué deseas hacer: "
   Si opción < 1 || opción > 5
      Imprimir "Número incorrecto para el renglón."
      Imprimir "El valor debe ser entre 1 y 5."
      Imprimir "Introduce un valor válido."
   Leer opción
   // procesar esa opción
   Segun opción Hacer
      1 : .
         Si es válido
            Imprimir "Ese movimiento sí es válido."
         Sino es válido
            Imprimir "Ese movimiento no es válido."
         FinSi
      2 : .
         Imprimir "Los posibles movimientos son: "
      3 : .
         Imprimir "Dame la columna deseada: "
            Si columnaDada < 'a' || columnaDada > 'h'
               Imprimir "Carácter incorrecto para la
                         columna"
            FinSi
            Imprimir "El valor debe ser entre 'a' y 'h' "
            Imprimir "Introduce un valor válido"
         Imprimir "Dame el renglón deseado: "
            Si renglónDado < 1 || renglónDado > 8
               Imprimir "Número incorrecto para el
                         renglón"
            FinSi
            Imprimir "El valor debe ser entre 1 y 8 "
            Imprimir "Introduce un valor válido"
         Imprime  imprimeTablero(pieza)

      4 : .
         Imprimir "Vamos a crear una pieza..."
          Creamos la pieza y la devolvemos
      5:.
         Salimos del menú
   FinSegun
  Hasta Que opcion = 5
Fin
```

Pseudocódigo: Proyecto pasado

Inicio
1. Construimos el tablero.
2. Inicializamos el tablero para la interfaz.
2.1 Agregamos las casillas al tablero.
2.2 Pintamos las casillas (blanco o café).
3. Inicializamos la pieza.
3.1 Creamos la pieza.
3.1.1 Asignamos las imágenes correspondientes a las piezas.
3.3 Indicamos el lugar en la que se localiza.
3.4 Asignamos el color correspondiente (blanco o negro.
4. Agregamos la pieza al tablero.
4.1 Creamos la pieza.
5. Para cada click que se haga sobre una casilla sin una pieza, no hacemos nada.
6. Para cada click que se haga sobre una casilla con una pieza.
6.1 Indicamos todos los posibles movimientos de la pieza seleccionada.
6.2 Dibujamos sobre el tablero los posibles movimientos.
Fin

Pseudocódigo: Proyecto mejorado

6 Pruebas y Mejoras

6.1 Pruebas

- MainTest
  Ejecuta todas las pruebas de las piezas (Peón, Torre, Caballo, Alfil, Reina y Rey)

- TestAlfil
  En esta prueba se prueban los movimientos que la pieza debería poder realizar, incluyendo así los movimientos no válidos,y que la pieza no salga del tablero.

- TestCaballo
  Se prueban los movimientos que la pieza Caballo debería poder realizar, incluyendo así los movimientos no válidos,y que la pieza no salga del tablero.

- TestPeon
  Se prueban los movimientos que la pieza Peón debería poder realizar, incluyendo así los movimientos no válidos,y que la pieza no salga del tablero.

- TestReina
  Se prueban los movimientos que la pieza Reina debería poder realizar, incluyendo así los movimientos no válidos,y que la pieza no salga del tablero.

- TestRey
  se prueban los movimientos que la pieza Rey debería poder realizar, incluyendo así los movimientos no válidos,y que la pieza no salga del tablero.

- TestTorre
  se prueban los movimientos que la pieza Torre debería poder realizar, incluyendo así los movimientos no válidos,y que la pieza no salga del tablero.

- TestMovimientos
  En este prueba, probaremos los movimientos compuestos por los saltaos hacía la izquierda/derecha y/o arriba/abajo de diferentes magnitudes.

- TestPieza
  Prueba que usaremos para probar las piezas en: TestAlfil, TestCaballo, TestPeon, TestReina, TestRey, y TestTorre.

6.2 Mejoras

El proyecto incluye las siguientes mejoras:

- Posiciones válidas
  El programa devuelve las posiciones a las que se pueden mover las piezas (*color verde*) incluyendo los casos en el que puedan capturar a una pieza enemiga.

- Añadir tablero y piezas (*blancas y negras*) del ajedrez
  Para este proyecto decidimos agregar las piezas y el tablero para un mejor diseño y una mejor presentación del programa, ya que anteriormente la información solamente se mostraba desde la terminal.

- Información correcta
  Otra mejora para el proyecto es respecto a la información que se proporciona, pues como mencionábamos anteriormente el programa a pesar de devolver posiciones válidas para las piezas también devolvía posiciones que lógicamente son correctas, pero están fuera del tablero. Así que para este proyecto se decidió arreglar esa información regresando posiciones válidas que sólo están dentro del tablero.

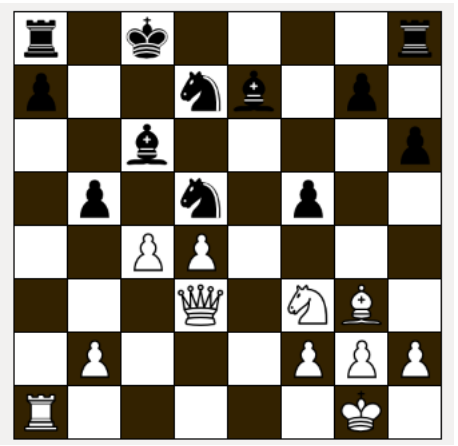

Fig1:Tablero y Piezas

Fig2:Movimientos válidos

7 Análisis estadístico

Debido a que el programa es interactivo, no vimos la necesidad de crear el archivo de bash, ya que nuestro programa no tarda un tiempo preciso en ejecutarse y básicamente todo dependerá de lo que el usuario deseé hacer.

8 Situación extraordinaria

Una situación extraordinaria con la que tuvimos que lidiar, fue respecto a los movimientos del caballo, pues al momento de probarlo con las pruebas nos dimos cuenta de que el programa no calculaba bien los movimientos del caballo, por lo que tuvimos que hallar la manera de solucionarlo para este nuevo proyecto, si no hubieran sido por las pruebas el problema aún seguiría y fue ahí donde nos dimos cuenta de la importancia del porque siempre deben ir primero las pruebas antes que todo.

9 Plan a futuro

En un futuro el programa podría mejorarse en:

- Hacer un menú para que el usuario pueda elegir entre las distintas modalidades (usuario vs usuario, usuario vs computadora, contrarreloj, etc) y distintos niveles (principiante, intermedio, avanzado) en las que desea jugar.

- Las piezas puedan moverse, es decir que el juego pueda ser jugable, pues en este proyecto solo mostramos el tablero y las piezas junto con sus movimientos válidos.
- Para lograr que las piezas puedan moverse y el ajedrez pueda ser jugable, necesitaríamos del uso inteligencia artificial y redes para lograrlo.
- Hacer una interfaz más amigable para el usuario.

Se llegó a la conclusión de que un costo aceptable para este programa será de $2,500.00

10 División del trabajo

El diseño del programa fue discutido entre los miembros del equipo, de tal manera que un integrante se encargaría de programar las piezas(*movimientos*) y la interfaz, mientras que las pruebas fuerón trabajadas en conjunto entre los otros dos miembros del equipo.