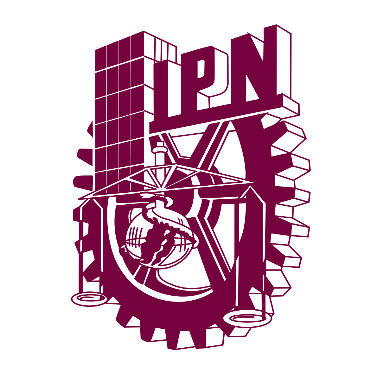
****

**Instituto Politécnico Nacional**

**Escuela Superior de Computo**

*Alumno:*

* Monroy Ramírez Oscar G.

***Grupo:*** *2CM23*

***Unidad de Aprendizaje:*** *Algoritmos y Estructuras de Datos*

***Evidencia:*** *Proyecto De Arboles MINIMAX*

***Docente:*** *De Luna Caballero Roberto*

***Fecha****: 13 de enero de 2021*

Contenido

[Introducción 4](#_Toc59050266)

[Marco Teórico 5](#_Toc59050267)

[Recursividad 5](#_Toc59050268)

[Ejemplo sencillo 5](#_Toc59050269)

[Arboles Binarios (Arboles B) 7](#_Toc59050270)

[Tipos de árboles binarios 7](#_Toc59050271)

[Implementación en C 8](#_Toc59050272)

[Estructura con manejo de memoria dinámica, siendo el puntero que apunta al árbol de tipo Arbol: 8](#_Toc59050273)

[Estructura con arreglo indexado: 8](#_Toc59050274)

[Recorridos sobre árboles binarios 8](#_Toc59050275)

[Recorridos en profundidad 8](#_Toc59050276)

[Recorrido en postorden 9](#_Toc59050277)

[Recorrido en inorden 9](#_Toc59050278)

[Recorridos en amplitud (o por niveles) 10](#_Toc59050279)

[Creación de árboles a partir de los recorridos 11](#_Toc59050280)

[Métodos para almacenar Árboles Binarios 11](#_Toc59050281)

[Codificación de árboles n-arios como árboles binarios 12](#_Toc59050282)

[Arboles Binarios de búsqueda (Arboles ABB) 13](#_Toc59050283)

[ANÁLISIS DE LA EFICIENCIA DE LAS OPERACIONES. 16](#_Toc59050284)

[Ejemplo de ARBOL ABB programado por OGMR 18](#_Toc59050285)

[Operación del programa (Inserción Manual): 36](#_Toc59050286)

[Conclusion. 38](#_Toc59050292)

[Referencias Bibliográficas: 38](#_Toc59050293)

# Introducción

Una computadora es una maquina que maneja información. El estudio de la ciencia de la computación incluye saber la forma en que se organiza la información en una computadora, como puede manejarse y la manera en que puede utilizarse esa información. Por tanto, es de suma importancia para un estudiante de la computación, entender los conceptos de organización y manejo de la información para continuar con el estudio de nuestra diciplina.

Si la ciencia de la computación es fundamental el estudio de la información, la primera pregunta que surge es: ¿qué es la información? Por desgracia, aunque ese concepto es la piedra angular de la diciplina, la interrogante anterior no puede contestar con precisión. En que sentido, el concepto de la información en esta diciplina es similar a los conceptos de punto, recta y plano para la geometría: son un grupo de términos indefinidos con los que se pueden hacer proposiciones, pero no pueden definirse con conceptos más elementales.

En la geometría es posible hablar de la longitud de una recta, a pesar de que el concepto de recta es en si mismo indefinido. La longitud de una recta es una medida cuantitativa. De manera similar, en la ciencia de la computación podemos hablar de cantidades de información. La unidad básica de información es el ***bit*** cuyo valor confirma una de dos posibilidades excluyentes. Por ejemplo, un conmutador puede estar en una de dos posiciones pero no ambas al mismo tiempo pero el hecho es que al estar en una posición representa un bit.

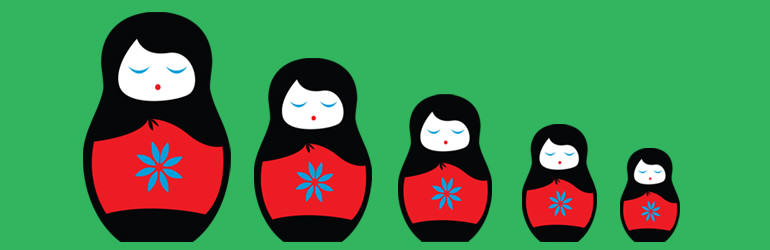
Pues asi es la parte estructurada de la computacion, Que nos permite encontrar diversas formas métodos, con los cual es podemos avanzar en diversas partes yasea para usarlas como herramientas que funcionen como grandes precursores en todas las operaciones básicas que realizamos a la hora de realizar diferentes operaciones en nuestros propios programas, siendo así la estructura de datos una manera de construir un mapa para no perderse a la hora de El manejo de esa intensa y extensa información, métodos que veremos a lo largo del desarrollo de este trabajo.

La búsqueda en árboles es un método de búsqueda simple, dinámico y eficiente considerado como uno de los fundamentales en Ciencia de la Computación. De toda la terminología sobre árboles,tan sólo recordar que la propiedad que define un árbol binario es que cada nodo tiene a lo más un hijo a la izquierda y uno a la derecha.Para construir los algoritmos consideraremos que cada nodo contiene un registro con un valor clave a través del cual efectuaremos las búsquedas.En las implementaciones que presentaremos sólo se considerará en cada nodo del árbol un valor del tipo *de elemento* aunque en un caso general ese tipo estará compuesto por dos:una clave indicando el campo por el cual se realiza la ordenación y una información asociada a dicha clave o visto de otra forma,una información que puede ser compuesta en la cual existe definido un orde

# 

# Marco Teórico

## Recursividad



Es una técnica utilizada en programación que nos permite que un bloque de instrucciones se ejecute un cierto número de veces (el que nosotros determinemos). A veces es algo complicado de entender, pero no os preocupéis. Cuando veamos los ejemplos estará clarísimo. En Java, como en otros muchos lenguajes, los métodos pueden llamarse a sí mismos. Gracias a esto, podemos utilizar a nuestro favor la recursividad en lugar de la iteración para resolver determinados tipos de problemas. **[1]:**

### Ejemplo sencillo

Vamos a ver un pequeño ejemplo que no hace absolutamente nada. Es un método cuyo único objetivo es llamarse a sí mismo: **[1]:**

void cuentaRegresiva () {

cuentaRegresiva();

}

 Si ejecutáis esto, os va a dar un error en la pila (mítico StackOverflow Error, Biblia de los programadores).

Como podemos ver, se ha llamado al método **cuentaRegresiva**porque vamos a mostrar por pantalla la cuenta atrás de un número que nosotros pasemos como parámetro a la función. Por ejemplo, para hacer la cuenta atrás de 10 sin recursividad, haríamos: **[1]:**

for( int i = 10; i >= 0; i--) {

System.out.println(i);

}

 Ahora, para hacerlo de manera recursiva, tendríamos que pasar como parámetro un número. Además, tras imprimir ese número, llamaremos a la misma función con el número actual restando uno: **[1]:**

void cuentaRegresiva(int numero) {

System.out.println(numero);

cuentaRegresiva(numero - 1);

}

Es lo que os he comentado arriba. Llamamos a la función con un 10. Imprimimos el 10 y llamamos a la función con un 9. Imprimimos el 9 y llamamos a la función con un 8. Así hasta el fin de los días. Digo hasta el fin de los días porque os va a saltar error si ejecutáis esto así directamente: **[1]:**

public class Recursividad {

static void cuentaRegresiva(int numero) {

System.out.println(numero);

cuentaRegresiva(numero - 1);

}

public static void main(String[] args) {

cuentaRegresiva(10);

}

}

 Problema: llamada infinita. Para ello, lo que tenemos que hacer es que cuando el número sea 0, deje de llamar a la función. Para eso, metemos una [estructura condicional](http://geekytheory.com/tutorial-3-java-estructuras-condicionales-y-excepciones/) de toda la vida. Gracias al condicional, dejará de ejecutarse a partir de 0: **[1]:**

void cuentaRegresiva(int numero) {

System.out.println(numero);

if(numero > 0) {

cuentaRegresiva(numero - 1);

}

}

# Arboles MINIMAX

Algoritmo de decisión para minimizar la pérdida máxima aplicada en juegos de adversarios

• Información completa (cada jugador conoce el estado del otro)

• Elección del mejor movimiento para cada jugador, suponiendo que el contrincante escogerá el peor

• El espacio de estados se representa mediante árboles alternados, donde:

**Nodo:** Representa una situación del juego.

**Sucesores de un nodo:** Situaciones del juego a las que se accede por movimientos legales aplicando sus reglas.

**Nivel:** Contiene todas las situaciones posibles para uno de los jugadores.

El algoritmo Minimax es un procedimiento recursivo y el corte de la recursión está dado por alguna de las siguientes condiciones:

* Gana algún jugador o Se han explorado N capas, siendo N el límite establecido
* Se ha agotado el tiempo de exploración
* Se ha llegado a una situación estática donde no hay grandes cambios de un nivel a otro.

## Representación de los juegos

• Posición inicial.

• Conjunto de operadores o reglas del juego (definen movimientos legales)

• Estado terminal

• Función de utilidad, ej. gana, pierde, empata

Pasos del Algoritmo Minimax

1. Generación del árbol de juego. Se generarán todos los nodos hasta llegar a un estado terminal.

2. Cálculo de los valores de la función de utilidad para cada nodo terminal.

3.Calcular el valor de los nodos superiores a partir del valor de los inferiores. Alternativamente se elegirán los valores mínimos y máximos representando los movimientos del jugador y del oponente, de ahí el nombre de Minimax.

4.Elegir la jugada valorando los valores que han llegado al nivel superior.

• El algoritmo explorará los nodos del árbol asignándoles un valor numérico mediante una función de utilidad, empezando por los nodos terminales y subiendo hacia la raíz. • Colocar 0 ó 1 en los nodos terminales dependiendo si gana MIN o MAX

• La función de utilidad definirá lo buena que es la posición para un jugador cuando la alcanza.

• Se requiere de una estrategia que garantice llegar a estados terminales ganadores independientemente de lo que haga el oponente.

• Un valor positivo indica la ventaja de un jugador y uno negativa la ventaja del otro.

• El jugador que espera valores positivos se conoce como maximizador

• El jugador que espera valores negativos se conoce como minimizador

• El maximizador busca movimientos que lo conduzcan al mayor número positivo

• El minimizador busca movimientos que lo conduzcan al menor número negativo

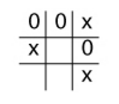
## IA MINIMAX GATO

Debido a la gran demanda de tecnología, la IA es una ciencia aplicada que consiste en incluir inteligencia a máquinas creadas por el hombre, estas tecnologías de IA también son aplicadas en juegos para darle una mejor interacción humano-computador.

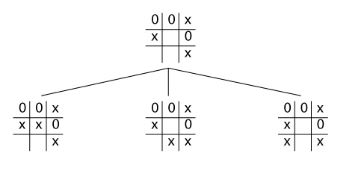
El algoritmo de minimax en simples palabras consiste en la elección del mejor movimiento para el computador, suponiendo que el contrincante escogerá uno que lo pueda perjudicar, para escoger la mejor opción este algoritmo realiza un árbol de búsqueda con todos los posibles movimientos, luego recorre todo el árbol de soluciones del juego a partir de un estado dado, es decir, según las casillas que ya han sido rellenadas. Por tanto, minimax se ejecutará cada vez que le toque mover a la IA.

En el algoritmo Minimax el espacio de búsqueda queda definido por:

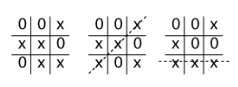
Estado inicial: Es una configuración inicial del juego, es decir, un estado en el que se encuentre el juego. Para nuestro ejemplo sería:



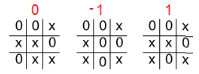
Operadores: Corresponden a las jugadas legales que se pueden hacer en el juego, en el caso del tres en raya no puedes marcar una casilla ya antes marcada.



Condición Terminal: Determina cuando el juego se acabó, en nuestro ejemplo el juego termina cuando un jugador marca tres casillas seguidas iguales, ya se horizontalmente, verticalmente o en diagonal, o se marcan todas las casillas (empate) .

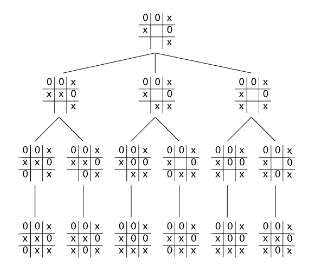


Función de Utilidad: Da un valor numérico a una configuración final de un juego. En un juego en donde se puede ganar, perder o empatar, los valores pueden ser 1, 0, o -1.

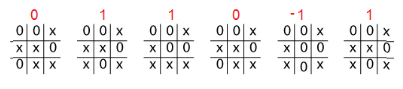


Implementación Minimax: Los pasos que sigue minimax pueden variar, pero lo importante es tener una idea clara de cómo es su funcionamiento, los pasos a seguir son:

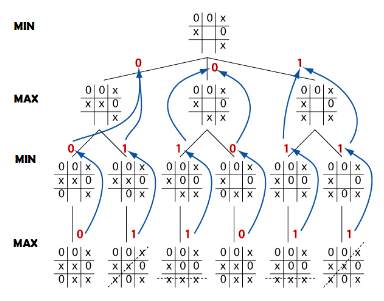
El algoritmo primero generar un árbol de soluciones completo a partir de un nodo dado. veamos el siguiente ejemplo:



Para cada nodo final, buscamos la función de utilidad de estos. En nuestro ejemplo usaremos un 0 para las partidas que terminen en empate, un 1 para las que gane la IA y un -1 para las que gane el jugador humano.



Y lo que hará el algoritmo Minimax cuando vaya regresando hacia atrás, será comunicarle a la llamada recursiva superior cuál es el mejor nodo hoja alcanzado hasta el momento. Cada llamada recursiva tiene que saber a quién le toca jugar, para analizar si el movimiento realizado pertenece a la IA o al otro jugador, ya que cuando sea el turno de la IA nos interesa MAXIMIZAR el resultado, y cuando sea el turno del rival MINIMIZAR su resultado.



Al final el algoritmo nos devolverá la jugada que debe realizar la máquina para maximizar sus posibilidades y bloquear las posibilidades del rival.

# Ejemplo de ARBOL MINIMAX programado por OGMR

#include <stdio.h>

#include <stdlib.h>

#include <graphics.h>

#define N 9

enum { O,X,EMPTY };

int HUMAN = 0;

int COMP = 1;

int contTableros=0;

int Tablero1[N];

int Tablero2[N];

int Tablero3[N];

int Tablero4[N];

int Tablero5[N];

int Tablero6[N];

int Tablero7[N];

int Tablero8[N];

int Tablero9[N];

// empezamos un tablero nuevo TABLEROS AUXILIARES INHABILITADOS

void InvocarTablero(int Tablero[]) {

for(int i=0; i<N; i++){

Tablero[i] = EMPTY;

Tablero1[i] = EMPTY;

Tablero2[i] = EMPTY;

Tablero3[i] = EMPTY;

Tablero4[i] = EMPTY;

Tablero5[i] = EMPTY;

Tablero6[i] = EMPTY;

Tablero7[i] = EMPTY;

Tablero8[i] = EMPTY;

Tablero9[i] = EMPTY;

}

}

// Imprimimos el Tablero en consola y finalmente graficamente

void ImpresionTablero(const int Tablero[]) {

int x=810, y=85;

char Ficha[] = { 'O','X','-' };

char m;

setlinestyle(3, 3, 3);

setcolor(2);

settextstyle(BOLD\_FONT, HORIZ\_DIR,5);

outtextxy(750,10,"Gato MINIMAX");

setcolor(BROWN);

settextstyle(BOLD\_FONT, HORIZ\_DIR, 3);

setcolor(GREEN);

line(0,80,1910,80);

setcolor(BLUE);

//Base de PRIMER NIVEL

line(905,80,905,380);

line(1005,80,1005,380);

line(805,180,1105,180);

line(805,280,1105,280);

printf("\n Tablero\n\n");

for(int i=0; i<N; i++, x=x+100) {

if(i != 0 && i%3 == 0){

printf("\n\n");

y=y+100;

x=810;

}

printf(" %c ",Ficha[Tablero[i]]);

if(Ficha[Tablero[i]]=='O'){

settextstyle(BOLD\_FONT, HORIZ\_DIR,7);

setcolor(GREEN);

outtextxy(x,y,"O");

x=x+15;

}

else if (Ficha[Tablero[i]]=='X'){

settextstyle(BOLD\_FONT, HORIZ\_DIR,8);

setcolor(RED);

outtextxy(x,y,"X");

x=x+15;

}

if(Ficha[Tablero[i]]=='-'){

x=x+15;

}

}

delay(2000);

cleardevice();

contTableros++;

if (contTableros==2){

for(int i=0, j=0; i<N; i++, j++) {

Tablero1[j]=Tablero[i];

}

}

else if (contTableros==3){

for(int i=0, j=0; i<N; i++, j++) {

Tablero2[j]=Tablero[i];

}

}

else if (contTableros==4){

for(int i=0, j=0; i<N; i++, j++) {

Tablero3[j]=Tablero[i];

}

}

else if (contTableros==5){

for(int i=0, j=0; i<N; i++, j++) {

Tablero4[j]=Tablero[i];

}

}

else if (contTableros==6){

for(int i=0, j=0; i<N; i++, j++) {

Tablero5[j]=Tablero[i];

}

}

else if (contTableros==7){

for(int i=0, j=0; i<N; i++, j++) {

Tablero6[j]=Tablero[i];

}

}

else if (contTableros==8){

for(int i=0, j=0; i<N; i++, j++) {

Tablero7[j]=Tablero[i];

}

}

else if (contTableros==9){

for(int i=0, j=0; i<N; i++, j++) {

Tablero8[j]=Tablero[i];

}

}

else if (contTableros==10){

for(int i=0, j=0; i<N; i++, j++) {

Tablero9[j]=Tablero[i];

}

}

printf("\n\n");

}

// Se revisa si se lleno el tablero

// retorna 1(true) si esta lleno

// retorna 0(false) si falta al menos una celda vacia

int VerificacionTermino(const int Tablero[]) {

for(int i=0; i<N; i++)

if(Tablero[i] == EMPTY)

return 0;

return 1;

}

// Hace el movimiento en la celda indicada, con el lado indicado

void IntroducirMovimiento(int Tablero[], const int celda, const int lado) {

Tablero[celda] = lado;

}

// Revisa si se ha hecho 3 en raya

// returns 1 si es correcto

// returns 0 si no

int TresLinea(const int Tablero[], const int lado) {

// horizontal

if(Tablero[0] == lado && Tablero[1] == lado && Tablero[2] == lado)

return 1;

if(Tablero[3] == lado && Tablero[4] == lado && Tablero[5] == lado)

return 1;

if(Tablero[6] == lado && Tablero[7] == lado && Tablero[8] == lado)

return 1;

// vertical

if(Tablero[0] == lado && Tablero[3] == lado && Tablero[6] == lado)

return 1;

if(Tablero[1] == lado && Tablero[4] == lado && Tablero[7] == lado)

return 1;

if(Tablero[2] == lado && Tablero[5] == lado && Tablero[8] == lado)

return 1;

// diagonal

if(Tablero[0] == lado && Tablero[4] == lado && Tablero[8] == lado)

return 1;

if(Tablero[2] == lado && Tablero[4] == lado && Tablero[6] == lado)

return 1;

return 0;

}

// devuelve la puntuación del lado

// +10 si gana un bando particular

// -10 si un bando particular pierde

// 0 si se empata o aún no se ha completado el juego

int TenemosGanador(const int Tablero[], const int lado) {

if(TresLinea(Tablero,lado))

return 10;

if(TresLinea(Tablero,!lado))

return -10;

return 0;

}

// devuelve qué movimientos da la puntuación MAXIMA

int max(int \*ListaPuntaje,int ContadorMov,int \*ListaMov,int \*MejorMov) {

if(!ContadorMov)

return 0;

int max = -20;

for(int i=0;i<ContadorMov;i++) {

if(ListaPuntaje[i] > max) {

max = ListaPuntaje[i];

\*MejorMov = ListaMov[i];

}

}

return max;

}

// devuelve qué movimientos da la puntuación mínima

int min(int \*ListaPuntaje,int ContadorMov,int \*ListaMov,int \*MejorMov) {

if(!ContadorMov)

return 0;

int min = +20;

for(int i=0;i<ContadorMov;i++) {

if(ListaPuntaje[i] < min) {

min = ListaPuntaje[i];

\*MejorMov = ListaMov[i];

}

}

return min;

}

int CorazonIA(int Tablero[], int lado, int \*Profundidad) {

int ListaMov[N];

int ContadorMov = 0;

int MejorMov;

int PuntajeMov[N];

int MejorPuntaje;

MejorPuntaje = TenemosGanador(Tablero,COMP);

if(MejorPuntaje)

return MejorPuntaje - \*Profundidad;

// Llena ListaMov[]

for(int i=0;i<N;i++) {

if(Tablero[i] == EMPTY)

ListaMov[ContadorMov++] = i;

}

// loop para todos los movimientos

int Movimiento;

for(int i=0;i<ContadorMov;i++) {

Movimiento = ListaMov[i];

IntroducirMovimiento(Tablero,Movimiento,lado);

(\*Profundidad)++;

PuntajeMov[i] = CorazonIA(Tablero,!lado,Profundidad);

(\*Profundidad)--;

IntroducirMovimiento(Tablero,Movimiento,EMPTY);

}

if(lado == COMP) {

//Optiene Max de su tiro

MejorPuntaje = max(PuntajeMov,ContadorMov,ListaMov,&MejorMov);

}

if(lado == HUMAN) {

// Optiene el Min del Humano

MejorPuntaje = min(PuntajeMov,ContadorMov,ListaMov,&MejorMov);

}

if(\*Profundidad != 0)

return MejorPuntaje;

else

return MejorMov;

}

int MovimientoIA(int \*Tablero,int lado) {

int Profundidad = 0;

int MovimientoMejorado = CorazonIA(Tablero,lado,&Profundidad);

printf("Para ser invensible se tiene que buscar lo mejor.... Mejor elección en: %d\n",MovimientoMejorado+1);

return MovimientoMejorado;

}

int MovimientoMortal(const int \*Tablero) {

printf("\nIntroducir el movimiento!!\n\n");

int movimiento;

while(1) {

scanf("%d",&movimiento);

if(Tablero[movimiento-1] == EMPTY && movimiento >=0 && movimiento <=9)

break;

else

printf("No es posible ahi, lo mejor sera que te lo pienses mejor: ");

}

return movimiento-1;

}

void InicioGato() {

printf("\nEscoje X o O. O mueve primero!!\n\n");

setlinestyle(3, 3, 3);

setcolor(2);

settextstyle(BOLD\_FONT, HORIZ\_DIR,5);

outtextxy(600,10,"BIENVENIDO AL JUEGO DEL GATO");

setcolor(BLUE);

settextstyle(BOLD\_FONT, HORIZ\_DIR, 5);

outtextxy(50,50,"INSTRUCCIONES:");

setcolor(BROWN);

settextstyle(BOLD\_FONT, HORIZ\_DIR, 3);

outtextxy(50,100,"Este es el juego clasico del gato, jugaras con la IA (MINIMAX) perfecta, no tendrás margen de error");

setcolor(BLUE);

settextstyle(BOLD\_FONT, HORIZ\_DIR, 4);

outtextxy(50,150,"Escoje X o O. O mueve primero!!");

delay(4000);

cleardevice();

while(1) {

char eleccion = getchar();

getchar();

if(eleccion == 'O') {

HUMAN = O;

COMP = X;

break;

}

if(eleccion == 'X') {

HUMAN = X;

COMP = O;

break;

}

else {

printf("DEBE DE ELEGIR UNA OPCION CORRECTA\n");

}

}

int gameOver = 0;

int lado = O;

int movimiento;

int Tablero[N];

InvocarTablero(Tablero);

ImpresionTablero(Tablero);

while(!gameOver) {

if(lado == HUMAN) {

//Movimiento del jugador

movimiento = MovimientoMortal(Tablero);

IntroducirMovimiento(Tablero,movimiento,lado);

ImpresionTablero(Tablero);

}

else {

//Movimiento del Ordenador

movimiento = MovimientoIA(Tablero,lado);

IntroducirMovimiento(Tablero,movimiento,lado);

ImpresionTablero(Tablero);

}

//Ver si gano alguien

if(TresLinea(Tablero,lado)) {

printf("Game Over\n");

gameOver = 1;

if(lado == COMP)

printf("Se te advirtio, la IA es invensible!!!\n");

else

printf("Este mensaje nunca se verá, pues el que un humano gane, es imposible\n");

}

//Ver si en su defecto hubo empate

if(VerificacionTermino(Tablero)) {

printf("Game Over\n");

gameOver = 1;

printf("Supongo que aun asi no ganaste... Toma la tregua.\n");

}

lado = !lado;

}

}

void Arbol();

int main() {

initwindow(1910,400);

InicioGato();

closegraph();

Arbol();

return 0;

}

void Arbol(){

char ptb[5];

int contProf=1;

int cont=1;

printf("\nAhora que se concluyo, veamos el arbol de decision que se ha seguido en la partida\n\n");

initwindow(1910,1000);

setlinestyle(3, 3, 3);

setcolor(2);

settextstyle(BOLD\_FONT, HORIZ\_DIR, 5);

outtextxy(750,10,"Arboles MINIMAX");

settextstyle(BOLD\_FONT, HORIZ\_DIR, 3);

setcolor(RED);

outtextxy(10,50,"Se Hicieron movimientos:");

setcolor(BROWN);

settextstyle(BOLD\_FONT, HORIZ\_DIR, 5);

outtextxy(400,180,"BASE INICIAL");

settextstyle(BOLD\_FONT, HORIZ\_DIR, 3);

setcolor(GREEN);

line(0,80,1910,80);

setcolor(BLUE);

//Base de PRIMER NIVEL

line(905,80,905,380);

line(1005,80,1005,380);

line(805,180,1105,180);

line(805,280,1105,280);

//ENLACES DE PRIMER A SEGUNDO NIVEL

setcolor(RED);

line(955,380,167,480);

line(955,380,362,480);

line(955,380,557,480);

line(955,380,752,480);

line(955,380,947,480);

line(955,380,1142,480);

line(955,380,1357,480);

line(955,380,1532,480);

line(955,380,1727,480);

setcolor(BLUE);

//BASE DE SEGUNDO NIVEL

//Opcion No.1

line(77,540,257,540);

line(77,600,257,600);

line(137,480,137,660);

line(197,480,197,660);

setcolor(BROWN);

circle(107,510,25);

setcolor(BLUE);

//Opcion No.2

line(272,540,452,540);

line(272,600,452,600);

line(332,480,332,660);

line(392,480,392,660);

setcolor(BROWN);

circle(362,510,25);

setcolor(BLUE);

//Opcion No.3

line(467,540,647,540);

line(467,600,647,600);

line(527,480,527,660);

line(587,480,587,660);

setcolor(BROWN);

circle(617,510,25);

setcolor(BLUE);

//Opcion No.4

line(662,540,842,540);

line(662,600,842,600);

line(722,480,722,660);

line(782,480,782,660);

setcolor(BROWN);

circle(692,570,25);

setcolor(BLUE);

//Opcion No.5

line(857,540,1037,540);

line(857,600,1037,600);

line(917,480,917,660);

line(977,480,977,660);

setcolor(BROWN);

circle(947,570,25);

setcolor(BLUE);

//Opcion No.6

line(1052,540,1232,540);

line(1052,600,1232,600);

line(1112,480,1112,660);

line(1172,480,1172,660);

setcolor(BROWN);

circle(1202,570,25);

setcolor(BLUE);

//Opcion No.7

line(1247,540,1427,540);

line(1247,600,1427,600);

line(1307,480,1307,660);

line(1367,480,1367,660);

setcolor(BROWN);

circle(1277,630,25);

setcolor(BLUE);

//Opcion No.8

line(1442,540,1622,540);

line(1442,600,1622,600);

line(1502,480,1502,660);

line(1562,480,1562,660);

setcolor(BROWN);

circle(1532,630,25);

setcolor(BLUE);

//Opcion No.9

line(1637,540,1817,540);

line(1637,600,1817,600);

line(1697,480,1697,660);

line(1757,480,1757,660);

setcolor(BROWN);

circle(1787,630,25);

setcolor(BLUE);

setcolor(BROWN);

outtextxy(97,670,"Opcion No.1");

outtextxy(292,670,"Opcion No.2");

outtextxy(487,670,"Opcion No.3");

outtextxy(682,670,"Opcion No.4");

outtextxy(877,670,"Opcion No.5");

outtextxy(1072,670,"Opcion No.6");

outtextxy(1267,670,"Opcion No.7");

outtextxy(1462,670,"Opcion No.8");

outtextxy(1657,670,"Opcion No.9");

delay(3000);

cleardevice();

//PRIMER TIRADA

//Opciones 1 AL 8

cont=1;

for(int i=0;Tablero1[i]==2;i++){

cont++;

}

setlinestyle(3, 3, 3);

setcolor(2);

settextstyle(BOLD\_FONT, HORIZ\_DIR, 5);

outtextxy(750,10,"Arboles MINIMAX");

settextstyle(BOLD\_FONT, HORIZ\_DIR, 3);

setcolor(RED);

outtextxy(10,50,"Se Hicieron movimientos:");

setcolor(BROWN);

settextstyle(BOLD\_FONT, HORIZ\_DIR, 5);

outtextxy(400,180,"Tirada No.1");

settextstyle(BOLD\_FONT, HORIZ\_DIR, 3);

setcolor(GREEN);

line(0,80,1910,80);

setcolor(BLUE);

if(cont==1){////////////////////////////////////////////////////////////////

//Base de PRIMER NIVEL Opcion No.1

line(905,80,905,380);

line(1005,80,1005,380);

line(805,180,1105,180);

line(805,280,1105,280);

setcolor(BROWN);

circle(855,130,45);

setcolor(BLUE);

//ENLACES DE PRIMER A SEGUNDO NIVEL

setcolor(RED);

line(955,380,167,480);

line(955,380,362,480);

line(955,380,557,480);

line(955,380,752,480);

line(955,380,947,480);

line(955,380,1142,480);

line(955,380,1357,480);

line(955,380,1532,480);

setcolor(BLUE);

//BASE DE SEGUNDO NIVEL

//Opcion No.1

line(77,540,257,540);

line(77,600,257,600);

line(137,480,137,660);

line(197,480,197,660);

setcolor(BROWN);

circle(107,510,25);

line(137,480,197,540);

line(137,540,197,480);

setcolor(BLUE);

//Opcion No.2

line(272,540,452,540);

line(272,600,452,600);

line(332,480,332,660);

line(392,480,392,660);

setcolor(BROWN);

circle(302,510,25);

line(392,480,452,540);

line(392,540,452,480);

setcolor(BLUE);

//Opcion No.3

line(467,540,647,540);

line(467,600,647,600);

line(527,480,527,660);

line(587,480,587,660);

setcolor(BROWN);

circle(497,510,25);

line(467,540,527,600);

line(467,600,527,540);

setcolor(BLUE);

//Opcion No.4

line(662,540,842,540);

line(662,600,842,600);

line(722,480,722,660);

line(782,480,782,660);

setcolor(BROWN);

circle(692,510,25);

line(722,540,782,600);

line(722,600,782,540);

setcolor(BLUE);

//Opcion No.5

line(857,540,1037,540);

line(857,600,1037,600);

line(917,480,917,660);

line(977,480,977,660);

setcolor(BROWN);

circle(887,510,25);

line(977,540,1037,600);

line(977,600,1037,540);

setcolor(BLUE);

//Opcion No.6

line(1052,540,1232,540);

line(1052,600,1232,600);

line(1112,480,1112,660);

line(1172,480,1172,660);

setcolor(BROWN);

circle(1082,510,25);

line(1052,600,1112,660);

line(1052,660,1112,600);

setcolor(BLUE);

//Opcion No.7

line(1247,540,1427,540);

line(1247,600,1427,600);

line(1307,480,1307,660);

line(1367,480,1367,660);

setcolor(BROWN);

circle(1277,510,25);

line(1327,600,1367,660);

line(1327,660,1367,600);

setcolor(BLUE);

//Opcion No.8

line(1442,540,1622,540);

line(1442,600,1622,600);

line(1502,480,1502,660);

line(1562,480,1562,660);

setcolor(BROWN);

circle(1472,510,25);

line(1562,600,1622,660);

line(1562,660,1622,600);

setcolor(BLUE);

setcolor(BROWN);

outtextxy(97,670,"Opcion No.1");

outtextxy(292,670,"Opcion No.2");

outtextxy(487,670,"Opcion No.3");

outtextxy(682,670,"Opcion No.4");

outtextxy(877,670,"Opcion No.5");

outtextxy(1072,670,"Opcion No.6");

outtextxy(1267,670,"Opcion No.7");

outtextxy(1462,670,"Opcion No.8");

}

if(cont==2){ //////////////////////////////////////////////////////////////////

//Base de PRIMER NIVEL Opcion No.2

line(905,80,905,380);

line(1005,80,1005,380);

line(805,180,1105,180);

line(805,280,1105,280);

setcolor(BROWN);

circle(955,130,45);

setcolor(BLUE);

//ENLACES DE PRIMER A SEGUNDO NIVEL

setcolor(RED);

line(955,380,167,480);

line(955,380,362,480);

line(955,380,557,480);

line(955,380,752,480);

line(955,380,947,480);

line(955,380,1142,480);

line(955,380,1357,480);

line(955,380,1532,480);

setcolor(BLUE);

//BASE DE SEGUNDO NIVEL

//Opcion No.1

line(77,540,257,540);

line(77,600,257,600);

line(137,480,137,660);

line(197,480,197,660);

setcolor(BROWN);

circle(107,510,25);

setcolor(BLUE);

//Opcion No.2

line(272,540,452,540);

line(272,600,452,600);

line(332,480,332,660);

line(392,480,392,660);

setcolor(BROWN);

circle(362,510,25);

setcolor(BLUE);

//Opcion No.3

line(467,540,647,540);

line(467,600,647,600);

line(527,480,527,660);

line(587,480,587,660);

setcolor(BROWN);

circle(617,510,25);

setcolor(BLUE);

//Opcion No.4

line(662,540,842,540);

line(662,600,842,600);

line(722,480,722,660);

line(782,480,782,660);

setcolor(BROWN);

circle(692,570,25);

setcolor(BLUE);

//Opcion No.5

line(857,540,1037,540);

line(857,600,1037,600);

line(917,480,917,660);

line(977,480,977,660);

setcolor(BROWN);

circle(947,570,25);

setcolor(BLUE);

//Opcion No.6

line(1052,540,1232,540);

line(1052,600,1232,600);

line(1112,480,1112,660);

line(1172,480,1172,660);

setcolor(BROWN);

circle(1202,570,25);

setcolor(BLUE);

//Opcion No.7

line(1247,540,1427,540);

line(1247,600,1427,600);

line(1307,480,1307,660);

line(1367,480,1367,660);

setcolor(BROWN);

circle(1277,630,25);

setcolor(BLUE);

//Opcion No.8

line(1442,540,1622,540);

line(1442,600,1622,600);

line(1502,480,1502,660);

line(1562,480,1562,660);

setcolor(BROWN);

circle(1532,630,25);

setcolor(BLUE);

setcolor(BROWN);

outtextxy(97,670,"Opcion No.1");

outtextxy(292,670,"Opcion No.2");

outtextxy(487,670,"Opcion No.3");

outtextxy(682,670,"Opcion No.4");

outtextxy(877,670,"Opcion No.5");

outtextxy(1072,670,"Opcion No.6");

outtextxy(1267,670,"Opcion No.7");

outtextxy(1462,670,"Opcion No.8");

}

if(cont==3){//////////////////////////////////////////////////////////////

//Base de PRIMER NIVEL

line(905,80,905,380);

line(1005,80,1005,380);

line(805,180,1105,180);

line(805,280,1105,280);

setcolor(BROWN);

circle(1055,130,45);

setcolor(BLUE);

//ENLACES DE PRIMER A SEGUNDO NIVEL

setcolor(RED);

line(955,380,167,480);

line(955,380,362,480);

line(955,380,557,480);

line(955,380,752,480);

line(955,380,947,480);

line(955,380,1142,480);

line(955,380,1357,480);

line(955,380,1532,480);

setcolor(BLUE);

//BASE DE SEGUNDO NIVEL

//Opcion No.1

line(77,540,257,540);

line(77,600,257,600);

line(137,480,137,660);

line(197,480,197,660);

setcolor(BROWN);

circle(107,510,25);

setcolor(BLUE);

//Opcion No.2

line(272,540,452,540);

line(272,600,452,600);

line(332,480,332,660);

line(392,480,392,660);

setcolor(BROWN);

circle(362,510,25);

setcolor(BLUE);

//Opcion No.3

line(467,540,647,540);

line(467,600,647,600);

line(527,480,527,660);

line(587,480,587,660);

setcolor(BROWN);

circle(617,510,25);

setcolor(BLUE);

//Opcion No.4

line(662,540,842,540);

line(662,600,842,600);

line(722,480,722,660);

line(782,480,782,660);

setcolor(BROWN);

circle(692,570,25);

setcolor(BLUE);

//Opcion No.5

line(857,540,1037,540);

line(857,600,1037,600);

line(917,480,917,660);

line(977,480,977,660);

setcolor(BROWN);

circle(947,570,25);

setcolor(BLUE);

//Opcion No.6

line(1052,540,1232,540);

line(1052,600,1232,600);

line(1112,480,1112,660);

line(1172,480,1172,660);

setcolor(BROWN);

circle(1202,570,25);

setcolor(BLUE);

//Opcion No.7

line(1247,540,1427,540);

line(1247,600,1427,600);

line(1307,480,1307,660);

line(1367,480,1367,660);

setcolor(BROWN);

circle(1277,630,25);

setcolor(BLUE);

//Opcion No.8

line(1442,540,1622,540);

line(1442,600,1622,600);

line(1502,480,1502,660);

line(1562,480,1562,660);

setcolor(BROWN);

circle(1532,630,25);

setcolor(BLUE);

setcolor(BROWN);

outtextxy(97,670,"Opcion No.1");

outtextxy(292,670,"Opcion No.2");

outtextxy(487,670,"Opcion No.3");

outtextxy(682,670,"Opcion No.4");

outtextxy(877,670,"Opcion No.5");

outtextxy(1072,670,"Opcion No.6");

outtextxy(1267,670,"Opcion No.7");

outtextxy(1462,670,"Opcion No.8");

}

if(cont==4){//////////////////////////////////////////////////////////////

//Base de PRIMER NIVEL

line(905,80,905,380);

line(1005,80,1005,380);

line(805,180,1105,180);

line(805,280,1105,280);

setcolor(BROWN);

circle(855,230,45);

setcolor(BLUE);

//ENLACES DE PRIMER A SEGUNDO NIVEL

setcolor(RED);

line(955,380,167,480);

line(955,380,362,480);

line(955,380,557,480);

line(955,380,752,480);

line(955,380,947,480);

line(955,380,1142,480);

line(955,380,1357,480);

line(955,380,1532,480);

setcolor(BLUE);

//BASE DE SEGUNDO NIVEL

//Opcion No.1

line(77,540,257,540);

line(77,600,257,600);

line(137,480,137,660);

line(197,480,197,660);

setcolor(BROWN);

circle(107,510,25);

setcolor(BLUE);

//Opcion No.2

line(272,540,452,540);

line(272,600,452,600);

line(332,480,332,660);

line(392,480,392,660);

setcolor(BROWN);

circle(362,510,25);

setcolor(BLUE);

//Opcion No.3

line(467,540,647,540);

line(467,600,647,600);

line(527,480,527,660);

line(587,480,587,660);

setcolor(BROWN);

circle(617,510,25);

setcolor(BLUE);

//Opcion No.4

line(662,540,842,540);

line(662,600,842,600);

line(722,480,722,660);

line(782,480,782,660);

setcolor(BROWN);

circle(692,570,25);

setcolor(BLUE);

//Opcion No.5

line(857,540,1037,540);

line(857,600,1037,600);

line(917,480,917,660);

line(977,480,977,660);

setcolor(BROWN);

circle(947,570,25);

setcolor(BLUE);

//Opcion No.6

line(1052,540,1232,540);

line(1052,600,1232,600);

line(1112,480,1112,660);

line(1172,480,1172,660);

setcolor(BROWN);

circle(1202,570,25);

setcolor(BLUE);

//Opcion No.7

line(1247,540,1427,540);

line(1247,600,1427,600);

line(1307,480,1307,660);

line(1367,480,1367,660);

setcolor(BROWN);

circle(1277,630,25);

setcolor(BLUE);

//Opcion No.8

line(1442,540,1622,540);

line(1442,600,1622,600);

line(1502,480,1502,660);

line(1562,480,1562,660);

setcolor(BROWN);

circle(1532,630,25);

setcolor(BLUE);

setcolor(BROWN);

outtextxy(97,670,"Opcion No.1");

outtextxy(292,670,"Opcion No.2");

outtextxy(487,670,"Opcion No.3");

outtextxy(682,670,"Opcion No.4");

outtextxy(877,670,"Opcion No.5");

outtextxy(1072,670,"Opcion No.6");

outtextxy(1267,670,"Opcion No.7");

outtextxy(1462,670,"Opcion No.8");

}

if(cont==5){//////////////////////////////////////////////////////////

//Base de PRIMER NIVEL

line(905,80,905,380);

line(1005,80,1005,380);

line(805,180,1105,180);

line(805,280,1105,280);

setcolor(BROWN);

circle(955,230,45);

setcolor(BLUE);

//ENLACES DE PRIMER A SEGUNDO NIVEL

setcolor(RED);

line(955,380,167,480);

line(955,380,362,480);

line(955,380,557,480);

line(955,380,752,480);

line(955,380,947,480);

line(955,380,1142,480);

line(955,380,1357,480);

line(955,380,1532,480);

setcolor(BLUE);

//BASE DE SEGUNDO NIVEL

//Opcion No.1

line(77,540,257,540);

line(77,600,257,600);

line(137,480,137,660);

line(197,480,197,660);

setcolor(BROWN);

circle(107,510,25);

setcolor(BLUE);

//Opcion No.2

line(272,540,452,540);

line(272,600,452,600);

line(332,480,332,660);

line(392,480,392,660);

setcolor(BROWN);

circle(362,510,25);

setcolor(BLUE);

//Opcion No.3

line(467,540,647,540);

line(467,600,647,600);

line(527,480,527,660);

line(587,480,587,660);

setcolor(BROWN);

circle(617,510,25);

setcolor(BLUE);

//Opcion No.4

line(662,540,842,540);

line(662,600,842,600);

line(722,480,722,660);

line(782,480,782,660);

setcolor(BROWN);

circle(692,570,25);

setcolor(BLUE);

//Opcion No.5

line(857,540,1037,540);

line(857,600,1037,600);

line(917,480,917,660);

line(977,480,977,660);

setcolor(BROWN);

circle(947,570,25);

setcolor(BLUE);

//Opcion No.6

line(1052,540,1232,540);

line(1052,600,1232,600);

line(1112,480,1112,660);

line(1172,480,1172,660);

setcolor(BROWN);

circle(1202,570,25);

setcolor(BLUE);

//Opcion No.7

line(1247,540,1427,540);

line(1247,600,1427,600);

line(1307,480,1307,660);

line(1367,480,1367,660);

setcolor(BROWN);

circle(1277,630,25);

setcolor(BLUE);

//Opcion No.8

line(1442,540,1622,540);

line(1442,600,1622,600);

line(1502,480,1502,660);

line(1562,480,1562,660);

setcolor(BROWN);

circle(1532,630,25);

setcolor(BLUE);

setcolor(BROWN);

outtextxy(97,670,"Opcion No.1");

outtextxy(292,670,"Opcion No.2");

outtextxy(487,670,"Opcion No.3");

outtextxy(682,670,"Opcion No.4");

outtextxy(877,670,"Opcion No.5");

outtextxy(1072,670,"Opcion No.6");

outtextxy(1267,670,"Opcion No.7");

outtextxy(1462,670,"Opcion No.8");

}

if(cont==6){////////////////////////////////////////////////////////

//Base de PRIMER NIVEL

line(905,80,905,380);

line(1005,80,1005,380);

line(805,180,1105,180);

line(805,280,1105,280);

setcolor(BROWN);

circle(1055,230,45);

setcolor(BLUE);

//ENLACES DE PRIMER A SEGUNDO NIVEL

setcolor(RED);

line(955,380,167,480);

line(955,380,362,480);

line(955,380,557,480);

line(955,380,752,480);

line(955,380,947,480);

line(955,380,1142,480);

line(955,380,1357,480);

line(955,380,1532,480);

setcolor(BLUE);

//BASE DE SEGUNDO NIVEL

//Opcion No.1

line(77,540,257,540);

line(77,600,257,600);

line(137,480,137,660);

line(197,480,197,660);

setcolor(BROWN);

circle(107,510,25);

setcolor(BLUE);

//Opcion No.2

line(272,540,452,540);

line(272,600,452,600);

line(332,480,332,660);

line(392,480,392,660);

setcolor(BROWN);

circle(362,510,25);

setcolor(BLUE);

//Opcion No.3

line(467,540,647,540);

line(467,600,647,600);

line(527,480,527,660);

line(587,480,587,660);

setcolor(BROWN);

circle(617,510,25);

setcolor(BLUE);

//Opcion No.4

line(662,540,842,540);

line(662,600,842,600);

line(722,480,722,660);

line(782,480,782,660);

setcolor(BROWN);

circle(692,570,25);

setcolor(BLUE);

//Opcion No.5

line(857,540,1037,540);

line(857,600,1037,600);

line(917,480,917,660);

line(977,480,977,660);

setcolor(BROWN);

circle(947,570,25);

setcolor(BLUE);

//Opcion No.6

line(1052,540,1232,540);

line(1052,600,1232,600);

line(1112,480,1112,660);

line(1172,480,1172,660);

setcolor(BROWN);

circle(1202,570,25);

setcolor(BLUE);

//Opcion No.7

line(1247,540,1427,540);

line(1247,600,1427,600);

line(1307,480,1307,660);

line(1367,480,1367,660);

setcolor(BROWN);

circle(1277,630,25);

setcolor(BLUE);

//Opcion No.8

line(1442,540,1622,540);

line(1442,600,1622,600);

line(1502,480,1502,660);

line(1562,480,1562,660);

setcolor(BROWN);

circle(1532,630,25);

setcolor(BLUE);

setcolor(BROWN);

outtextxy(97,670,"Opcion No.1");

outtextxy(292,670,"Opcion No.2");

outtextxy(487,670,"Opcion No.3");

outtextxy(682,670,"Opcion No.4");

outtextxy(877,670,"Opcion No.5");

outtextxy(1072,670,"Opcion No.6");

outtextxy(1267,670,"Opcion No.7");

outtextxy(1462,670,"Opcion No.8");

}

if(cont==7){/////////////////////////////////////////////////////////

//Base de PRIMER NIVEL

line(905,80,905,380);

line(1005,80,1005,380);

line(805,180,1105,180);

line(805,280,1105,280);

setcolor(BROWN);

circle(855,330,45);

setcolor(BLUE);

//ENLACES DE PRIMER A SEGUNDO NIVEL

setcolor(RED);

line(955,380,167,480);

line(955,380,362,480);

line(955,380,557,480);

line(955,380,752,480);

line(955,380,947,480);

line(955,380,1142,480);

line(955,380,1357,480);

line(955,380,1532,480);

setcolor(BLUE);

//BASE DE SEGUNDO NIVEL

//Opcion No.1

line(77,540,257,540);

line(77,600,257,600);

line(137,480,137,660);

line(197,480,197,660);

setcolor(BROWN);

circle(107,510,25);

setcolor(BLUE);

//Opcion No.2

line(272,540,452,540);

line(272,600,452,600);

line(332,480,332,660);

line(392,480,392,660);

setcolor(BROWN);

circle(362,510,25);

setcolor(BLUE);

//Opcion No.3

line(467,540,647,540);

line(467,600,647,600);

line(527,480,527,660);

line(587,480,587,660);

setcolor(BROWN);

circle(617,510,25);

setcolor(BLUE);

//Opcion No.4

line(662,540,842,540);

line(662,600,842,600);

line(722,480,722,660);

line(782,480,782,660);

setcolor(BROWN);

circle(692,570,25);

setcolor(BLUE);

//Opcion No.5

line(857,540,1037,540);

line(857,600,1037,600);

line(917,480,917,660);

line(977,480,977,660);

setcolor(BROWN);

circle(947,570,25);

setcolor(BLUE);

//Opcion No.6

line(1052,540,1232,540);

line(1052,600,1232,600);

line(1112,480,1112,660);

line(1172,480,1172,660);

setcolor(BROWN);

circle(1202,570,25);

setcolor(BLUE);

//Opcion No.7

line(1247,540,1427,540);

line(1247,600,1427,600);

line(1307,480,1307,660);

line(1367,480,1367,660);

setcolor(BROWN);

circle(1277,630,25);

setcolor(BLUE);

//Opcion No.8

line(1442,540,1622,540);

line(1442,600,1622,600);

line(1502,480,1502,660);

line(1562,480,1562,660);

setcolor(BROWN);

circle(1532,630,25);

setcolor(BLUE);

setcolor(BROWN);

outtextxy(97,670,"Opcion No.1");

outtextxy(292,670,"Opcion No.2");

outtextxy(487,670,"Opcion No.3");

outtextxy(682,670,"Opcion No.4");

outtextxy(877,670,"Opcion No.5");

outtextxy(1072,670,"Opcion No.6");

outtextxy(1267,670,"Opcion No.7");

outtextxy(1462,670,"Opcion No.8");

}

if(cont==8){//////////////////////////////////////////////////////////////

//Base de PRIMER NIVEL

line(905,80,905,380);

line(1005,80,1005,380);

line(805,180,1105,180);

line(805,280,1105,280);

setcolor(BROWN);

circle(955,330,45);

setcolor(BLUE);

//ENLACES DE PRIMER A SEGUNDO NIVEL

setcolor(RED);

line(955,380,167,480);

line(955,380,362,480);

line(955,380,557,480);

line(955,380,752,480);

line(955,380,947,480);

line(955,380,1142,480);

line(955,380,1357,480);

line(955,380,1532,480);

setcolor(BLUE);

//BASE DE SEGUNDO NIVEL

//Opcion No.1

line(77,540,257,540);

line(77,600,257,600);

line(137,480,137,660);

line(197,480,197,660);

setcolor(BROWN);

circle(107,510,25);

setcolor(BLUE);

//Opcion No.2

line(272,540,452,540);

line(272,600,452,600);

line(332,480,332,660);

line(392,480,392,660);

setcolor(BROWN);

circle(362,510,25);

setcolor(BLUE);

//Opcion No.3

line(467,540,647,540);

line(467,600,647,600);

line(527,480,527,660);

line(587,480,587,660);

setcolor(BROWN);

circle(617,510,25);

setcolor(BLUE);

//Opcion No.4

line(662,540,842,540);

line(662,600,842,600);

line(722,480,722,660);

line(782,480,782,660);

setcolor(BROWN);

circle(692,570,25);

setcolor(BLUE);

//Opcion No.5

line(857,540,1037,540);

line(857,600,1037,600);

line(917,480,917,660);

line(977,480,977,660);

setcolor(BROWN);

circle(947,570,25);

setcolor(BLUE);

//Opcion No.6

line(1052,540,1232,540);

line(1052,600,1232,600);

line(1112,480,1112,660);

line(1172,480,1172,660);

setcolor(BROWN);

circle(1202,570,25);

setcolor(BLUE);

//Opcion No.7

line(1247,540,1427,540);

line(1247,600,1427,600);

line(1307,480,1307,660);

line(1367,480,1367,660);

setcolor(BROWN);

circle(1277,630,25);

setcolor(BLUE);

//Opcion No.8

line(1442,540,1622,540);

line(1442,600,1622,600);

line(1502,480,1502,660);

line(1562,480,1562,660);

setcolor(BROWN);

circle(1532,630,25);

setcolor(BLUE);

setcolor(BROWN);

outtextxy(97,670,"Opcion No.1");

outtextxy(292,670,"Opcion No.2");

outtextxy(487,670,"Opcion No.3");

outtextxy(682,670,"Opcion No.4");

outtextxy(877,670,"Opcion No.5");

outtextxy(1072,670,"Opcion No.6");

outtextxy(1267,670,"Opcion No.7");

outtextxy(1462,670,"Opcion No.8");

}

if(cont==9){////////////////////////////////////////////////////////////////

//Base de PRIMER NIVEL

line(905,80,905,380);

line(1005,80,1005,380);

line(805,180,1105,180);

line(805,280,1105,280);

setcolor(BROWN);

circle(1055,330,45);

setcolor(BLUE);

//ENLACES DE PRIMER A SEGUNDO NIVEL

setcolor(RED);

line(955,380,167,480);

line(955,380,362,480);

line(955,380,557,480);

line(955,380,752,480);

line(955,380,947,480);

line(955,380,1142,480);

line(955,380,1357,480);

line(955,380,1532,480);

setcolor(BLUE);

//BASE DE SEGUNDO NIVEL

//Opcion No.1

line(77,540,257,540);

line(77,600,257,600);

line(137,480,137,660);

line(197,480,197,660);

setcolor(BROWN);

circle(107,510,25);

setcolor(BLUE);

//Opcion No.2

line(272,540,452,540);

line(272,600,452,600);

line(332,480,332,660);

line(392,480,392,660);

setcolor(BROWN);

circle(362,510,25);

setcolor(BLUE);

//Opcion No.3

line(467,540,647,540);

line(467,600,647,600);

line(527,480,527,660);

line(587,480,587,660);

setcolor(BROWN);

circle(617,510,25);

setcolor(BLUE);

//Opcion No.4

line(662,540,842,540);

line(662,600,842,600);

line(722,480,722,660);

line(782,480,782,660);

setcolor(BROWN);

circle(692,570,25);

setcolor(BLUE);

//Opcion No.5

line(857,540,1037,540);

line(857,600,1037,600);

line(917,480,917,660);

line(977,480,977,660);

setcolor(BROWN);

circle(947,570,25);

setcolor(BLUE);

//Opcion No.6

line(1052,540,1232,540);

line(1052,600,1232,600);

line(1112,480,1112,660);

line(1172,480,1172,660);

setcolor(BROWN);

circle(1202,570,25);

setcolor(BLUE);

//Opcion No.7

line(1247,540,1427,540);

line(1247,600,1427,600);

line(1307,480,1307,660);

line(1367,480,1367,660);

setcolor(BROWN);

circle(1277,630,25);

setcolor(BLUE);

//Opcion No.8

line(1442,540,1622,540);

line(1442,600,1622,600);

line(1502,480,1502,660);

line(1562,480,1562,660);

setcolor(BROWN);

circle(1532,630,25);

setcolor(BLUE);

setcolor(BROWN);

outtextxy(97,670,"Opcion No.1");

outtextxy(292,670,"Opcion No.2");

outtextxy(487,670,"Opcion No.3");

outtextxy(682,670,"Opcion No.4");

outtextxy(877,670,"Opcion No.5");

outtextxy(1072,670,"Opcion No.6");

outtextxy(1267,670,"Opcion No.7");

outtextxy(1462,670,"Opcion No.8");

}

sprintf(ptb,"%d",contTableros-1);

outtextxy(180,50,ptb);

delay(9000);

closegraph();

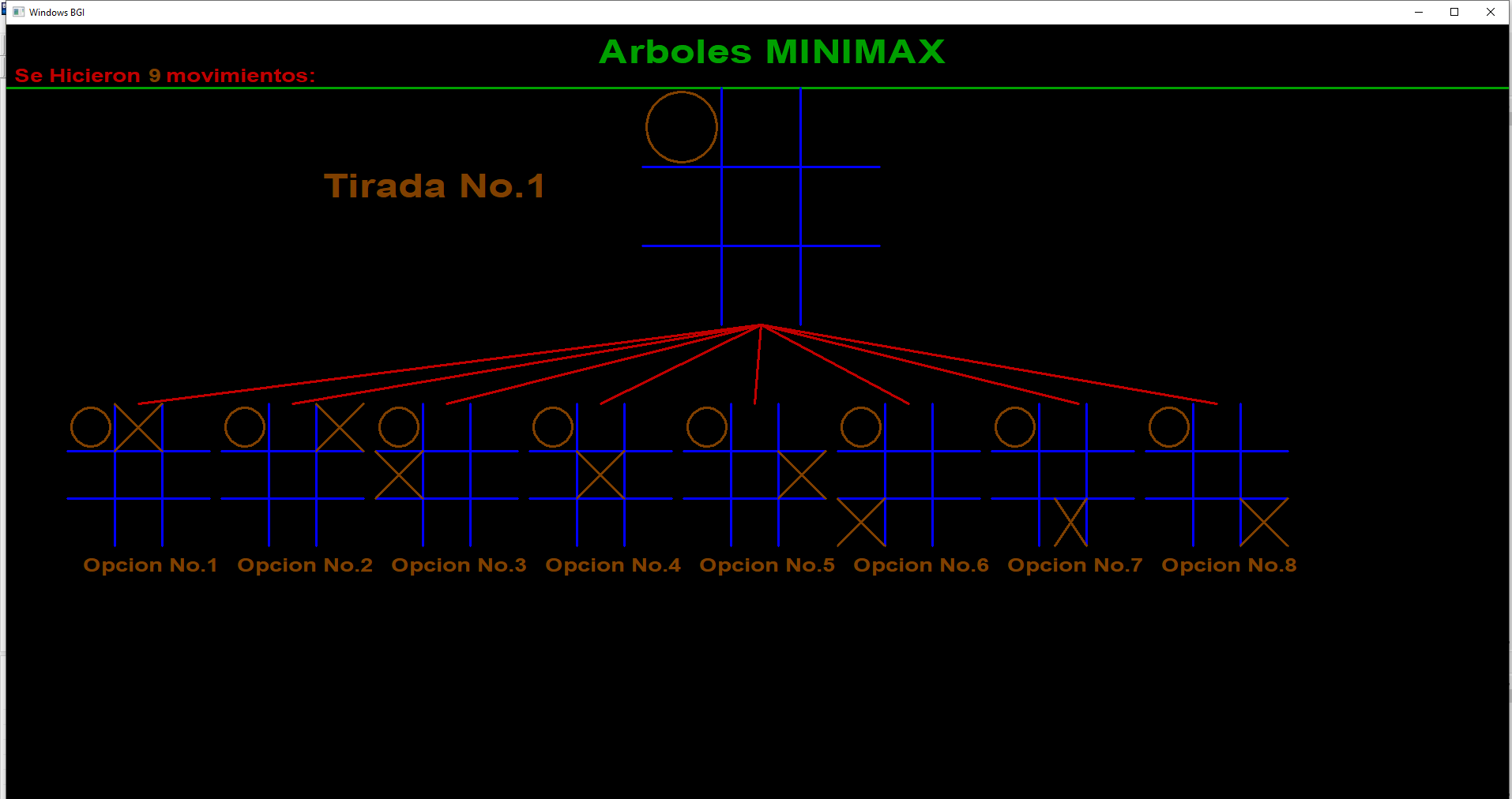
}

## Imágenes de Funcionamiento

### Menú Inicial

## 

## Programa por defecto



# 

# Conclusion.

Es de esta forma que podemos concluir que todos los que nos podemos encontrar en los diversos programas que realizamos, tienen diversas formas de resolverse sin perder de vista que cada una de estas formas tienen diversas desventajas y ventajas en su haber, parte de esto es el gran trabajo del programador descubrir cuáles son y de qué manera utilizarlas en su beneficio o como herramienta, para hacer de su trabajo más fácil sencillo y eficiente, pues el código por sí mismo es solo código, es la lógica del programador la que logra que todo forme parte de un ciclo en el cual se realizan diversas partes y tareas específicas.

# Referencias Bibliográficas:

**[1]:**

Geeky Theory, & Geeky Theory. (2020). ¿Qué es la recursividad? Retrieved November 24, 2020, from Geeky Theory website: https://geekytheory.com/que-es-la-recursividad

‌**[2]:**

I. Bruno and L. Takeyas, “Algoritmo Minimax ALGORITMO MINIMAX” [Online]. Available: http://www.itnuevolaredo.edu.mx/takeyas/Apuntes/Inteligencia%20Artificial/Apuntes/IA/Minimax.pdf. [Accessed: 13-Jan-2021]

‌**[3]:**

su, “El algoritmo Minimax y su aplicación en un juego,” *DevCode Tutoriales*, Jun-2016. [Online]. Available: https://devcode.la/tutoriales/algoritmo-minimax/. [Accessed: 13-Jan-2021]

‌