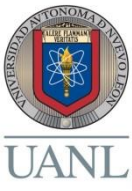

Universidad Autónoma de Nuevo León

FCFM

Inteligencia Artificial

Tarea 5 – Python

Equipo:

Alexis Emiliano Eguia Sifuentes	1861817
Leonardo Martin Vázquez Cruz	1679807
Bryan Aguirre Tudon	1735839

Definición de la tarea:

```
"""
Pseodocodigo
funcion Minimax(S, max=true)
    if s es terminal:
            retirn f(s)
    if Max:
        v = -infinito
        for each hijo en S:
            v = max(v, Minimax(hijo, False))
        retuir v
    else:
        v = inifinito
        for each hijo en S
        v = min(v,MiniMax(hijo,True))
        return v
#Evaluar primero el nodo(Hoja,Hijo,Inicial)
"""
```

Programar el algoritmo minimax para el problema tic-tac-toe (El juego del gato). Como les expliqué en el salón, pueden hacerlo de diferentes maneras:
1) Como un algoritmo que sugiere al jugador que movimiento realizar.
2) Un algoritmo con el que puedan jugar, es decir, ustedes hacen un movimiento y el algoritmo usando minimax, decide el suyo.
Para cualquiera de los dos casos tienen dos formas también de resolverlo:
1) Expanden todo el árbol y le asignan 1 si es un estado ganador, 0 si es un empate, -1 si es un estado perdedor.
2) Pueden expandir solo cierta cantidad de estados y evaluar como está el tablero.

La tarea es en equipo de no más de 3 personas, de igual forma se va a entregar un reporte explicando su algoritmo, pseudocódigos y experimentos realizados, así como una conclusión de cuál fue mejor.

Programa(Interfaz Grafica)

```python
from tkinter import *
from frames import *
from tkinter import messagebox

ventana = Tk()                      # VENTANA  RAIZ
ventana.geometry("700×600")         # DIMENCIONES
ventana.title("Tres en raya")       # TITULO
ventana.resizable(0,0)              # NO REDIMENZIONABLE

cargar_ventana(ventana)             # CARGAMOS LOS OBJETOS

ventana.mainloop()                  # MOSTRAMOS LA PANTALLA
```

Nosotros agregamos como extra, el colocarle una interfaz grafica para que sea mas cómodo el manejó del TABLERO del juego TIC TAC TOE

Clase Nodo

Clase NODO controla nuestro árbol de posibilidades aplicando el algoritmo MINMAX

```python
from random import randint
from principales import evaluar_tablero
class Nodo():
    def __init__(self, tablero,turno):
        self.tablero = tablero
        self.turno = turno
        self.casillas_libres = []
        self.lista_relaciones = []
        self.estado = evaluar_tablero(self.tablero)
        self.valor = ""
        if turno == 'O':
            self.maximo = True
        else:
            self.maximo = False

        indice = 0
        for casilla in tablero:
            if casilla == '.':
                self.casillas_libres.append(indice)
            indice += 1

        if self.estado == -2:
            for indice in self.casillas_libres:
                tablero_aux = []
                for casilla in tablero:
                    tablero_aux.append(casilla)
                tablero_aux.insert(indice,turno)
                tablero_aux.pop(indice+1)
                if turno == "X":
                    turno_aux = 'O'
                else:
                    turno_aux = 'X'

self.lista_relaciones.append(Nodo(tablero_aux,turno_aux))
        if len(self.lista_relaciones) != 0:
            if self.maximo:
                self.nodo_optimo = self.max()
                self.valor = self.nodo_optimo.getValor()
            else:
                self.nodo_optimo = self.min()
                self.valor = self.nodo_optimo.getValor()
        else:
            self.valor = self.getEstado()


    #Metodos Get
    def getTablero(self):
        return self.tablero
    def getListaRelaciones(self):
        return self.lista_relaciones
    def getTurno(self):
        return self.turno
    def getCasillasLibres(self):
        return self.casillas_libres
    def getEstado(self):
        return self.estado
    def getValor(self):
        return self.valor
    def getNodoOptimo(self):
        return self.nodo_optimo


    #Metodos Set
    def setTablero(self,tablero):
        self.tablero = tablero
    def setListaRelaciones(self,lista):
        self.lista_relaciones = lista
    def setTurno(self,turno):
        self.turno = turno

    #Metodos
    def max(self):
        long = len(self.lista_relaciones)-1
        maximo = self.lista_relaciones[randint(0,long)]
        for nodo_hijo in self.lista_relaciones:
            if nodo_hijo.valor > maximo.valor:
                maximo = nodo_hijo
        return maximo
    def min(self):
        long = len(self.lista_relaciones)-1
        minimo = self.lista_relaciones[randint(0,long)]
        for nodo_hijo in self.lista_relaciones:
            if nodo_hijo.valor < minimo.valor:
                minimo = nodo_hijo
        return minimo
```

En el contructor de la clase damos como parámetro el tablero y el turno del jugador y definimos atributos del nodo

Observamos cuales son las casillas vacías

Cambiamos de turno y creamos los nodos hijos del tablero que se paso como parámetro en la clase

Métodos Get, los cuales nos facilitan manipular los atributos de la clase

Métodos SET, los cuales nos permiten actualizar los atributos importantes

Método **MINMAX**

MAX -> Calcula la mejor jugada de la IA

MIN -> Calcula la mejor jugada del PLAYER

Evaluar TABLERO

```python
def evaluar_tablero(tablero):
    #return -1 cuando ganas
    #return 1 cuando pierdes
    #return 0 cuando es empate
    #return -2 NO es nodo terminal
    c_central = tablero[4]

    if c_central != ".":
        #Verificamos si en las digonales se ha ganado
        if tablero[0] == c_central and tablero[8] == c_central or tablero[6] == c_central and tablero[2] == c_central :
            if c_central == 'X':
                return -1
            else:
                return 1

        #Verificamos si en la fila y columna que contenga c_central se ha ganado
        if tablero[1] == c_central and tablero[7] == c_central or tablero[3] == c_central and tablero[5] == c_central:
            if c_central == 'X':
                return -1
            else:
                return 1

    #Verificamos si en las filas y columnas restantes se ha ganado
    c_esquina1 = tablero[0]
    if c_esquina1 != '.':
        if tablero[1] == c_esquina1 and tablero[2] == c_esquina1 or tablero[3] == c_esquina1 and tablero[6] == c_esquina1:
            if c_esquina1 == 'X':
                return -1
            else:
                return 1

    c_esquina2 = tablero[8]
    if c_esquina2 != '.':
        if tablero[2] == c_esquina2 and tablero[5] == c_esquina2 or tablero[6] == c_esquina2 and tablero[7] == c_esquina2:
            if c_esquina2 == 'X':
                return -1
            else:
                return 1

    #Verificamos si NO es un nodo terminal
    for i in range(0,9):
        if tablero[i] == '.':
            return -2

    #Si no se cumple ningun caso anterior entonces es  Empate
    return 0
```

La función retorna los 4 posibles casos:

- Ganar(Que el jugador gane)
- Perder(Que el jugador pierda)
- Empate(No gana la IA ni el jugador)
- En curso(La partida aun no concluye)

Codigo Intefaz Grafica

```python
from tkinter import *
from tkinter import messagebox
from principales import evaluar_tablero,imprimir_tablero
from nodo import Nodo

class Marcador():

    def __init__(self):
        self.victorias=0
        self.derrotas=0
        self.empates=0

    def getVic(self):
        return self.victorias
    def getDer(self):
        return self.derrotas
    def getEmp(self):
        return self.empates

    def incVic(self):
        self.victorias += 1
    def incDer(self):
        self.derrotas += 1
    def incEmp(self):
        self.empates += 1
def incVic(texto):
    marc.incVic()
    texto.config(text=f"Victorias: {marc.getVic()}")
def incDer(texto):
    marc.incDer()
    texto.config(text=f"Derrotas: {marc.getDer()}")
def incEmp(texto):
    marc.incEmp()
    texto.config(text=f"Empates: {marc.getEmp()}")
marc=Marcador()
v=0
e=0
d=0
lista=[]
def cargar_ventana(ventana):
    # CREAMOS EL CONTENEDOR
    encabezado = Frame(ventana, width=700, height=120)

    # AGREGAMOS ESTILOS AL CONTENEDOR
    encabezado.config(bg="Black")

    # OBJETOS DEL CONTENEDOR
    titulo = Label(encabezado,width=12)
    titulo.config(
        text="Tres en raya",
        bg="Black",
        fg="Yellow",
        font=('Stencil',40)
    )
    titulo.pack()

    # EMPAQUETAMOS CONTENEDOR
    encabezado.pack(fill=X, expand=True, side=TOP, anchor=N)
    encabezado.pack_propagate(False)


    #FUENTE Y TAMAÑO DE LA LETRA
    fuente = ('Fixedsys',18)
    fondo = "Black"
    color = "lime"


    # CREAMOS EL CONTENEDOR
    marcador = Frame(encabezado, width=700, height=50)

    # AGREMOS ESTILO AL CONTENEDOR
    marcador.config(bg=fondo)

    # OBJETOS DEL CONTENEDOR
    victorias = Label(marcador)
    empates = Label(marcador)
    derrotas = Label(marcador)

    # AGREGAMOS ESTILOS A LOS OBJETOS
    victorias.config(
        text=f"Victoria: {marc.getVic()}",
        width=10,
        bg=fondo,
        fg=color,
        font=fuente
    )
    empates.config(
        text=f"Empates: {marc.getEmp()}",
        width=10,
        bg=fondo,
        fg=color,
        font=fuente
    )
    derrotas.config(
        text=f"Derrotas: {marc.getDer()}",
        width=10,
        bg=fondo,
        fg=color,
        font=fuente
    )
```

```python
# EMPAQUTAMOS LOS OBJETOS AL CONTENEDOR
    victorias.pack(side=LEFT, fill=X, expand=True)
    empates.pack(side=LEFT, fill=X, expand=True)
    derrotas.pack(side=LEFT, fill=X, expand=True)

    # EMPAQUETAMOS EL CONTENEDOR
    marcador.pack(fill=X, expand=True, side=TOP, anchor=N)
    marcador.pack_propagate(False)


    # FONDO DEL CONTENEDOR(menu)
    fondo="lightBlue"
    # FUENTE
    fuente =  "Fixedsys"

    # CREAMOS EL CONTENEDOR
    menu = Frame(ventana)

    # AGREGAMOS ESTILO AL CONTENEDOR
    menu.config(
        width=200,
        height=480,
        bg=fondo
    )

    # OBJETOS DEL CONTENEDOR
    titulo = Label(menu)
    empezar = Button(menu)
    reiniciar = Button(menu)
    pregunta = Label(menu)
    respuesta1 = Radiobutton(menu)
    respuesta2 = Radiobutton(menu)
    void = Label(menu)
    void1 = Label(menu)
    void2 = Label(menu)
    opcion = StringVar()
    opcion.set(None)

    def f_reiniciar(tablero):
        for casilla in tablero:
            casilla.config(
                text=" ",
                fg="Black"
            )

    # AGREGAR ESTILOS A LOS OBJETOS
    titulo.config(
        text="Menu Principal",
        bg=fondo,
        font=('fuente',17)
    )
    empezar.config(
        text="Empezar",
        font=(fuente,16),
        padx=10,
        pady=10,
    )
    reiniciar.config(
        text="Reiniciar",
        font=(fuente,16),
        command=lambda:f_reiniciar(lista),
        padx=10,
        pady=10
    )
    pregunta.config(
        text="¿Quien iniciara\nla partida?",
        font=(fuente,16),
        padx=10,
        bg=fondo
    )
    respuesta1.config(
        text="Player",
        value="Player",
        variable=opcion,
        font=(fuente,16),
        bg=fondo
    )
    respuesta2.config(
        text="Maquina",
        value="Maquina",
        variable=opcion,
        font=(fuente,16),
        bg=fondo
    )
    void.config(
        text=" ",
        font=('Mistral',1),
        height=1,
        bg=fondo
        )
    void1.config(
        text=" ",
        font=('Mistral',1),
        height=1,
        bg=fondo
        )
    void2.config(
        text=" ",
        font=('Mistral',12),
        height=1,
        bg=fondo
        )
```

```python
# EMPAQUETAMOS LOS OBJETOS
    titulo.pack()
    void.pack()
    empezar.pack()
    void1.pack()
    reiniciar.pack()
    void2.pack()
    pregunta.pack()
    respuesta1.pack()
    respuesta2.pack()

    # EMPAQUETAMOS EL CONTENEDOR
    menu.place(x=0,y=120)
    menu.pack_propagate(False)


    #FUENTE Y TAMAÑO DE LA LETRA(tablero)
    fuente=('Fixedsys',50)

    # FONDO DEL CONTENEDOR(tablero)
    fondo="#31FA31"

    # CREAMOS EL CONTENEDOR
    tablero = Frame(ventana)

    # AGREGAMOS LOS ESTILOS AL CONTENEDOR
    tablero.configure(
        width=500,
        height=480,
        bd=15,
        bg=fondo
    )

    # OBJETOS DEL CONTENEDOR
    fila1 = Frame(tablero)
    fila2 = Frame(tablero)
    fila3 = Frame(tablero)
    casilla1 = Button(fila1)
    casilla2 = Button(fila1)
    casilla3 = Button(fila1)
    casilla4 = Button(fila2)
    casilla5 = Button(fila2)
    casilla6 = Button(fila2)
    casilla7 = Button(fila3)
    casilla8 = Button(fila3)
    casilla9 = Button(fila3)
    void = Label(tablero)

    # AGREGAMOS LOS ESTILOS A LOS OBJETOS
    casilla1.config(
        text=" ",
        width=3,
        height=1,
        padx=10,
        pady=10,
        font=fuente,
        relief=SOLID,
        bg=fondo,
        fg="Black",
        command=lambda: colocar_ficha(casilla1,opcion,victorias,derrotas,empates)
    )
    casilla2.config(
        text=" ",
        width=3,
        height=1,
        padx=10,
        pady=10,
        font=fuente,
        relief=SOLID,
        bg=fondo,
        fg="Black",
        command=lambda: colocar_ficha(casilla2,opcion,victorias,derrotas,empates)
    )
    casilla3.config(
        text=" ",
        width=3,
        height=1,
        padx=10,
        pady=10,
        font=fuente,
        relief=SOLID,
        bg=fondo,
        fg="Black",
        command=lambda: colocar_ficha(casilla3,opcion,victorias,derrotas,empates)
    )
    casilla4.config(
        text=" ",
        width=3,
        height=1,
        padx=10,
        pady=10,
        font=fuente,
        relief=SOLID,
        bg=fondo,
        fg="Black",
        command=lambda: colocar_ficha(casilla4,opcion,victorias,derrotas,empates)
    )
    casilla5.config(
        text=" ",
        width=3,
        height=1,
        padx=10,
        pady=10,
        font=fuente,
        relief=SOLID,
        bg=fondo,
        fg="Black",
        command=lambda: colocar_ficha(casilla5,opcion,victorias,derrotas,empates)
    )
    casilla6.config(
        text=" ",
        width=3,
        height=1,
        padx=10,
        pady=10,
        font=fuente,
        relief=SOLID,
        bg=fondo,
        fg="Black",
        command=lambda: colocar_ficha(casilla6,opcion,victorias,derrotas,empates)
    )
    casilla7.config(
        text=" ",
        width=3,
        height=1,
        padx=10,
        pady=10,
        font=fuente,
        relief=SOLID,
        bg=fondo,
        fg="Black",
        command=lambda: colocar_ficha(casilla7,opcion,victorias,derrotas,empates)
    )
    casilla8.config(
        text=" ",
        width=3,
        height=1,
        padx=10,
        pady=10,
        font=fuente,
        relief=SOLID,
        bg=fondo,
        fg="Black",
        command=lambda: colocar_ficha(casilla8,opcion,victorias,derrotas,empates)
    )
    casilla9.config(
        text=" ",
        width=3,
        height=1,
        padx=10,
        pady=10,
        font=fuente,
        relief=SOLID,
        bg=fondo,
        fg="Black",
        command=lambda: colocar_ficha(casilla9,opcion,victorias,derrotas,empates)
    )
    void.config(
        text="",
        font=('Mistral',12),
        width=1,
        height=60,
        bg=fondo
        )
```

```
    # EMPAQUETAMOS LOS OBJETOS
    void.pack(side=LEFT,anchor=N)
    fila1.pack()
    fila2.pack()
    fila3.pack()
    casilla1.pack(side=LEFT, anchor=N)
    casilla2.pack(side=LEFT, anchor=N)
    casilla3.pack(side=LEFT, anchor=N)
    casilla4.pack(side=LEFT, anchor=N)
    casilla5.pack(side=LEFT, anchor=N)
    casilla6.pack(side=LEFT, anchor=N)
    casilla7.pack(side=LEFT, anchor=N)
    casilla8.pack(side=LEFT, anchor=N)
    casilla9.pack(side=LEFT, anchor=N)


    #LISTA(tablero)
    lista.append(casilla1)
    lista.append(casilla2)
    lista.append(casilla3)
    lista.append(casilla4)
    lista.append(casilla5)
    lista.append(casilla6)
    lista.append(casilla7)
    lista.append(casilla8)
    lista.append(casilla9)

    # EMPAQUETAMOS EL CONTENEDOR
    tablero.place(x=200,y=120)
    tablero.pack_propagate(False)

def leer_tablero(lista):
    tablero=[]
    for cas in lista:
        c=cas['text']
        c=c.upper()
        if c != ' ':
            tablero.append(c)
        else:
            tablero.append('.')
    return tablero
def
colocar_ficha(casilla,opcion,victorias,derrotas,empates):
    if opcion.get() != 'None':
        tablero = leer_tablero(lista)
        vacio=True
        for cas in tablero:
            if cas != '.':
                vacio = False
                break

        #COLOCAMOS FICHA
        if vacio and opcion.get()=='Maquina':
            if evaluar_tablero(tablero) == -2:
                nodo_inicial = Nodo(tablero,"O")
                Nodo_optimo = nodo_inicial.nodo_optimo
                tablero_nodo_optimo =
Nodo_optimo.getTablero()
                casilla_optima = 0
                for casilla_i in tablero:
                    if tablero[casilla_optima] !=
tablero_nodo_optimo[casilla_optima]:
                        break
                    casilla_optima += 1
                casilla = lista[casilla_optima]
                casilla.config(
                    text="o",
                    fg="Blue"
                )
                tablero = leer_tablero(lista)
                imprimir_tablero(tablero)
        else:
            if evaluar_tablero(tablero)==-2 and
casilla['text']==' ':
                casilla.config(
                    text="x",
                    fg="red"
                )
                tablero=leer_tablero(lista)
                resultado=evaluar_tablero(tablero)

                #Creamos el arbol y buscamos la mejor
jugada para la IA
                if resultado == -2:
                    nodo_inicial = Nodo(tablero,"O")
                    Nodo_optimo = nodo_inicial.nodo_optimo
                    tablero_nodo_optimo =
Nodo_optimo.getTablero()
                    casilla_optima = 0
                    for casilla_i in tablero:
                        if tablero[casilla_optima] !=
tablero_nodo_optimo[casilla_optima]:
                            break
                        casilla_optima += 1
                    imprimir_tablero(tablero)
                    casilla = lista[casilla_optima]
                    casilla.config(
                        text="o",
                        fg="Blue"
                    )
    else:
        messagebox.showinfo("Partida","Selecciona quien
iniciara la partida!")

    tablero_l = leer_tablero(lista)
    resultado=evaluar_tablero(tablero_l)
    if resultado==-1: #ganar
        messagebox.showinfo("Resultado de la
partida","Felicidades has ganado!")
        incVic(victorias)
    elif resultado==1: #perdiste
        messagebox.showinfo("Resultado de la
partida","Suerte para la proxima!\nPerdiste :(")
        incDer(derrotas)
    elif resultado==0: #empate
        messagebox.showinfo("Resultado de la
partida","Nadie gano\nsera la proxima vez!")
        incEmp(empates)
a1 == 'X':
                return -1
            else:
                return 1

    c_esquina2 = tablero[8]
    if c_esquina2 != '.':
        if tablero[2] == c_esquina2 and tablero[5] ==
c_esquina2 or tablero[6] == c_esquina2 and tablero[7] ==
c_esquina2:
            if c_esquina2 == 'X':
                return -1
            else:
                return 1

    #Verificamos si NO es un nodo terminal
    for i in range(0,9):
        if tablero[i] == '.':
            return -2

    #Si no se cumple ningun caso anterior entonces es
Empate
    return 0
```