   
   
  **Unidad Profesional Interdisciplinaria de Ingeniería Campus Zacatecas (UPIIZ)**

**REPORTE DEL EJERCICIO DEL PROBLEMA DEL CABALLO**

**Materia: Analisis de Algoritmos**

**Instituto Politécnico Nacional**

**Fecha:11/09/2019**

**Alumno: José Moisés Luna Montes  
 No.Boleta: 2017670761**

**Ingeniería en Sistemas Computacionales**

**Grupo:3cm1**

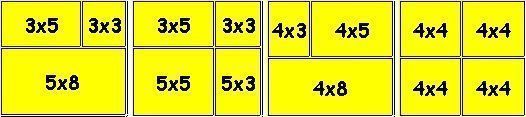
**MARCO TEORICO**

El problema del caballo es un antiguo problema matemático en el que se pide que teniendo una cuadrícula de n x n casillas y un [caballo de ajedrez](https://es.wikipedia.org/wiki/Caballo_(ajedrez)) colocado en una posición cualquiera ( x, y ), el caballo debe pasar por todas las casillas con sus movimientos ya establecidos y una sola vez. Lo que resulta en n2-1 movimientos.

El trabajo más importante en relación a este problema, se atribuye al genial [Leonhard Euler](http://soymatematicas.com/que-sabes-de-euler/), que destacó por sus ingeniosas y fantásticas soluciones.  
Una de las soluciones que dio este genio matemático asombró por su belleza. Euler construyó un cuadrado mágico donde las filas y las columnas sumaban 260. El caballo se desplaza desde la casilla 1 hasta la 64 en orden numérico. Puedes comprobarlo en la siguiente imagen.

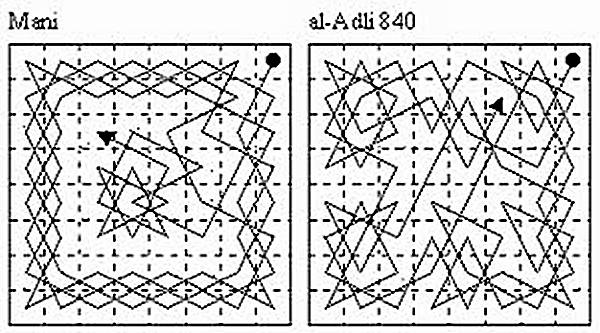
Si ya de por sí, el desarrollo de la marcha del caballo por todo el tablero es muy difícil de conseguir, añádele además conseguir un [cuadrado mágico](http://soymatematicas.com/cuadrados-magicos/).

Siempre ayuda dividir un problema en pequeñas partes. Una buena estrategia inicial sería dividir el tablero en pequeñas porciones. Hay que tener claro que recorridos son posibles y enlazarlos hasta completar el tablero.



Aunque Euler planteó principalmente recorridos cerrados, que son más elegantes y además permiten dar solución al problema desde cualquier casilla inicial,  para obtener el cuadrado mágico estableció algunas pautas para la resolución general de este problema.

Una de las primeras soluciones conocidas data del siglo IX. En efecto, en un manuscrito del árabe Abu Zakariya Yahya ben Ibrahim al-Hakim se encuentran documentados dos recorridos válidos. Uno de ellos pertenece a un jugador de ajedrez llamado Ali C. Mani y el otro a Al-Adli ar-Rumi, un aficionado del que se sabe también escribió un libro sobre una forma de [ajedrez popular](https://www.neoteo.com/kvetka-la-mejor-manera-de-analizar-una-partida-de-ajedrez/) por esa época llamado “Shatranj”.



En 1995, Martin Löbbing e Ingo Wegener pusieron a trabajar 20 ordenadores Sun -potentes para la época- durante cuatro meses y publicaron un documento en el que proclamaban que el número de recorridos posibles en un tablero de 8×8 era 33.439.123.484.294. Dos años más tarde, en 1997, Brendan McKay encaró el problema del caballo dividiendo el tablero en dos mitades y llego a un resultado algo menor: “solo” existirían 13.267.364.410.532 recorridos posibles. Para tener una idea de lo que significan estos números, basta saber que si un robot fuese capaz de mover el caballo para que complete un recorrido por segundo, demoraría más de 420 años en probarlos a todos.

**DESARROLLO**

Recursos necesarios :  
\*Maquina con neatbeans, eclipse u otro entorno de programación.   
\*Tener instalado Java con sus diversas rutas en path.

**Código en java:**

**package caballo;**

**/\*\***

**\***

**\* @author Mouxes**

**\*/**

**public class Caballo {**

**static int[][] tablero;**

**public int x;//valor del tamaño en el eje vertical**

**public int y;//valor del tamaño en el eje horizontal**

**public Caballo(int x, int y) {**

**tablero = new int[x][y]; //declaramos el tamaño del tablero de tamaño x\*y**

**this.x = x;**

**this.y = y;**

**//llenamos el tablero de cero**

**for (int i = 0; i < x; i++) {**

**for (int j = 0; j < y; j++) {**

**tablero[i][j] = 0;**

**}**

**}**

**}**

**boolean valorValido(int vert, int hor) { //validador de si los valores estan dentro de la tabla**

**if (vert < 0 || vert > x - 1 || hor < 0 || hor > y - 1) { //**

**return false;**

**}**

**if (tablero[vert][hor] != 0) {**

**return false;**

**}**

**return true;**

**}**

**public boolean rutaDelCaballo(int vert, int hor, int num) {//metodo que llama los demás metodos y llena el tablero**

**if (valorValido(vert, hor)) {**

**tablero[vert][hor] = num;//si la inicionado inicial en las cordenadas vert,hor esta dentro del tablero entonces toma el valor de num**

**}**

**int[][] posibles = obtSaltosPosibles(vert, hor);//sacamos los posibles saltos que dara el caballo**

**ordenarSaltos(posibles);//ordenamos los posibles saltos dependiendo de los posibles saltos que tengan.**

**for (int i = 0; i < posibles.length; i++) {**

**if (rutaDelCaballo(posibles[i][0], posibles[i][1], num + 1)) {**

**return true; //a partir de ya haber sacado los saltos del caballo haremos un tipo de recursividad que genere las siguientes nibilidades para rellenar el tablero**

**}**

**}**

**if (num == x \* y) {**

**return true; // Si el num toma el valor de x\*y este ya recorrio toda el tablero y se termina**

**}**

**return false;**

**}**

**int[][] obtSaltosPosibles(int vert, int hor) {**

**//se sacaran los diversos saltos a partir de la forma en la que nos enseño el profesor de la clases de analisis de Algoritmos**

**/\***

**| | | | | | | |**

**| | | | | | | |**

**| | | 3 | | 4 | | |**

**| | 2 | | | | 5 | |**

**| | | | 1 | | | |**

**| | 9 | | | | 6 | |**

**| | | 8 | | 7 | | |**

**| | | | | | | |**

**| | | | | | | |**

**\*/**

**int[][] valor = new int[8][2]; //guardamos los Saltos en una matriz de 8 por 2 para guardar a donde saltara a partir de la ubicacion**

**int n = 0;**

**if (valorValido(vert - 1, hor - 2)) {**

**valor[n][0] = vert - 1;**

**valor[n++][1] = hor - 2;**

**}**

**if (valorValido(vert - 2, hor - 1)) {**

**valor[n][0] = vert - 2;**

**valor[n++][1] = hor - 1;**

**}**

**if (valorValido(vert - 2, hor + 1)) {**

**valor[n][0] = vert - 2;**

**valor[n++][1] = hor + 1;**

**}**

**if (valorValido(vert - 1, hor + 2)) {**

**valor[n][0] = vert - 1;**

**valor[n++][1] = hor + 2;**

**}**

**if (valorValido(vert + 1, hor + 2)) {**

**valor[n][0] = vert + 1;**

**valor[n++][1] = hor + 2;**

**}**

**if (valorValido(vert + 2, hor + 1)) {**

**valor[n][0] = vert + 2;**

**valor[n++][1] = hor + 1;**

**}**

**if (valorValido(vert + 2, hor - 1)) {**

**valor[n][0] = vert + 2;**

**valor[n++][1] = hor - 1;**

**}**

**if (valorValido(vert + 1, hor - 2)) {**

**valor[n][0] = vert + 1;**

**valor[n++][1] = hor - 2;**

**}**

**int[][] aux = new int[n][2]; //declaramos un auxiliar de tamaño n \* 2 que nos ayude a guardarlos**

**for (int i = 0; i < n; i++) { //guardamos con un for**

**aux[i][0] = valor[i][0];**

**aux[i][1] = valor[i][1];**

**}**

**return aux; //returnamos el aux**

**}**

**void ordenarSaltos(int[][] posibles) { //ordenaremos los saltos dependiendo de la cantidad posibles Saltos que tiene de menor a mayor**

**for (int i = 0; i < posibles.length; i++) {**

**for (int j = i + 1; j < posibles.length; j++) {**

**int cantSaltosI = obtSaltosPosibles(posibles[i][0], posibles[i][1]).length; //Sacamos la cantidad de posibles saltos**

**int cantSaltosj = obtSaltosPosibles(posibles[j][0], posibles[j][1]).length; //Sacamos la cantidad de posibles saltos de la posicion i+1**

**if (cantSaltosj < cantSaltosI) { //comparamos una posicion con la otra y si la posicion i+1 tiene menor cantidad que i entonces cambiamos la posicion a la derecha**

**int aux0 = posibles[i][0];**

**int aux1 = posibles[i][1];**

**posibles[i][1] = posibles[j][1];**

**posibles[j][1] = aux1;**

**posibles[i][0] = posibles[j][0];**

**posibles[j][0] = aux0;**

**}**

**}**

**}**

**}**

**public void mostrarTablero() {//mostramos el tablero**

**for (int i = 0; i < x; i++) {**

**for (int j = 0; j < y; j++) {**

**System.out.printf(" %2d ", tablero[i][j]);//es para mostrar en 2d**

**}**

**System.out.println("");**

**}**

**}**

**public static void main(String[] args) {**

**int x=8;**

**int y=8;**

**for (int i = 0; i < x; i++) {**

**for (int j = 0; j < y; j++) {**

**Caballo cab = new Caballo(x, y);// vamos al constructor**

**cab.rutaDelCaballo(i, j, 1);//ejecutara el metodo**

**cab.mostrarTablero();//llamamos el metodo para mostrar el tablero**

**System.out.println("\n");**

**}**

**}**

**}**

**}**

BIBLIOGRAFIA

<https://www.neoteo.com/el-problema-del-caballo/>[Ariel Palazzesi](https://www.neoteo.com/miembros/ariel-palazzesi/profile/home/)

<https://soymatematicas.com/problema-del-caballo/> SOYMATEMATICAS/Justo Fernandez/2017

<https://www.youtube.com/watch?v=fMrwk6uf6-c> Enrique Negrete Lopez/8 Marzo 2011