In [ ]:

```
from tkinter import *
from Principal import *

class matrizDatos:


    def __init__(self, master):

        frame = Frame(master)
        frame.pack(side=TOP)
        frame.config(bg="seashell4")

        #-------------------------------------------------------------------------
        minMax = ["max","min"]
        self.opcion = StringVar()
        self.opcion.set(minMax[0])
        self.titulo = Label(frame,text="Escoja Máximo o Mínimo:" ,bg="seashell4")
        self.titulo.grid(row=0,sticky=W)
        self.menuOpciones = OptionMenu(frame,self.opcion,*minMax)
        self.menuOpciones.grid(row=0,column=1)
        #-------------------------------------------------------------------------
        self.space = Label(frame,text="Ingrese valores",bg="seashell4")
        self.space.grid(row=2,sticky=W)

        self.varLab = Label(frame,text="Variables",bg="seashell4")
        self.varLab.grid(row=3,sticky=W)
        self.variables = Spinbox(frame,from_=2, to=5,state="readonly",width=10)
        self.variables.grid(row=4,sticky=W)
        #-------------------------------------------------------------------------
        self.resLab = Label(frame,text="Restricciones",bg="seashell4")
        self.resLab.grid(row=5,sticky=W)
        self.restricciones = Spinbox(frame,from_=2, to=5,state="readonly",width=10)
        self.restricciones.grid(row=6,sticky=W)
        #-------------------------------------------------------------------------

        lin = Label(frame,text="Aceptar para continuar",bg="seashell4")
        lin.grid(row=7,sticky=W)

        self.button = Button(frame,text="Aceptar", relief = RAISED,command = lambda:self.fu
        self.button.grid(row=8,sticky=W)



    def funcionObjetivo(self,master,opcion,variables,restricciones,boton):
        boton.destroy()
        vas = int(variables.get())
        res = int(restricciones.get())
        frame2 = Frame(master)
        frame2.pack(side=TOP)
        columCount = 0
        func = Label(frame2,text=opcion.get()+" = ")
        func.grid(row=0,column=columCount)
        columCount+=1

        funcEspacios = []
        funcEspacios.append([])
        for i in range(0,vas):

            cuadrito = Entry(frame2,width=5,relief=RAISED)
```

```
            funcEspacios[0].append(cuadrito)       #necesito control de cuadritos
            cuadrito.grid(row=0,column=columCount)
            columCount+=1

            x = "x"+str(i+1)
            xpos = Label(frame2,text=x)
            xpos.grid(row=0,column=columCount)
            columCount+=1

            if i+1!=vas:
                suma = Label(frame2,text=" + ")
                suma.grid(row=0,column=columCount)
                columCount+=1


        lin = Label(frame2,text="")
        lin.grid(row=9,sticky=W)

        self.buttonx = Button(frame2,text="Aceptar", relief = RAISED,command = lambda:self.
        self.buttonx.grid(row=10,sticky=W)

        # self.button2 = Button(frame2,text="Aceptar", relief = RAISED,command = lambda:sel
        # self.button2.grid(row=10,sticky=W)
        #self.hola(master,opcion,variables,restricciones)

    def restriccionesLlenar(self,master,opcion,variables,restricciones,funcEspacios,buttonx
        buttonx.destroy()
        for p in funcEspacios:
            for q in p:
                q.config(state="readonly")
        frame4 = Frame(master)
        frame4.pack()


        for i in range(0,restricciones):
            columCount=0
            funcEspacios.append([])
            for y in range(0,variables):
                cuadrito = Entry(frame4,width=5,relief=RAISED)
                funcEspacios[i+1].append(cuadrito)       #necesito control de cuadritos
                cuadrito.grid(row=i,column=columCount)
                columCount+=1

                x = "x"+str(y+1)
                xpos = Label(frame4,text=x)
                xpos.grid(row=i,column=columCount)
                columCount+=1

                if y+1!=variables:
                    suma = Label(frame4,text=" + ")
                    suma.grid(row=i,column=columCount)
                    columCount+=1

            simbolo = [">=","<=","="]
            self.simb = StringVar()
            self.simb.set(simbolo[0])
            self.menuOpciones = OptionMenu(frame4,self.simb,*simbolo)
            self.menuOpciones.grid(row=i,column=columCount)
            columCount+=1

            cuadrito = Entry(frame4,width=5,relief=RAISED)
```

```
            funcEspacios[i+1].append(cuadrito)       #necesito control de cuadritos
            cuadrito.grid(row=i,column=columCount)
            columCount+=1

            funcEspacios[i+1].append(self.simb)

        lin = Label(frame4,text="")
        lin.grid(row=9,sticky=W)

        self.button2 = Button(frame4,text="Aceptar", relief = RAISED,command = lambda:self.
        self.button2.grid(row=10,sticky=W)


    def printear(self,master,opcion,variables,restricciones,funcEspacios,button2,frame4):

        button2.destroy()
        simbolos = []
        for s in range(1,len(funcEspacios)):            #simbolos de restricciones
            simbolos.append(funcEspacios[s][-1].get())

        resultado = []
        resultado.append(opcion.get())        #pega la opcion max o min
        resultado.append(str(variables)+","+str(restricciones)) #pega num variablesy restri

        linea = []
        for x in funcEspacios[0]:
            linea.append(x.get())
        resultado.append(linea)         #pega la funcion max o min

        for x in range(1,restricciones+1):
            linea = []
            for y in range(0,len(funcEspacios[x])-1):
                linea.append(funcEspacios[x][y].get())
            linea.append(simbolos[x-1])
            resultado.append(linea)     #pega lineas de la matriz

        resultado = self.estandarizarResultado(resultado)


        frame5 = Frame(master)
        frame5.pack(side=BOTTOM)
        ulabel = Label(frame5)
        ulabel.grid(row=0)

        main(resultado)
        archivo=open("solucionDosFases","r")
        lineas = archivo.readlines()
        lineaUltima = lineas[-1]
        ulabel.config(text=lineaUltima)


    def estandarizarResultado(self,resultadoAux):

        arregloString = []

        arregloString.append(resultadoAux[0])
        arregloString.append(resultadoAux[1])
        for i in range(2, len(resultadoAux)):
            x=""
            for j in range(len(resultadoAux[i])):
                if j < len(resultadoAux[i])-1:
```

```
                    x = x + str(resultadoAux[i][j]) + ','
                else:
                    x = x + str(resultadoAux[i][j])
            arregloString.append(x)
        #print(arregloString)
        return arregloString



root = Tk()
root.geometry("600x400")
root.resizable(True, False)
root.title("Simplex")
root.config(bg="seashell4")
root.iconbitmap("")
matriz = matrizDatos(root)      #objeto matriz
root.mainloop()
```