**INSTITUTO TECNOLÓGICO DE TIJUANA**

**SUBDIRECCIÓN ACADÉMICA**

**DEPARTAMENTO DE SISTEMAS Y COMPUTACIÓN**

**PERIODO:** Agosto-Diciembre 2017

**Carrera:** Ingeniería en Sistemas Computacionales

**Materia:** Ecuaciones Diferenciales

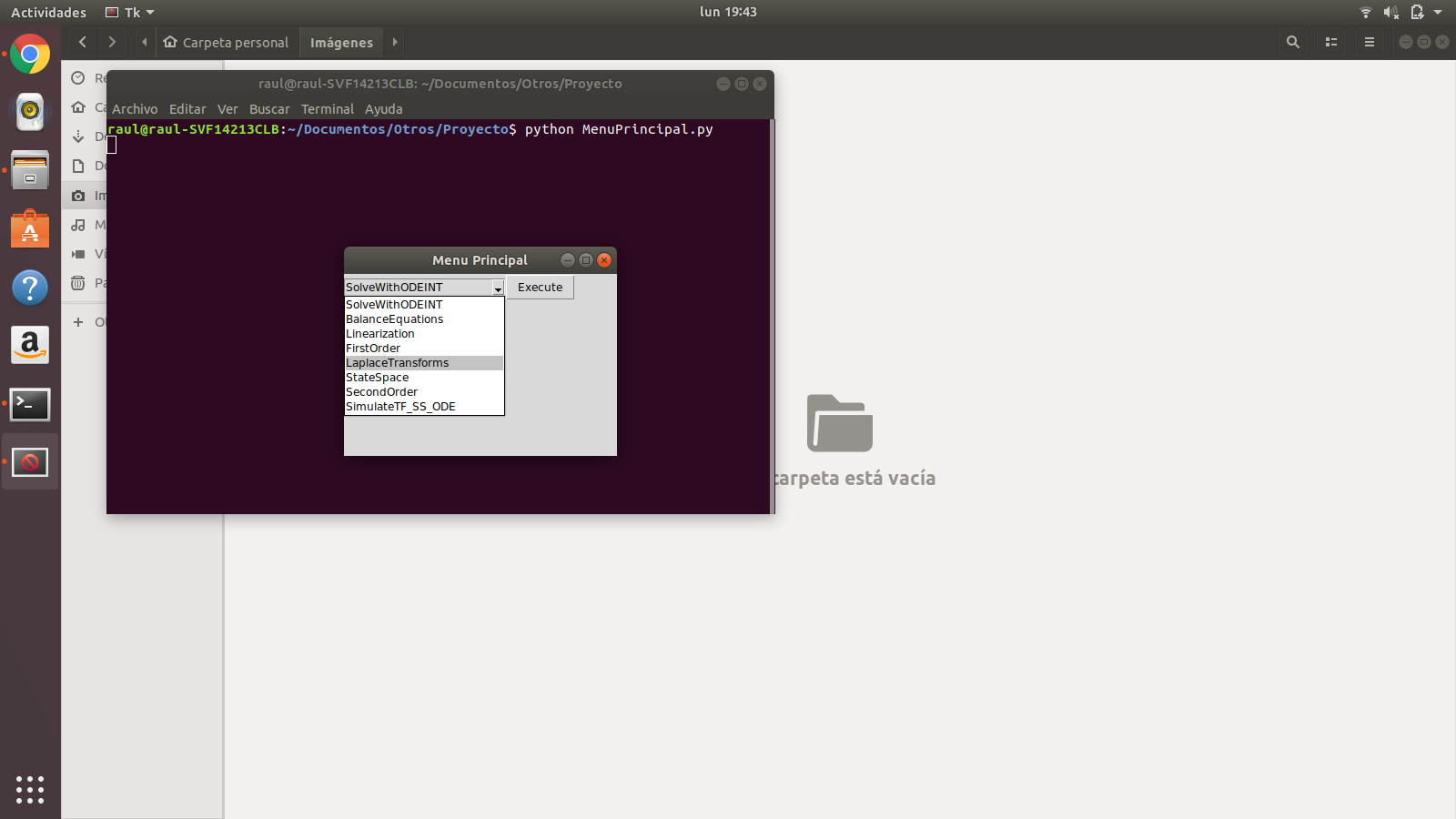
**’’Documentación Programa de Ecuaciones”**

**Docente:** Jorge Luis Herrera Arellano

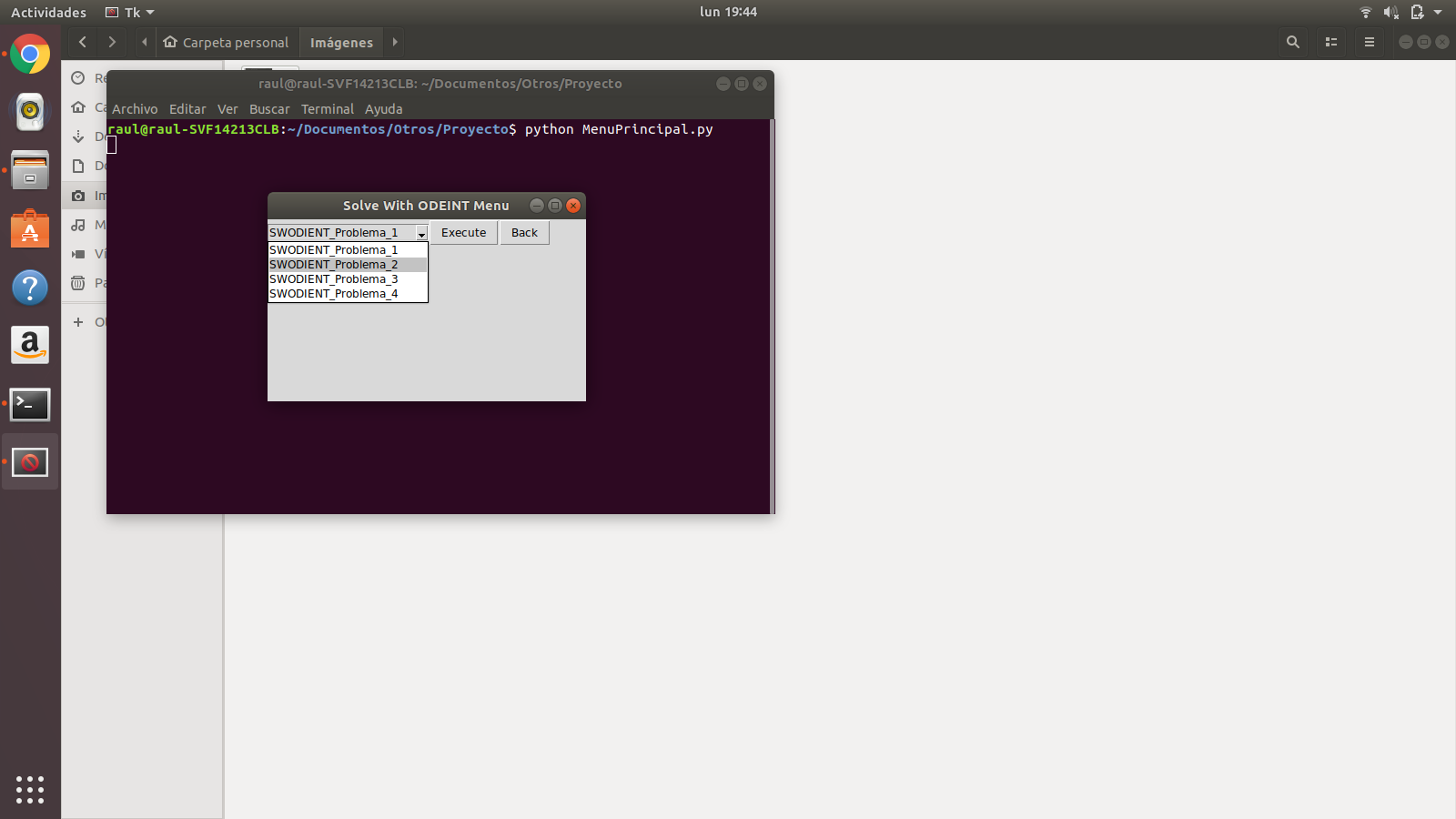
**Aula:** Q205

**Horario:** 2:00 – 3:00 P.M.

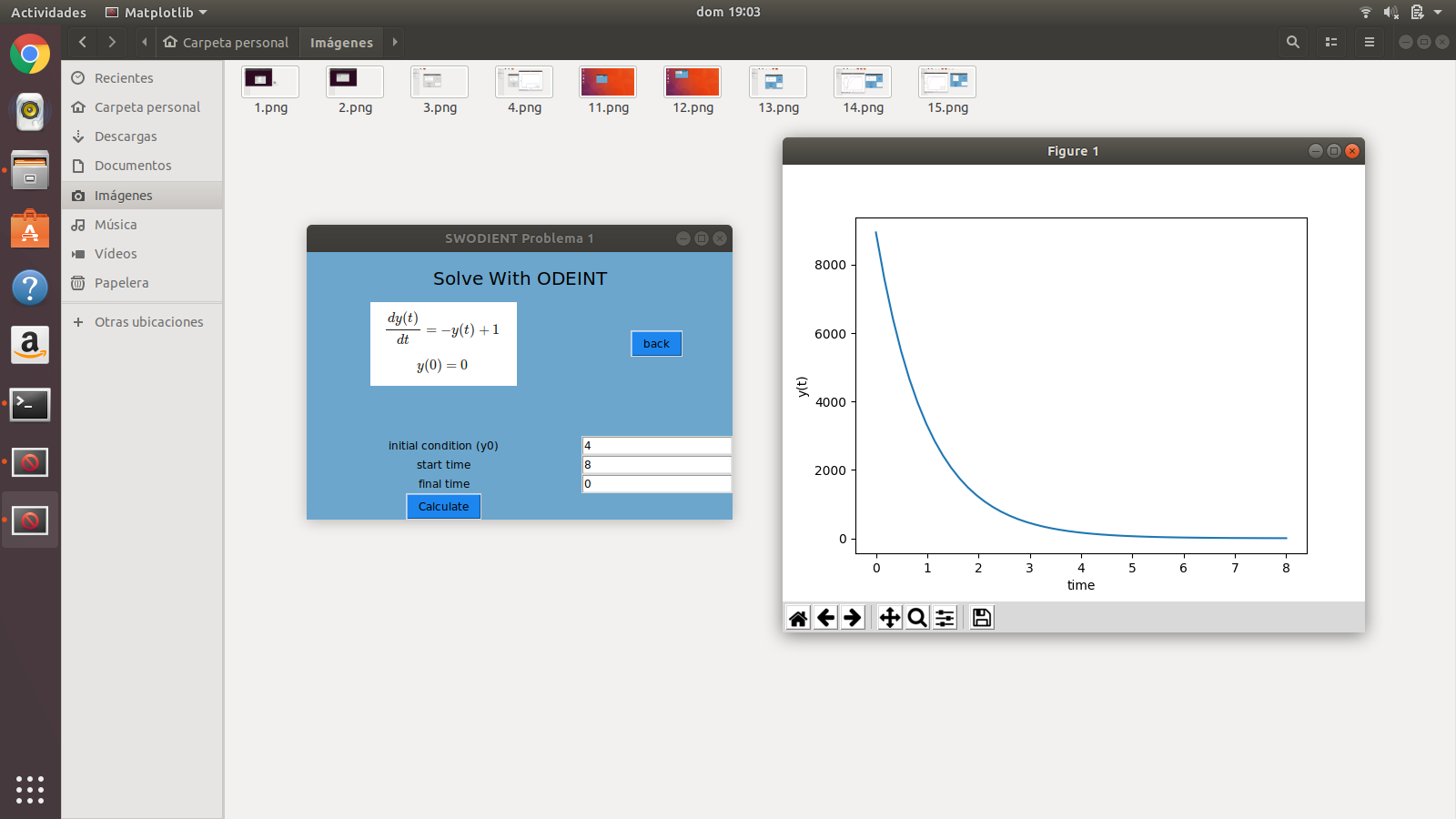
Al ejecutar nuestro programa, muestra la pantalla principal la cual despliega un menú con los diferentes temas que se vieron en la materia de ecuaciones diferenciales:



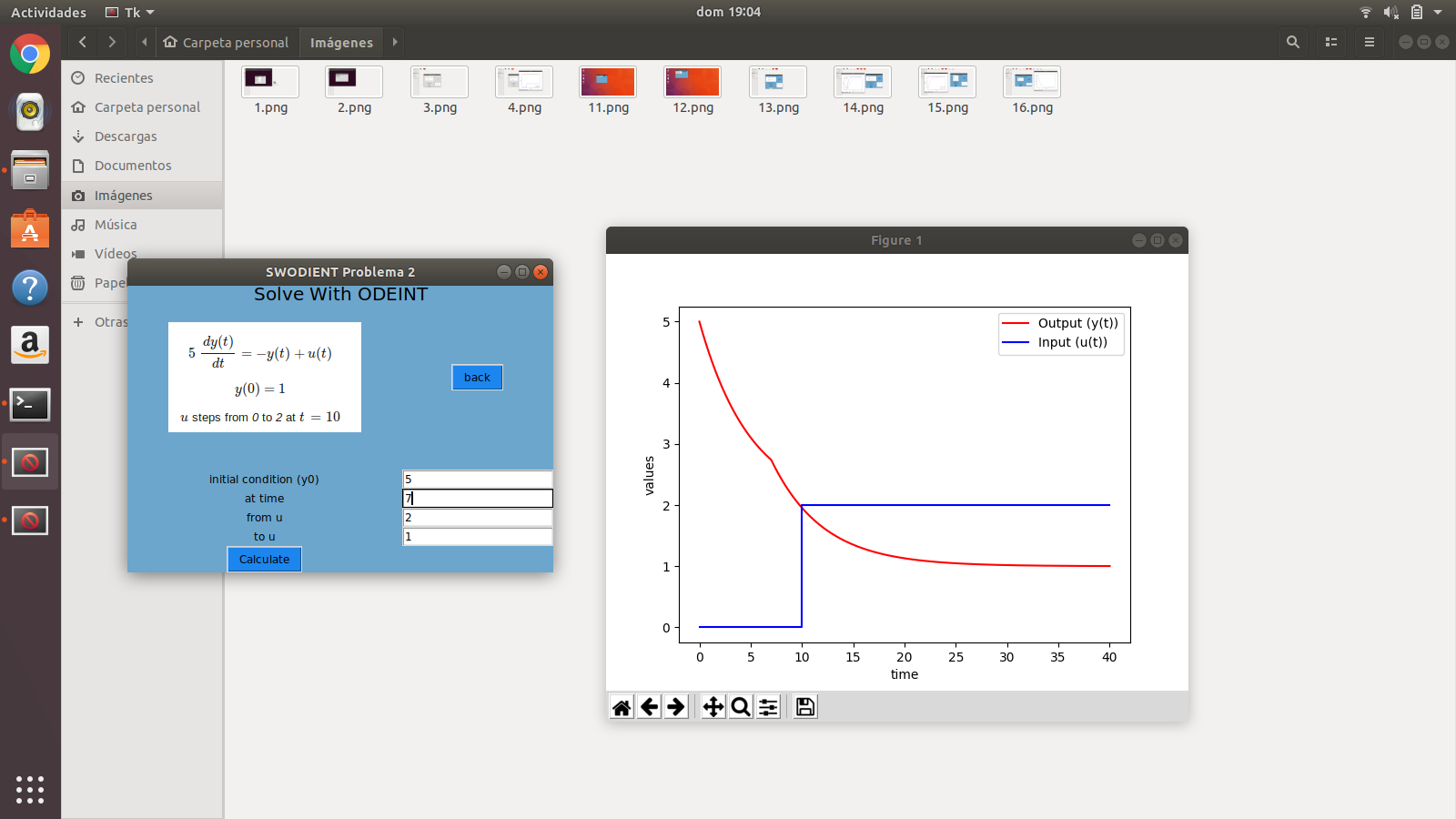
Algunos temas contienen algunos submenús los cuales tienen ecuaciones diferentes y por lo tanto tienen un resultado diferente al momento de resolverlas. En este caso se selecciona un subtema.



Ahora se despliega una nueva pantalla en la cual se muestra la ecuación con la que se resuelve este problema, se ingresan las variables que se piden y posteriormente se da click en el botón calcular, para mostrar la gráfica con los parámetros dados en la ventana anterior.



Para regresar se da click en el botón back , de nuevo se selecciona un nuevo tema en la cual se sigue la misma rutina.



**Código:**

Para poder ejecutar el programa sin errores es necesario instalar estas librerías:

sudo apt install python-pip

sudo apt install python-tk

pip install scipy

pip install numpy

pip install matplotlib

pip install sympy

En nuestro archivo de menú principal tenemos estas librerias en uso:

import Tkinter as tk

import ttk

import os

import time

Posteriormente tenemos la ubicación de donde se manda a llamar los otros archivos que componen todo el programa con el método ejecutar ():

def Ejecutar():

funcion()

os.system('python ~/Documentos/Otros/Proyecto/funciones/'+combo.get()+'.py')

funcion2()

Ahora tenemos nuestro combobox principal en donde se seleccionan los temas:

Primero le decimos a nuestro combo box que seleccione solo un elemento a la vez y le damos los valores que llevara dentro nuestro combo. Posteriormente se le da nombre a la ventana y las medidas correspondientes. Luego el botón que se agregó gracias a la librería tkinter le decimos que ejecute la función de seleccionar.

combo = ttk.Combobox(master, state="readonly")

combo.grid(row=0, column=0)

combo['values']= ('SolveWithODEINT','BalanceEquations','Linearization','FirstOrder','LaplaceTransforms','StateSpace','SecondOrder','SimulateTF\_SS\_ODE')

combo.current(0)

master.title("Menu Principal")

master.geometry("300x200+10+10")

tk.Button(master, text="Execute", command=lambda: Ejecutar()).grid(row=0, column=2)

tk.mainloop()

Tenemos una sub ventana que necesita las siguientes librerías, esta ventana es para seleccionar los subtemas de las diferentes ecuaciones que hay.

from Tkinter import \*

import numpy as np

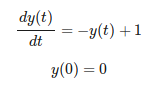
from scipy.integrate import odeint

import matplotlib.pyplot as plt

import Tkinter

from PIL import Image, ImageTk

Ahora estamos en una ventana de las varias funciones que ejecuta el programa es necesario declarar las variables que vamos a utilizar para poder declarar funciones posteriormente. En este caso la fórmula que utilizaremos para resolver esta ecuación será la siguiente:



class val():

def \_\_init\_\_(self,y0, to, tf):

self.y0 = y0

self.to = to

self.tf = tf

def PedirDatos(y0, to, tf):

y0 = float(y0)

to = int(to)

tf = int(tf)

return val(y0, to, tf)

Ahora que ya se declararon las variables necesarias para resolver esta ecuación se necesitan declara funciones para que realicen el procedimiento siguiendo la ecuación:

# function that returns dy/dt

def model(y,t):

dydt = -y + 1.0

return dydt

def SWODIENT\_Problema\_1(y0, to, tf):

# initial condition

va = PedirDatos(y0, to, tf)

y0 = va.y0

# time points

t = np.linspace(va.to,va.tf)

# solve ODE

y = odeint(model,y0,t)

Una vez que se tiene las funciones listas, se muestra la gráfica con los resultados que se ingresaron como parámetros que capturo el usuario a través de nuestras variables. (Es aquí cuando la librería matplot se utiliza)

# plot results

plt.plot(t,y)

plt.xlabel('time')

plt.ylabel('y(t)')

plt.show()

Posteriormente se agregan detalles como ajuste de la imagen, la ruta de donde se obtiene y el título de la ventana, botones, etc.

imagenAnchuraMaxima=300

imagenAlturaMaxima=200

# abrimos una imagen

img = Image.open('/home/raul/Documentos/Otros/Proyecto/imagenes/SWODIENT\_Problema\_1.png')

# modificamos el tamaño de la imagen

img.thumbnail((imagenAnchuraMaxima,imagenAlturaMaxima), Image.ANTIALIAS)

master = Tkinter.Tk()

# titulo de la ventana

master.title("SWODIENT Problema 1")

# Convertimos la imagen a un objeto PhotoImage de Tkinter

tkimage = ImageTk.PhotoImage(img)

# Ponemos la imagen en un Lable dentro de la ventana

label=Tkinter.Label(master, image=tkimage, width=imagenAnchuraMaxima, height=imagenAlturaMaxima)

label.grid(row=0, column=0)

# añadimos un botón para cerrar

atras=Tkinter.Button(master, text="back",command=master.quit)

atras.grid(row=0, column=1)

Finalmente se agrega etiquetas los cuales se les asigna una posición:

Label(master, text="initial condition (y0)").grid(row=2)

Label(master, text="start time").grid(row=3)

Label(master, text="final time").grid(row=4)

Asignación de valores:

v1 = StringVar()

v2 = StringVar()

v3 = StringVar()

e1 = Entry(master, textvariable=v1)

e2 = Entry(master, textvariable=v2)

e3 = Entry(master, textvariable=v3)

Posición:

e1.grid(row=2, column=1)

e2.grid(row=3, column=1)

e3.grid(row=4, column=1)

En esta parte se manda a llamar a la otra ventana en donde aparecerá la gráfica ya que este botón es encargado de procesar todo lo que ya se ha visto arriba.

Button(master, text="Calculate", command=lambda: SWODIENT\_Problema\_1(v1.get(), v2.get(), v3.get())).grid(row=5)

# Mostramos la ventana

master.mainloop()

En teoría básicamente toda la estructura es casi lo mismo solo cambian las variables y funciones dependiendo la ecuación sin embargo esto es la base para poder modificar o editar este código fuente.

***Referencias:***

<http://apmonitor.com/pdc/index.php/Main/SolveDifferentialEquations>

